Bulletin of the Iranian Mathematical Society
https://doi.org/10.1007/s41980-021-00530-w

ORIGINAL PAPER

A Spectral Collocation Method for Computer Virus Spread Case of Delayed Optimal Control Problem

Mehdi Shahini[1] · **Asyieh Ebrahimzadeh**[2] · **Raheleh Khanduzi**[1]

Received: 1 April 2020 / Revised: 25 December 2020 / Accepted: 18 January 2021

Abstract
In this essay, a method is presented for approximating the optimal control problem (OCP) with nonlinear delay differential equations using global collocation at Legendre–Gauss–Radau points. This OCP models the spread of the computer virus. The operational matrices and collocation method together with hybrid Legendre polynomials and block-pulse functions are applied to discretize the model and convert it into a large-scale finite-dimensional nonlinear programming that can be solved by the existing well-developed methods in Matlab software. The numerical simulations show that the proposed collocation method leads to high precision solutions. Finally, the effects of the parameters that appeared in the model are analyzed. The experimental results demonstrate that the strategy for increasing the failure and the retrieval rates and decreasing the epidemic rate and the number of new computers gives a considerable influence on controlling and restraining the propagation of computer viruses.

Keywords Computer virus · Optimal control · Delay differential equation · Operational matrices · Spectral collocation method

Mathematics Subject Classification 49M25 · 49J15

Communicated by Davoud Mirzaei.

✉ Asyieh Ebrahimzadeh
a.ebrahimzadeh@cfu.ac.ir

Mehdi Shahini
mehdi.shahini@gonbad.ac.ir

Raheleh Khanduzi
khanduzi@gonbad.ac.ir

[1] Department of Mathematics and Statistics, Gonbad Kavous University, Gonbad Kavous, Iran

[2] Department of Educational Mathematics, Farhangian University, Tehran, Iran

Published online: 01 April 2021

1 Introduction and Background

A computer virus is a malicious program that can be reproduced and spread to other computers. Computer viruses have been suffered tremendous losses over the past decades. Given the popularity of the internet and wireless networks, the epidemic potential of computer viruses has been extremely increased. Actually, computer viruses have become a great threat to our works and our lives. Dynamic modeling is an essential way of studying the spreading path of a computer virus on the Internet. Since Kephart and White [1] formulated the first spread model of a computer virus, kinds of research in the relevant field have been reported. In the last decade, studies on computer viruses have chiefly targeted on spreading the viruses on fully connected and complex networks. The propagation of viruses on fully connected networks is based on the assumption that any computer is accessible to any other computer across the network. On the other hand, the spread of the virus on complex networks is originated from the fact that the Internet depends strongly on the power-law degree distribution. In industries, the influence of the virus and its damages is on average due to an increase in public awareness of innovation and high technology. Therefore, the issue is still ongoing. Chiefly, mathematical formulas assist in analyzing the persistent states, reducing the propagation of the virus, controlling the spread of the virus, and describing the diffusion model [2–10]. On the other hand, infrastructures and information technology often depend on computer systems and Internet connections. By increasing dependence on computer systems, we discover that cyberattacks are increasing. To prevent this or to significantly reduce the rate of the cyberattack, there is a need for formulating deterministic models based on a control measure to propagate the computer virus.

Mathematical models play a crucial role in providing deep knowledge in many phenomena, including the virus spread and their control. Again, for these years, it has become an essential tool that considers the dynamic behaviors of many computer systems and viruses, so that the appropriate decision against disturbances is adopted. We classified the key characteristics of the literature on the propagation of computer viruses. The encoding plan of existing researches according to the model and problem type is presented in Table 1. The related studies based on the model, problem, network type, and solution approach are shown in Table 2. Indeed, Table 2 displays a stipulated classification of the concerned studies. In Table 2, the mark "✓" indicates that the attribute exists in the literature, while the mark "✗" shows that it does not exist.

There are several types of mathematical models for spreading the virus that can describe the diffusion model. Most of the studies addressed in Table 2 presented differential equations for the mathematical representation of the diffusion model and a few papers proposed optimal control of differential equations. The ODE, PDE, DDE, IDE, SDE, RRE, FDE, and DMRRE play a very important and valuable role in modeling the diffusion phenomena of computer viruses and numerical methods most commonly have been developed for the solution of these equations. The literature (Kephart and White [2], Yuan et al. [5], Ren et al. [19], Mishra and Saini [9], Zhang et al. [20], Barthélemy et al. [21], Shi et al. [22], and Draief et al. [24]) shows outstanding overall performance that comes from using differential equations on modeling virus propagation in information systems.

Table 1 Encoding plan of the available models and problems

Model type	Abbreviation	Problem type	Abbreviation
Susceptible infected	SI	Ordinary differential equation	ODE
Susceptible infected susceptible	SIS	Partial differential equation	PDE
Susceptible infected removed	SIR	Delayed differential equation	DDE
Susceptible infectious recovered susceptible	SIRS	Integro-differential equation	IDE
Susceptible infected external susceptible	SIES	Stochastic differential equation	SDE
Susceptible exposed infectious recovered	SEIR	reaction rate equation	RRE
Susceptible exposed infectious recovered susceptible	SEIRS	Dynamic mean-field reaction rate equation	DMRRE
Susceptible exposed infectious quarantined recovered susceptible	SEIQRS	Functional differential equation	FDE
Susceptible latent breaking susceptible	SLBS		
Susceptible infected countermeasure susceptible	SICS		
SEIRS with delay	SEIRSD		
SLBS with delay	SLBSD		
SEIQRS with delay	SEIQRSD		
SEIRS with stochastic perturbation	SEIRSS		

Relevant studies on the optimal control problems under differential equation systems proposed the spread models of computer virus based on the FDE, DDE, or ODE. Although some studies, i.e., Chen et al. [29], Ren et al. [31], and Zhu et al. [33], developed the controlled DDEs as a computer virus spread model, but up to now, the only work on the OCPs which took into account SEIRSD and DDE was made by Zhu et al. [33]. Since the problem is converted to a large-scale nonlinear programming problem (NLP), many nonlinear optimization solvers can be used to solve it. There is considerable attention given to the development of efficient methods (or hybrid approaches) that can produce high-quality numerical solutions for large-scale problems. Sharma et al. [28] introduced a strong and efficient formulation of the virus spread model for solving the FDE using collocation optimal control methods. The potential benefit of the collocation methods is that they decrease the number of parameters and constraints of the nonlinear problem. Also, they increase the precision of the numerical solution and they give a better mastery over discontinuities of the control functions. Notably, applying the collocation methods provides a large-scale number of variables [35] that results in a large NLP.

The spectral methods have an acceptable and satisfactory performance to solve the differential equations of the virus spread model. They can easily monitor and detect nonlinear interactions that contribute to antivirus effects and reduce system loss. A key advantage of these methods is very-low-phase error. Despite the significant advantages

Table 2 A classification of relevant researches on the model of computer virus spread in the literature

Papers	Model	Problem	Network		Solution approach
			Fully connected	Complex	
Kephart and White [2]	SIS	ODE	✓	✗	Discrete time method
Billings et al. [11]	SIS	ODE	✓	✗	Discrete time method
Ren et al. [3]	SIR	ODE	✓	✗	Numerical method
Zhu et al. [12]	SIR	ODE	✓	✗	*dsolve* function in Matlab
Gan et al. [4]	SIRS	ODE	✓	✗	Discrete time method
Gan et al. [13]	SIRS	ODE	✓	✗	Discrete time method
Gan et al. [14]	SIES	ODE	✓	✗	Lyapunov method
Yuan and Chen [15]	SEIR	ODE	✓	✗	Numerical method
Yuan et al. [5]	SEIR	PDE	✓	✗	Numerical method
Mishra and Pandey [16]	SEIRS	ODE	✓	✗	Runge–Kutta Fehlberg fourth–fifth-order method
Mishra and Saini [6]	SEIQRS	ODE	✓	✗	Numerical method
Yang and Yang [7]	SLBS	ODE	✓	✗	Numerical method
Yang and Yang [17]	SLBS	ODE	✓	✗	Numerical method
Yang and Yang [18]	SICS	ODE	✓	✗	Numerical method
Zhu et al. [8]	SICS	ODE	✓	✗	Lyapunov method
Ren et al. [19]	SEIRSD	DDE	✓	✗	Numerical method
Mishra and Saini [9]	SEIRSD	IDE	✓	✗	Numerical method
Zhang et al. [20]	SEIRSS	SDE	✓	✗	Lyapunov method
Yang and Yang [10]	SEIRSS	SDE	✓	✗	Numerical method

Table 2 continued

Papers	Model	Problem	Network		Solution approach
			Fully connected	Complex	
Barthélemy et al. [21]	SI	RRE	✗	✓	Numerical method
Shi et al. [22]	SIS	DMRRE	✗	✓	Uniform immunization method
d'Onofrio [23]	SIS	RRE	✗	✓	Spectral method
Draief et al. [24]	SIR	FDE	✗	✓	Probabilistic method
Griffin and Brooks [25]	SIR	ODE	✗	✓	Countermeasure competing method
Yang et al. [26]	SLBS	DMRRE	✗	✓	Numerical method
Kazeem et al. [27]	SEIR	OCP and ODE	✓	✗	Fourth-order Runge–Kutta method
Sharma et al. [28]	SEIRSD	OCP and FDE	✗	✓	Collocation method
Chen et al. [29]	SLBSD	OCP and DDE	✓	✗	Numerical method
Darajat et al. [30]	SLBS	OCP and ODE	✓	✗	Forward–backward sweep with Runge–Kutta method
Ren et al. [31]	SEIQRSD	OCP and DDE	✓	✗	Numerical method
Gan et al. [32]	SIES	OCP and ODE	✓	✗	Numerical method
Zhu et al. [33]	SEIRSD	OCP and DDE	✓	✗	Numerical method
Bi et al. [34]	SLBS	OCP and ODE	✓	✗	Forward–backward Euler method
Current study	SEIRSD	OCP and DDE	✓	✗	Spectral collocation method

of the spectral methods in solving computer virus spread models, the application of them is limited to the work of d'Onofrio [23].

We have observed that no work in the literature executes collocation and spectral methods simultaneously. This study proposes a novel hybrid approach based on collocation and spectral methods to solve the OCP containing DDEs and to improve the solution regarding the accuracy and running time. Concerning the SEIRSD model for the virus spread in computer networks, this study generates 243 random numerical examples of the model to consider different levels of the number of new computers, the epidemic rate of the infected computers, the recovered rate of the infected computers, the failure rate, the total population, and the time period. Moreover, there is no work in this field that simultaneously applies Legendre polynomials and block-pulse functions for controlled delayed differential equation systems that increase the computational speed. The following are the main contributions and advantages of this study:

- A collocation approach is developed to convert the desirable model into a finite dimensional mathematical programming problem.
- The utilized operational matrices of differentiation and integration are rather sparse, so the proposed approach is easy to implement.
- The main advantage of hybrid functions introduced in Sect. 3 is their efficiency, because the orders of block-pulse functions and Legendre polynomials are adjustable to obtain a highly accurate numerical solution.
- For the first time in the literature, the proposed method is used to solve the computer virus spread model.

The layout of this study is as follows: Sect. 2 formulates the optimal control problem. In Sect. 3, after introducing the hybrid functions, the operational matrices are given. Section 4 provides a framework to solve the problem formulated in Sect. 2. Section 5 provides numerical simulations and Sect. 6 is devoted to conclusion.

2 Description of Computer Virus Spread Model and Optimal Control Problem

Consider the following optimal control problem:
Problem $\mathcal{A}$:
The objective of this study is to propose an efficient numerical scheme based on operational matrices and collocation approach for solving the following objective functional:

$$\mathcal{J}(\mathcal{I}(t), \mathcal{U}(t)) = \int_0^T \left(\mathcal{I}(t) + \frac{\varepsilon(\mathcal{U}(t))^2}{2} \right) \mathrm{d}t, \tag{2.1}$$

with controlled delayed differential equation systems:

$$\frac{\mathrm{d}\mathcal{S}}{\mathrm{d}t} = (1-p)b - \mu\mathcal{S} - \beta\mathcal{S}(t-\tau_1)\mathcal{I}(t-\tau_1) + \nu\mathcal{R}(t-\tau_2) + \omega\mathcal{U}(t)\mathcal{I}, \tag{2.2a}$$

$$\frac{\mathrm{d}\mathcal{I}}{\mathrm{d}t} = \beta\mathcal{S}(t-\tau_1)\mathcal{I}(t-\tau_1) - (\mu+\gamma+\alpha)\mathcal{I} - \mathcal{U}(t)\mathcal{I}, \tag{2.2b}$$

$$\frac{\mathrm{d}\mathcal{R}}{\mathrm{d}t} = pb + \gamma\mathcal{I} - \mu\mathcal{R} - \nu\mathcal{R}(t-\tau_2) + (1-\omega)\mathcal{U}(t)\mathcal{I}, \tag{2.2c}$$

where b is the number of new computers, p is the immune rate of a new computer, β is the epidemic rate of an infected computer, μ is the failure rate of a computer, ν is the loss rate of safety the of a recovered computer, γ is the retrieval rate of an infected computer, α is the failure rate of an infected computer as a result of the virus activity, and τ_1 and τ_2 are the latent and the immune period, respectively. For this aim, let $[0, T]$ be the time period when a control strategy is planned for the system which presents a control function $\mathcal{U}(t)(0 \le t \le T)$, the budget for getting a great antivirus software on time t. Based on the $\mathcal{U}(t)$ activity, $\omega\mathcal{U}(t)\mathcal{I}$ infected computers are exposed to risk at time t, whereas $(1-\omega)\mathcal{U}(t)\mathcal{I}$ infected computers are recovered at time t, in which $\omega \in [0, 1]$. Our objective is to find the control function $\mathcal{U}(t)$, so that the functional $\mathcal{J}$ given in (2.1) is minimized, where, ε is a trade-off factor. $S(t)$, $I(t)$, and $R(t)$ are the states of the problems where $S(t)$ is the number of healthy computers that are susceptible to infection, $I(t)$ is the already infected computers that can transmit the virus to the healthy ones, and $R(t)$ is the recovered computers that cannot get the virus or transmit it [36,37]. The initial conditions are considered for the model as follows:

$$\begin{aligned}
\mathcal{S}(t) &= S_0(t), \quad -\tau_1 \le t \le 0\\
\mathcal{I}(t) &= I_0(t), \quad -\tau_1 \le t \le 0\\
\mathcal{R}(t) &= R_0(t), \quad -\tau_2 \le t \le 0.
\end{aligned}$$

The problem $\mathcal{A}$ can be rewritten as follows:

$$\min \int_0^T \mathcal{F}(I(t), \mathcal{U}(t))\mathrm{d}t, \tag{2.3}$$

$$X'(t) = \mathbf{f}(t, X(t), \mathcal{U}(t)), \tag{2.4}$$

or

$$\frac{\mathrm{d}X_i}{\mathrm{d}t} = \mathbf{f}_i(t, X(t), \mathcal{U}(t)), \quad i = 1, 2, 3,$$

where $\mathbf{f} : \mathbb{R} \times \mathbb{R}^3 \times \mathbb{R} \to \mathbb{R}^3$, $\mathcal{F} : \mathbb{R}^2 \to \mathbb{R}$, $X(t) = (X_1(t), X_2(t), X_3(t)) = (S(t), R(t), I(t))$ and $X_0 = (S_0, R_0, I_0)$ are initial conditions. The functions $\mathbf{f}_i(t, X(t), \mathcal{U}(t))$ for $i = 1, 2, 3$ and $\mathcal{F}(I(t), \mathcal{U}(t))$ are defined as follows:

$$\begin{aligned}
\mathbf{f}_1(t, X(t), \mathcal{U}(t)) &= (1-p)b - \mu\mathcal{S} - \beta\mathcal{S}(t-\tau_1)\mathcal{I}(t-\tau_1) + \nu\mathcal{R}(t-\tau_2) + \omega\mathcal{U}(t)\mathcal{I},\\
\mathbf{f}_2(t, X(t), \mathcal{U}(t)) &= \beta\mathcal{S}(t-\tau_1)\mathcal{I}(t-\tau_1) - (\mu+\gamma+\alpha)\mathcal{I} - \mathcal{U}(t)\mathcal{I},\\
\mathbf{f}_3(t, X(t), \mathcal{U}(t)) &= pb + \gamma\mathcal{I} - \mu\mathcal{R} - \nu\mathcal{R}(t-\tau_2) + (1-\omega)\mathcal{U}(t)\mathcal{I},
\end{aligned}$$

$$\mathcal{F}(I(t), \mathcal{U}(t)) = \left(\mathcal{I}(t) + \frac{\varepsilon(\mathcal{U}(t))^2}{2}\right).$$

The necessary assumptions to ensure the existence of the solution of problem (2.3) and (2.4) are investigated in interesting work [38].

3 Preliminaries

3.1 Hybrid Block-Pulse Functions and Legendre Polynomials

Hybrid block-pulse functions and Legendre polynomials (HBL), which are shown by $h_{nm}(t)$, are defined for $t \in [0, T)$ as follows:

$$h_{nm}(t) = \begin{cases} P_m\left(\frac{2N}{t_f}t - 2n + 1\right) & t \in \left[\frac{n-1}{N}t_f, \frac{n}{N}t_f\right), \\ 0 & \text{otherwise.} \end{cases}$$

The parameter $n = 1, 2, \ldots, N$ is the order of block-pulse functions and $P_m(s)$ is the Legendre polynomial of order $m = 0, 1, \ldots, M-1$ which is defined on $[-1, 1]$. These polynomials are obtained using the following recursive formula:

$$P_0(s) = 1, \quad P_1(s) = s$$
$$P_{m+1}(s) = \left(\frac{2m+1}{m+1}\right) s P_m(s) - \left(\frac{m}{m+1}\right) P_{m-1}(s), \quad m = 1, 2, 3 \ldots.$$

3.2 Function Approximation

Since both block-pulse functions and Legendre polynomials are complete and orthogonal, the set of hybrid functions is a complete orthogonal system in $L^2[0, t_f)$. Therefore, any function $f(t) \in L^2[0, t_f)$ can be expanded in terms of hybrid functions as:

$$f(t) = \sum_{n=1}^{\infty} \sum_{m=0}^{\infty} c_{nm} h_{nm}(t).$$

Truncating the infinite series, the function can be approximated as:

$$f(t) \simeq f_{NM}(t) = \sum_{n=1}^{N} \sum_{m=0}^{M-1} c_{nm} h_{nm}(t) = \mathbf{C}^T \mathbf{H}(t), \tag{3.1}$$

where

$$\mathbf{C} = \left[c_{10} \cdots c_{1M-1}\ c_{20} \cdots c_{2M-1} \cdots c_{N0} \cdots c_{NM-1}\right]^{\mathrm{T}}, \tag{3.2}$$

and

$$\mathbf{H}(t) = \left[h_{10}(t) \cdots h_{1M-1}(t)\ h_{20}(t) \cdots h_{2M-1}(t) \cdots h_{N0}(t) \cdots h_{NM-1}(t)\right]^{\mathrm{T}}. \tag{3.3}$$

3.3 Operational Matrices

Let $\mathcal{P}$ be the operational matrix for integration of vector $\mathbf{H}(t)$, that is:

$$\int_0^t \mathbf{H}(\tau)\mathrm{d}\tau \approx \mathcal{P}\mathbf{H}(t).$$

Using the properties of Legendre polynomials, $\mathcal{P}$ is a $NM \times NM$ matrix in the following form:

$$\mathcal{P} = \begin{bmatrix} \mathcal{E} & \mathcal{H} & \cdots & \mathcal{H} \\ 0 & \mathcal{E} & \cdots & \mathcal{H} \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \mathcal{E} \end{bmatrix},$$

where $\mathcal{E}$ and $\mathcal{H}$ are $M \times M$ matrices which are defined as follows:

$$\mathcal{H} = \frac{t_f}{2N}\begin{bmatrix} 2 & 0 & \cdots & 0 \\ 0 & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 0 \end{bmatrix},$$

$$\mathcal{E} = \frac{t_f}{2N}\begin{bmatrix} 1 & 1 & 0 & 0 & \cdots & 0 & 0 & 0 \\ \frac{-1}{3} & 0 & \frac{1}{3} & 0 & \cdots & 0 & 0 & 0 \\ 0 & \frac{-1}{5} & 0 & \frac{1}{5} & \cdots & 0 & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & 0 & \cdots & \frac{-1}{2M-3} & 0 & \frac{1}{2M-3} \\ 0 & 0 & 0 & 0 & \cdots & 0 & \frac{-1}{2M-1} & 0 \end{bmatrix}.$$

Also, let:

$$\hat{\mathbf{H}} = \int_0^{t_f} \mathbf{H}(\tau)\mathrm{d}\tau = \left[\mathcal{H}_1\ \mathcal{H}_1 \cdots \mathcal{H}_1\right]^{\mathrm{T}},$$

where $\mathcal{H}_1$ is the first row of $\mathcal{H}$ and is repeated N times:

$$\tilde{\mathbf{H}} = \int_0^{t_f} \mathbf{H}(\tau)\mathbf{H}^T(\tau)\mathrm{d}\tau = \begin{bmatrix} \mathcal{L} & \mathbf{0} & \cdots & \mathbf{0} \\ \mathbf{0} & \mathcal{L} & \cdots & \mathbf{0} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{0} & \mathbf{0} & \cdots & \mathcal{L} \end{bmatrix},$$

where $\mathcal{L}$ is a $M \times M$ matrix in the following form:

$$\mathcal{L} = \frac{t_f}{2N}\begin{bmatrix} 2 & 0 & \cdots & 0 \\ 0 & \frac{2}{3} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \frac{2}{2M-1} \end{bmatrix}.$$

For $i = 1, 2$, the delay operational matrix of hybrid functions corresponding to time-delay τ_i is shown by $\mathcal{D}_i$ and is given by:

$$\mathbf{H}(t - \tau_i) \simeq \mathcal{D}_i \mathbf{H}(t), \quad 0 < \tau_i \leq t. \tag{3.4}$$

To obtain $\mathcal{D}_i$, let ω be the smallest positive integer number for which $\omega\tau_i \in \mathbb{Z}$, and let λ be the greatest common divisor of the integers $\omega\tau_i$. If $\frac{\omega}{\lambda}$ is not an integer, define $\lambda = 1$ and let:

$$N = \frac{\omega}{\lambda} \times t_f.$$

Choosing N in this way leads to the minimum number of block-pulse functions.

The delay operational matrix of hybrid functions is a $NM \times NM$ matrix and is computed as follows:

$$\mathcal{D}_i = \begin{bmatrix} \mathbf{0}_{(N-\varsigma_i)M \times \varsigma_i M} & I_{(N-\varsigma_i)M} \\ \mathbf{0}_{\varsigma_i M \times \varsigma_i M} & \mathbf{0}_{\varsigma_i M \times (N-\varsigma_i)M} \end{bmatrix},$$

where $\varsigma_i = \frac{\omega\tau_i}{\lambda}$.

4 A Discretization Method

In this section, we illustrate a method based on the operational matrices and spectral collocation scheme to convert OCP into a nonlinear programming problem.

Let $s_0 < s_1 < \cdots < s_{K-1}$ be the roots of $P_K(s) + P_{K-1}(s)$ which are called "Legendre–Gauss–Radau nodes". They are defined on $[-1, 1)$ and include a fixed abscissa at the first point of the interval. Mapping these nodes to $\left[\frac{n-1}{N} t_f, \frac{n}{N} t_f\right)$ for $n = 1, 2, \ldots, N$, we have NK collocation nodes on $[0, t_f)$. Let $\mathbf{P}_{M \times K}$ be the matrix consist of the values of Legendre polynomials, i.e., $\mathbf{P}_{ij} = P_{i-1}(s_j)$. Define:

$$\bar{\mathbf{H}} = \begin{bmatrix} \mathbf{P} & \mathbf{0} & \cdots & \mathbf{0} \\ \mathbf{0} & \mathbf{P} & \cdots & \mathbf{0} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{0} & \mathbf{0} & \cdots & \mathbf{P} \end{bmatrix}. \tag{4.1}$$

In fact, $\bar{\mathbf{H}}$ is a $NM \times NK$ matrix which consists of values of hybrid functions at the collocation nodes.

Let us approximate each unknown function in (2.2a), (2.2b) and (2.2c) by applying the same way used in Eq. (3.1):

$$\frac{\mathrm{d}\bar{\mathcal{S}}}{\mathrm{d}t}(t) = \mathbf{S}^{\mathrm{T}}\mathbf{H}(t), \quad \frac{\mathrm{d}\bar{\mathcal{R}}}{\mathrm{d}t}(t) = \mathbf{R}^{\mathrm{T}}\mathbf{H}(t), \quad \frac{\mathrm{d}\bar{\mathcal{I}}}{\mathrm{d}t}(t) = \mathbf{I}^{\mathrm{T}}\mathbf{H}(t), \quad \bar{\mathcal{U}}(t) = \mathbf{U}^{T}\mathbf{H}(t). \tag{4.2}$$

$\mathbf{H}(t)$ is defined in Eq. (3.3), and vectors $\mathbf{S}$, $\mathbf{R}$, $\mathbf{I}$, and $\mathbf{U}$ are defined similar to (3.2). Applying the operational matrix of integration, $\mathcal{P}$, implies that:

$$\bar{\mathcal{S}}(t) = \left(\mathbf{S}^{\mathrm{T}}\mathcal{P} + \mathbf{S}_0^{\mathrm{T}}\right)\mathbf{H}(t), \quad \bar{\mathcal{R}}(t) = \left(\mathbf{R}^{\mathrm{T}}\mathcal{P} + \mathbf{R}_0^{\mathrm{T}}\right)\mathbf{H}(t), \quad \bar{\mathcal{I}}(t) = \left(\mathbf{I}^{\mathrm{T}}\mathcal{P} + \mathbf{I}_0^{\mathrm{T}}\right)\mathbf{H}(t). \tag{4.3}$$

Here, the vectors $\mathbf{S}_0$, $\mathbf{I}_0$, $\mathbf{R}_0$ are of order NM and are defined as follows:

$$\begin{aligned}
\mathbf{S}_0 &= \mathcal{S}(0) \times \left[\mathbf{e}_M\ \mathbf{e}_M \cdots \mathbf{e}_M\right]^{\mathrm{T}},\\
\mathbf{I}_0 &= \mathcal{I}(0) \times \left[\mathbf{e}_M\ \mathbf{e}_M \cdots \mathbf{e}_M\right]^{\mathrm{T}},\\
\mathbf{R}_0 &= \mathcal{R}(0) \times \left[\mathbf{e}_M\ \mathbf{e}_M \cdots \mathbf{e}_M\right]^{\mathrm{T}},
\end{aligned}$$

where $\mathbf{e}_M$ is a vector of order M in which the first element is one and the others are zero. By substituting Eqs. (3.4), (4.2), and (4.3) in OCP introduced by Eqs. (2.1) and evaluating at the collocation nodes, we derive the nonlinear programming problem named as problem $\bar{A}$.
Problem $\bar{A}$: The objective functional is:

$$\begin{aligned}
\mathcal{J}(\mathcal{I},\mathcal{U}) &= \left(\mathbf{I}^{\mathrm{T}}\mathcal{P} + \mathbf{I}_0{}^{\mathrm{T}}\right)\left(\int_0^{t_f}\mathbf{H}(\tau)\mathrm{d}\tau\right) + \frac{\varepsilon}{2}\mathbf{U}^{\mathrm{T}}\left(\int_0^{t_f}\mathbf{H}(\tau)\mathbf{H}^{\mathrm{T}}(\tau)\mathrm{d}\tau\right)\mathbf{U},\\
&= \left(\mathbf{I}^{\mathrm{T}}\mathcal{P} + \mathbf{I}_0{}^{\mathrm{T}}\right)\hat{\mathbf{H}} + \frac{\varepsilon}{2}\mathbf{U}^{\mathrm{T}}\tilde{\mathbf{H}}\mathbf{U}.
\end{aligned}$$

The constraints are:

$$\begin{aligned}
&-\mathbf{S}^{\mathrm{T}}\bar{\mathbf{H}} + (1-\rho)b - \mu\left(\mathbf{S}^{\mathrm{T}}\mathcal{P} + \mathbf{S}_0^{T}\right)\bar{\mathbf{H}} - \beta\left(\mathbf{S}^{\mathrm{T}}\mathcal{P} + \mathbf{S}_0^{T}\right)\mathcal{D}_1\bar{\mathbf{H}}\bar{\mathbf{H}}^{\mathrm{T}}\mathcal{D}_1^{\mathrm{T}}\left(\mathcal{P}^{\mathrm{T}}\mathbf{I} + \mathbf{I}_0\right)\\
&\quad + \nu\left(\mathbf{R}^{\mathrm{T}}\mathcal{P} + \mathbf{R}_0{}^{\mathrm{T}}\right)\mathcal{D}_2\bar{\mathbf{H}} + \omega\mathbf{U}^{\mathrm{T}}\bar{\mathbf{H}}\bar{\mathbf{H}}^{\mathrm{T}}\left(\mathcal{P}^{\mathrm{T}}\mathbf{I} + \mathbf{I}_0\right) = 0.\\
&-\mathbf{I}^{\mathrm{T}}\bar{\mathbf{H}} + \beta\left(\mathbf{S}^{\mathrm{T}}\mathcal{P} + \mathbf{S}_0^{T}\right)\mathcal{D}_1\bar{\mathbf{H}}\bar{\mathbf{H}}^{\mathrm{T}}\mathcal{D}_1^{\mathrm{T}}\left(\mathcal{P}^{\mathrm{T}}\mathbf{I} + \mathbf{I}_0\right) - (\mu+\gamma+\alpha)\left(\mathbf{I}^{\mathrm{T}}\mathcal{P} + I_0{}^{\mathrm{T}}\right)\bar{\mathbf{H}}\\
&\quad -\mathbf{U}^{\mathrm{T}}\bar{\mathbf{H}}\bar{\mathbf{H}}^{\mathrm{T}}\left(\mathcal{P}^{\mathrm{T}}\mathbf{I} + \mathbf{I}_0\right) = 0,\\
&-\mathbf{R}^{\mathrm{T}}\bar{\mathbf{H}} + \rho b + \gamma\left(\mathbf{I}^{\mathrm{T}}\mathcal{P} + I_0{}^{\mathrm{T}}\right)\bar{\mathbf{H}} - \mu\left(\mathbf{R}^{\mathrm{T}}\mathcal{P} + \mathbf{R}_0{}^{\mathrm{T}}\right)\bar{\mathbf{H}} - \nu\left(\mathbf{R}^{\mathrm{T}}\mathcal{P} + \mathbf{R}_0{}^{\mathrm{T}}\right)\mathcal{D}_2\bar{\mathbf{H}}\\
&\quad +(1-\omega)\mathbf{U}^{\mathrm{T}}\bar{\mathbf{H}}\bar{\mathbf{H}}^{\mathrm{T}}\left(\mathcal{P}^{\mathrm{T}}\mathbf{I} + \mathbf{I}_0\right) = 0.
\end{aligned}$$

The obtained nonlinear programming can be solved by any well-developed optimization algorithm. There are various methods and software packages for solving these problems, but in this paper, `fmincon` solver from MATLAB software is used. Although `fmincon` computes the gradient of the objective function and the Jacobian matrix of nonlinear constraints numerically, providing these options for the solver, raises the speed of the method.

The gradient of the objective function is given as follows:

$$\nabla \mathcal{J}(\mathcal{I}, \mathcal{U}) = \begin{bmatrix} \mathbf{0}_{NM} \\ \mathcal{P}\hat{\mathbf{H}} \\ \mathbf{0}_{NM} \\ \varepsilon\tilde{\mathbf{H}}\mathbf{U} \end{bmatrix}.$$

5 Numerical Simulations

In this section, we present applied numerical results to illustrate the efficiency of the LGR spectral collocation approach. All the numerical experiments are performed using Matlab software on a PC with Intel® Core™ i7 processor and 8GB RAM. The LGR spectral collocation method is tested on 243 numerical examples based on the parameters given in Table 3. Numerical results including the values of the objective functions together with the CPU time are reported in Tables 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, and 16 and Figs. 1, 2, 3, 4, and 5. respectively. Based on the aims of this study, the numerical simulation consists of three parts. In the first part, the effect of parameters α and γ on the numerical solutions is considered. In the second part, the relationship between the objective function and the parameters β and b is investigated. The third part is devoted to numerical convergence.

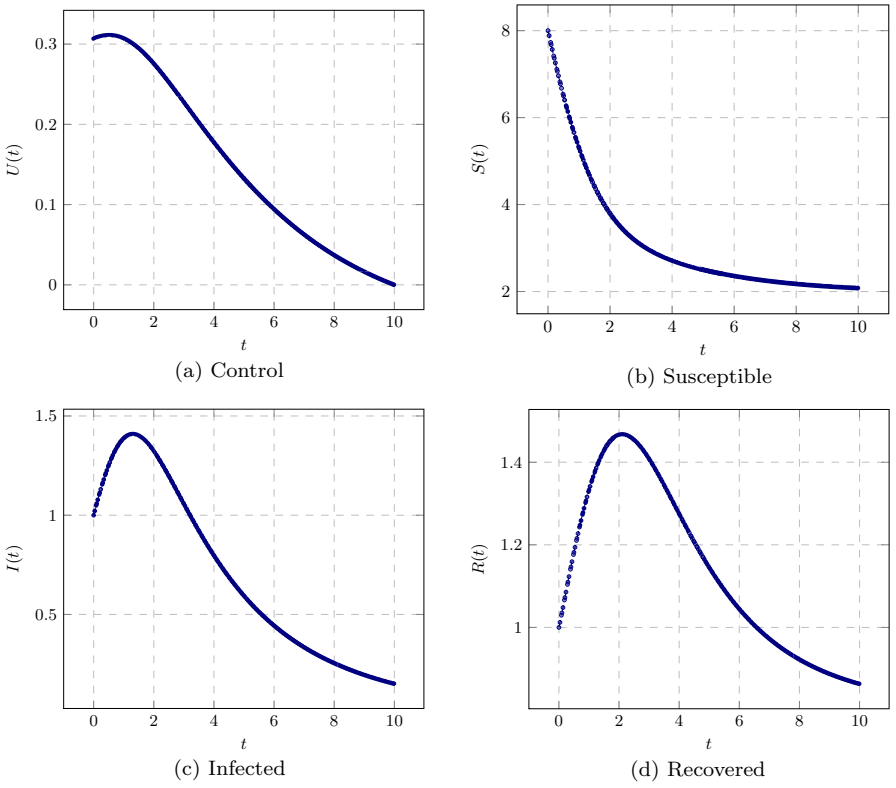
5.1 Random Generation of Numerical Examples

We generated 243 numerical examples in total, which vary in b (the number of new computers), β (the epidemic rate of an infected computer), γ (the retrieval rate of an infected computer), α (the failure rate of an infected computer as a result of the virus activity), and T (the time period). The examples are solved numerically with the proposed approach in Sect. 3. As given in [33], the parameter values are given in Table 3. The initial state variables are considered as $S(0) = 8, I(0) = 1,$ and $R(0) = 1$. In Tables 8, 9, 10, 11, 12, 13, 14, 15, and 16, the columns labeled as CPU time show the solution times in seconds used up by the LGR spectral collocation method. The running times of the proposed method for all examples are reported in Fig. 3. We observe that the LGR spectral collocation method needs increasing execution time as long as the parameters T and N grow. The number of the model variables is equal to $4 \times N \times M$. Therefore, the examples with more variables require longer solution times than the examples with fewer variables. This result can be ascribed to the search region of possible control functions, which expands as the total population increases.

The control variable $u(t)$ and the number of susceptible, infected and recovered computers versus at time t with associated parameters in example with $T = 10$, $N = 100$, $M = 3$, $\beta = 0.2$, $b = 0.9$, $\gamma = 0.35$, and $\alpha = 0.01$ are plotted in Fig. 1.

Table 3 Parameter values

Parameters	Values
T	1, 4, 10
N	10, 40, 100
β	0.2, 0.4, 0.6
b	0.9, 0.95, 1
γ	0.35, 0.4, 0.45
α	0.01, 0.05, 0.1
ρ	0.9
μ	0.32
ν	0.7
ϵ	10
ω	0.35
M	3



(a) Control (b) Susceptible

(c) Infected (d) Recovered

Fig. 1 The obtained results for example $100 - 10 - 1$

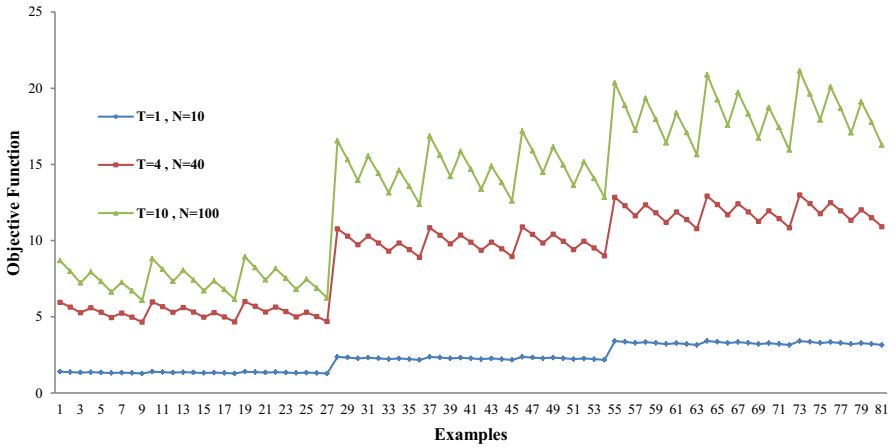


Fig. 2 Objective values of LGR spectral collocation method for 243 examples

5.2 Effect of Parameters α and γ on the Numerical Solutions

We first check whether the parameters α and γ influence network performance or objective function. To this end, there are three distinct levels of both parameters. Tables 8, 9, 10, 11, 12, 13, 14, 15, and 16 and Fig. 2 also demonstrate the objective function values for three different levels of α and γ obtained by implementing LGR spectral collocation method on the 243 numerical examples. This experiment displays that as parameters α and γ increase, the objective function takes on smaller values. The reason for changing the objective value is that parameters α and γ indicate the significant variation between the low and high budget allotted for antivirus software and infected computers. The smallest amount of objective function is obtained until the highest level of these parameters is considered. At this level, the viruses cannot impose the budget for buying antivirus software more than the smallest level; thus, it minimizes the number of infected computers of which are subsequently targeted by viruses. An increase in the parameters α and γ causes more infected computers are recovered, and the damage of computer networks based on the virus is decreased.

5.3 Sensitivity of the Numerical Solutions to Parameters β and b

Our next aim is to perform a sensitivity analysis concerning two parameters in the computer virus model: the epidemic rate of an infected computer (β) and the number of new computers (b). The columns of Tables 8, 9, 10, 11, 12, 13, 14, 15, and 16 corresponding to the objective function show that as β and b increase, the number of infected computers and the total budget for antivirus software during the time period increase. It is a result of the infected computers and the budget increased by the special epidemic rate of infected computers and new computers. As the level of virus activity in a computer network increases with the infection rate of computers, it exhibits a pattern for the diffusion phenomena. An increase in β from 0.2 to 0.6 causes more

Table 4 Numerical results for $T = 1$, $N = 10$, $\beta = 0.6$, $b = 1$, $\gamma = 0.45$, and $\alpha = 0.1$ with different values of M

M	Minimum value	CPU time	Error
2	3.14685019473956	0.31	4.04×10^{-04}
4	3.14725511848899	0.35	4.03×10^{-11}
8	3.14725511844869	3.43	1.15×10^{-12}
16	3.14725511844984	38.21	–

Table 5 Numerical results for $T = 4$, $N = 40$, $\beta = 0.4$, $b = 0.95$, $\gamma = 0.4$, and $\alpha = 0.05$ with different values of M

M	Minimum value	CPU time	Error
2	9.89631773621987	1.48	9.62×10^{-05}
4	9.89641402572560	17.79	6.91×10^{-13}
8	9.89641402572629	102.63	1.29×10^{-13}
16	9.89641402572616	1573.35	–

Table 6 Numerical results for $T = 10$, $N = 100$, $\beta = 0.2$, $b = 0.9$, $\gamma = 0.35$, and $\alpha = 0.05$ with different values of M

M	Minimum value	CPU time	Error
2	8.68170334973227	7	5.37×10^{-06}
4	8.68170872047445	65	4.97×10^{-14}
8	8.68170872047450	717	6.92×10^{-14}
16	8.68170872047457	7878	–

computers to be infected, so the objective function arises due to the fact that the parameter β affects it greatly. As it is shown in the tables, changing b from 0.9 to 1, does not have a considerably effect on the objective function. In brief, according to the values of objective functions, the effect of parameter β on virus spread is extremely high in comparison with the effect of parameter b.

5.4 Convergence

In this paper, numerical approximations of unknown values are given. The approximations depend on the parameters M and N. To show the convergence of the method, we consider one of these parameters fixed and solve the problem for different values of the other parameters. Since it is impossible to show all tables and figures related to parameters in Table 3, the illustrations are limited to some chosen values. At first, assume that N is fixed and $M = 2, 4, 8, 16$. The results are shown in Tables 4, 5, and 6 and Fig. 4. The errors which are given in tables are computed as follows:

$$E_M = |J_{2M} - J_M|, \tag{5.1}$$

where J_M is the value of the objective function obtained by using M basis functions. If the exact values of objective function exist, one can compute the errors as $|J* - J_M|$. However, since the problem has no exact solution, the errors are computed as Eq. (5.1).

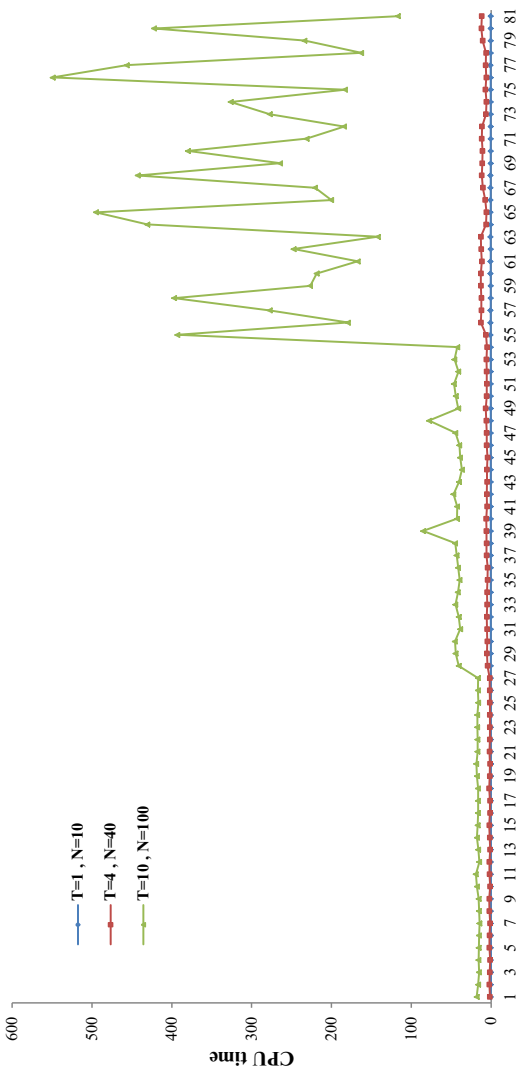
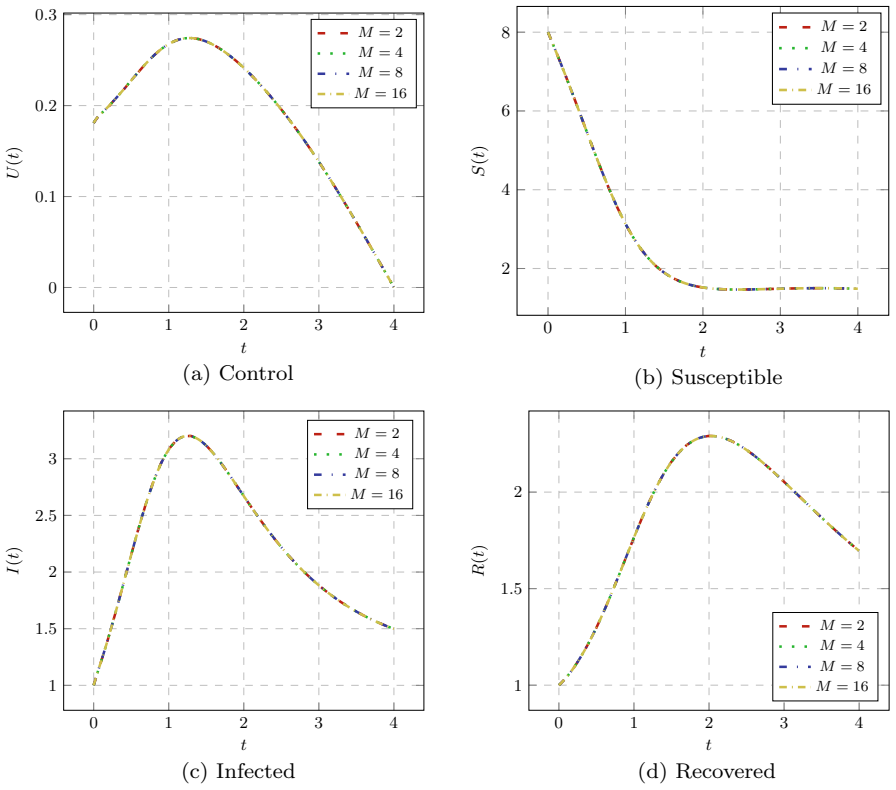


Fig. 3 Solution times of LGR spectral collocation method for 243 examples



(a) Control (b) Susceptible

(c) Infected (d) Recovered

Fig. 4 The effect of parameter M on convergence for $T = 4$, $N = 40$, $\beta = 0.4$, $b = 0.95$, $\gamma = 0.4$, and $\alpha = 0.05$

The illustrations demonstrate that for fixed N, small values of M give good results. As it is seen in these tables, by increasing the number of basis functions from $M = 2$ to $M = 4$, the rate of convergence is so high; but after that, because of the limitation of significant digits in software, the rate decreases.

To consider the effect of parameter N on convergence, the following notation is defined by:

$$E_N = |J_{2N} - J_N|,$$

where J_N is the value of objective function obtained using block-pulse functions of order N. In this case, the rate of convergence is computed as follows:

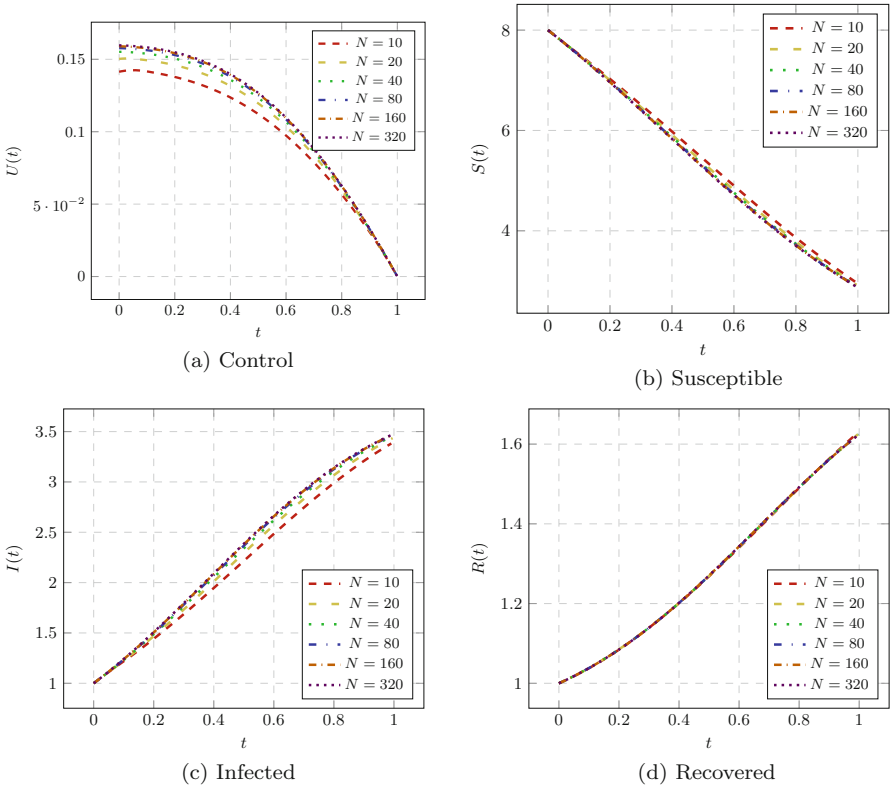
$$r \simeq r_N = \log_2 \left| \frac{J_N - J_{N/2}}{J_{2N} - J_N} \right| = \log_2 \left(\frac{E_{N/2}}{E_N} \right).$$

The results are given in Table 7 and Fig. 5.

As it is shown in these tables, by increasing N, errors decrease, but the rate of convergence is approximately linear. Investigating the proficiency of the proposed method to other problems such as ODEs and PDEs, we guess that the mainreason for

Table 7 Numerical results for $T = 1$, $M = 4$, $\beta = 0.4$, $b = 0.95$, $\gamma = 0.4$, and $\alpha = 0.05$ with different values of N

N	Objective function	CPU time	Error	r_N
10	2.27112172961506	0.93	6.29×10^{-2}	–
20	2.33408436815734	02.62	3.59×10^{-2}	0.809
40	2.37000339182419	22.27	1.92×10^{-2}	0.897
80	2.38928201736109	204.74	1.00×10^{-2}	0.946
160	2.3992865977706	1584.33	5.09×10^{-3}	0.972
320	2.40438548201707	13164.93	–	–



(a) Control (b) Susceptible

(c) Infected (d) Recovered

Fig. 5 The effect of parameter N on convergence for $T = 1$, $M = 4$, $\beta = 0.4$, $b = 0.95$, $\gamma = 0.4$, and $\alpha = 0.05$

the low rate of convergence is the method implemented by software for solving NLP. The illustrations in Fig. 5 demonstrate that by increasing N, the errors decrease and graphs come together.

Table 8 Numerical results for $T = 1$, $N = 10$, $M = 3$, and $\beta = 0.2$ with different combinations of the parameters b, γ, α

b	γ	α	Objective function	CPU time
0.9	0.35	0.01	1.395735194	0.485452
		0.05	1.36977252	0.2481293
		0.1	1.338167142	0.1691817
	0.4	0.01	1.363470737	0.187952
		0.05	1.338258142	0.1916695
		0.1	1.307564168	0.1556139
	0.45	0.01	1.332137373	0.2252013
		0.05	1.307651707	0.2133923
		0.1	1.277840999	0.205234
0.95	0.35	0.01	1.395960118	0.2727331
		0.05	1.369992129	0.2656626
		0.1	1.338380283	0.2885345
	0.4	0.01	1.363689058	0.2440363
		0.05	1.338471305	0.1731071
		0.1	1.307771054	0.1870731
	0.45	0.01	1.332349285	0.2305315
		0.05	1.307858613	0.236871
		0.1	1.278041814	0.1930179
1	0.35	0.01	1.396185088	0.1773287
		0.05	1.370211783	0.1664212
		0.1	1.338593469	0.1883339
	0.4	0.01	1.363907425	0.2438988
		0.05	1.338684511	0.2009496
		0.1	1.307977983	0.194499
	0.45	0.01	1.332561242	0.2277658
		0.05	1.308065563	0.186218
		0.1	1.278242672	0.1822824

6 Conclusion

In this essay, we introduce a Legendre–Gauss–Radau spectral collocation method based on operational matrices for approximating the optimal control problem with nonlinear delay differential equations that models the spread of a computer virus. Since the operational matrices are sparse, the presented method is so attractive and efficient from the numerical point of view. Also, another significant property of the collocation scheme is simple implementation. Applying this method, the problem is reduced to a nonlinear programming system. Many effective algorithms can be applied to solve the NLP. Illustrative examples show the validity and applicability of the proposed method. Besides, we investigate the effects of the failure rate of an infected computer (α), the retrieval rate of an infected computer (γ), the epidemic rate of an infected computer (β), and the number of new computers (b), which are characteristics of a

Table 9 Numerical results for $T = 1$, $N = 10$, $M = 3$, and $\beta = 0.4$ with different combinations of the parameters b, γ, α

b	γ	α	Objective function	CPU time
0.9	0.35	0.01	2.365781137	0.450962
		0.05	2.322649314	0.413151
		0.1	2.270022982	0.3225139
	0.4	0.01	2.312415864	0.3330317
		0.05	2.270417056	0.3904824
		0.1	2.219172251	0.390571
	0.45	0.01	2.260449082	0.30724
		0.05	2.219552814	0.4250282
		0.1	2.169652328	0.3719436
0.95	0.35	0.01	2.366519292	0.4413996
		0.05	2.323372336	0.4111455
		0.1	2.270727501	0.3342619
	0.4	0.01	2.313135303	0.4415285
		0.05	2.271121725	0.3423585
		0.1	2.219858862	0.2858341
	0.45	0.01	2.261150253	0.3031992
		0.05	2.220239572	0.2795358
		0.1	2.170321465	0.3343764
1	0.35	0.01	2.367257697	0.4681866
		0.05	2.324095604	0.4773475
		0.1	2.27143226	0.3711077
	0.4	0.01	2.313854986	0.3701393
		0.05	2.271826635	0.3691204
		0.1	2.220545709	0.3855725
	0.45	0.01	2.261851664	0.3698516
		0.05	2.220926564	0.3205218
		0.1	2.170990832	0.4017553

computer network infected with viruses. We observe that by increasing the parameters α and γ, more infected computers are recovered and the loss of computer network due to virus spread is reduced. Also, as the parameters β and b increase, the number of infected computers and the total budget for buying antivirus software increase. For more analysis, the impact of the time period and the total population on the network performance, using numerical simulation, is examined. We hope that the described method can accommodate other applied problems that are modeled in the form of OCP with integral or differential equations.

Table 10 Numerical results for $T = 1$, $N = 10$, $M = 3$, and $\beta = 0.6$ with different combinations of the parameters b, γ, α

b	γ	α	Objective function	CPU time
0.9	0.35	0.01	3.406024693	0.5331877
		0.05	3.348594518	0.5372786
		0.1	3.278340042	0.4885005
	0.4	0.01	3.335422055	0.4418742
		0.05	3.279328488	0.4316547
		0.1	3.210708118	0.5019493
	0.45	0.01	3.26645506	0.5030503
		0.05	3.211666296	0.4648978
		0.1	3.144641164	0.5277863
0.95	0.35	0.01	3.40744257	0.4827742
		0.05	3.349988135	0.5765422
		0.1	3.279703844	0.6306487
	0.4	0.01	3.336810141	0.5899578
		0.05	3.280692758	0.490694
		0.1	3.212043122	0.376473
	0.45	0.01	3.267813898	0.385087
		0.05	3.213001757	0.3970977
		0.1	3.145947902	0.3917367
1	0.35	0.01	3.40886101	0.4809862
		0.05	3.351382309	0.4477662
		0.1	3.281068195	0.4439165
	0.4	0.01	3.338198783	0.412185
		0.05	3.282057577	0.5215501
		0.1	3.213378669	0.4063096
	0.45	0.01	3.269173284	0.4228655
		0.05	3.21433776	0.4307014
		0.1	3.147255175	0.3485762

Table 11 Numerical results for $T = 4$, $N = 40$, $M = 3$, and $\beta = 0.2$ with different combinations of the parameters b, γ, α

b	γ	α	Objective function	CPU time
0.9	0.35	0.01	5.950145668	1.4769039
		0.05	5.63495912	2.0599499
		0.1	5.266767034	1.6177954
	0.4	0.01	5.586121395	1.1657523
		0.05	5.291514054	2.0958251
		0.1	4.947558373	1.9575374
	0.45	0.01	5.245006748	1.2115584
		0.05	4.969859167	2.0910418
		0.1	4.648796541	2.2031266
0.95	0.35	0.01	5.977457972	1.2079969
		0.05	5.66060807	1.9380329
		0.1	5.290446785	2.1198604
	0.4	0.01	5.611575612	1.2130427
		0.05	5.315388129	1.5830145
		0.1	4.9695656	2.3121974
	0.45	0.01	5.268689888	1.266826
		0.05	4.992044662	1.3278953
		0.1	4.669216335	2.3659261
1	0.35	0.01	6.004953386	1.6790741
		0.05	5.686432829	1.3935374
		0.1	5.314293132	1.9311169
	0.4	0.01	5.637205019	1.4131171
		0.05	5.339429988	1.4489999
		0.1	4.991731322	1.5402754
	0.45	0.01	5.292540133	1.4850667
		0.05	5.014389796	1.6274172
		0.1	4.689786457	1.6792666

Table 12 Numerical results for $T = 4$, $N = 40$, $M = 3$, and $\beta = 0.4$ with different combinations of the parameters b, γ, α

b	γ	α	Objective function	CPU time
0.9	0.35	0.01	10.7741106	4.3566213
		0.05	10.29295027	5.1888407
		0.1	9.728345061	4.6567128
	0.4	0.01	10.29803474	5.0191449
		0.05	9.841303138	5.2761498
		0.1	9.305167417	4.8697628
	0.45	0.01	9.844078712	5.003754
		0.05	9.410460948	4.4178679
		0.1	8.901263524	4.2562525
0.95	0.35	0.01	10.83397011	5.6130126
		0.05	10.35021974	5.6862122
		0.1	9.782512311	5.76078
	0.4	0.01	10.35565443	6.131616
		0.05	9.896414002	4.9908204
		0.1	9.35727523	5.3194273
	0.45	0.01	9.899518936	5.0443897
		0.05	9.463471832	5.1265147
		0.1	8.951368188	4.4394302
1	0.35	0.01	10.89415593	5.4678652
		0.05	10.40781561	5.4530504
		0.1	9.837004988	5.942806
	0.4	0.01	10.41360165	6.6483864
		0.05	9.951851881	5.4748203
		0.1	9.409708365	5.3607396
	0.45	0.01	9.955287437	5.4226117
		0.05	9.516809882	5.7382213
		0.1	9.001797618	4.9741912

Table 13 Numerical results for $T = 4$, $N = 40$, $M = 3$, and $\beta = 0.6$ with different combinations of the parameters b, γ, α

b	γ	α	Objective function	CPU time
0.9	0.35	0.01	12.84314417	6.3882445
		0.05	12.28401104	12.7481364
		0.1	11.62991369	12.1799992
	0.4	0.01	12.35085941	12.1050084
		0.05	11.81899243	12.6840333
		0.1	11.19657402	12.7140877
	0.45	0.01	11.8811333	11.5488626
		0.05	11.37506273	12.4017847
		0.1	10.78261486	12.6847271
0.95	0.35	0.01	12.91628806	6.2041805
		0.05	12.35421837	5.8450606
		0.1	11.69656557	7.2670535
	0.4	0.01	12.42175734	10.3077107
		0.05	11.88703135	11.6015513
		0.1	11.26115333	10.992446
	0.45	0.01	11.94984624	10.7949266
		0.05	11.44099237	11.7254408
		0.1	10.84517847	11.3264953
1	0.35	0.01	12.9897542	6.3404984
		0.05	12.42475812	5.8720406
		0.1	11.76356113	7.0015953
	0.4	0.01	12.49298352	6.0124299
		0.05	11.95540806	6.7081039
		0.1	11.32608088	6.3182069
	0.45	0.01	12.0188931	10.6222404
		0.05	11.50726485	12.1334045
		0.1	10.90809447	11.5737609

Table 14 Numerical results for $T = 10$, $N = 100$, $M = 3$, and $\beta = 0.2$ with different combinations of the parameters b, γ, α

b	γ	α	Objective function	CPU time
0.9	0.35	0.01	8.681708718	17.9998627
		0.05	7.990614007	16.1619061
		0.1	7.221229915	15.0246097
	0.4	0.01	7.935278149	15.6494717
		0.05	7.312930596	15.3769609
		0.1	6.620390644	15.0824155
	0.45	0.01	7.257758355	14.3654783
		0.05	6.698210592	15.0259145
		0.1	6.075663087	15.4379487
0.95	0.35	0.01	8.808828423	17.5966267
		0.05	8.103687564	18.98457
		0.1	7.318641081	14.7508161
	0.4	0.01	8.048233951	15.9054156
		0.05	7.413102753	17.618794
		0.1	6.706396747	16.7089505
	0.45	0.01	7.357675546	16.2042266
		0.05	6.786576683	16.2289766
		0.1	6.151307228	16.2587798
1	0.35	0.01	8.939548143	17.6747747
		0.05	8.220054561	18.3929735
		0.1	7.418969603	16.8776653
	0.4	0.01	8.164501584	16.8755787
		0.05	7.516279578	17.2615655
		0.1	6.795037412	17.1422959
	0.45	0.01	7.460608651	15.8901257
		0.05	6.877656425	16.4892854
		0.1	6.229307184	16.2750749

Table 15 Numerical results for $T = 10$, $N = 100$, $M = 3$, and $\beta = 0.4$ with different combinations of the parameters b, γ, α

b	γ	α	Objective function	CPU time
0.9	0.35	0.01	16.54772775	40.4769009
		0.05	15.31683818	44.2272584
		0.1	13.94724402	45.1629167
	0.4	0.01	15.5508242	38.5857032
		0.05	14.40886879	40.3731618
		0.1	13.1379532	44.6012384
	0.45	0.01	14.61513446	41.4451272
		0.05	13.55627012	39.3691047
		0.1	12.37753783	41.2559528
0.95	0.35	0.01	16.86015283	43.1341199
		0.05	15.60351687	44.8412568
		0.1	14.20388834	85.0735595
	0.4	0.01	15.84591989	42.6210278
		0.05	14.67911782	42.4598117
		0.1	13.37929921	47.0709733
	0.45	0.01	14.8932733	40.1813753
		0.05	13.81047477	36.364925
		0.1	12.60399225	38.7667306
1	0.35	0.01	17.17920345	39.6327739
		0.05	15.89694534	44.6570096
		0.1	14.4673385	77.7408264
	0.4	0.01	16.14778792	41.083773
		0.05	14.95621294	44.1599035
		0.1	13.62747994	46.4694216
	0.45	0.01	15.17829866	40.9429681
		0.05	14.07158539	45.8352172
		0.1	12.83727131	42.3445846

Table 16 Numerical results for $T = 10$, $N = 100$, $M = 3$, and $\beta = 0.6$ with different combinations of the parameters b, γ, α

b	γ	α	Objective function	CPU time
0.9	0.35	0.01	20.35020343	392.8598888
		0.05	18.87852235	179.1964377
		0.1	17.23711676	277.2930754
	0.4	0.01	19.33484389	396.9024361
		0.05	17.95521106	226.588858
		0.1	16.41479006	218.3905869
	0.45	0.01	18.37509415	166.9293607
		0.05	17.08133786	247.3703835
		0.1	15.63527584	141.8004736
0.95	0.35	0.01	20.88022405	430.5270612
		0.05	19.24663126	494.8869953
		0.1	17.57564265	200.1285034
	0.4	0.01	19.71323186	220.3568089
		0.05	18.30920568	442.6235708
		0.1	16.74009306	264.1120116
	0.45	0.01	18.73896901	379.7008159
		0.05	17.42156417	231.0733014
		0.1	15.94765006	184.0809563
1	0.35	0.01	21.14142752	277.0214269
		0.05	19.61977129	326.2175109
		0.1	17.91968945	182.93202
	0.4	0.01	20.09648498	549.1898699
		0.05	18.66845358	456.3452339
		0.1	17.07113048	162.5428471
	0.45	0.01	19.10793376	233.63209
		0.05	17.76726707	422.1640951
		0.1	16.26597037	116.7512762

Acknowledgements The first and third authors would like to thank Gonbad Kavous University for supporting this research work. The second author would like to appreciate the research council of Farhangian University for supporting this research.

Compliance with Ethical Standard

Conflict of interest The authors declare that they have no conflict of interest.

References

1. Kephart, J.O., White, S.R.: Directed-graph epidemiological models of computer viruses. In: B. A. Huberman (ed.) Computation: the micro and the macro view, pp. 71–102. World Scientific (1992)
2. Kephart, J., White, S.: IEEE Computer Society Symposium on Research in Security and Privacy, pp. 343–359. IEEE, Los Alamitos (1991)
3. Ren, J., Yang, X., Zhu, Q., Yang, L.X., Zhang, C.: A novel computer virus model and its dynamics. Nonlinear Anal. Real World Appl. **13**(1), 376–384 (2012)
4. Gan, C., Yang, X., Liu, W., Zhu, Q.: A propagation model of computer virus with nonlinear vaccination probability. Commun. Nonlinear Sci. Numer. Simul. **19**(1), 92–100 (2014)
5. Yuan, H., Chen, G., Wu, J., Xiong, H.: Towards controlling virus propagation in information systems with point-to-group information sharing. Decis. Support Syst. **48**(1), 57–68 (2009)
6. Mishra, B.K., Jha, N.: Seiqrs model for the transmission of malicious objects in computer network. Appl. Math. Model. **34**(3), 710–715 (2010)
7. Yang, L.X., Yang, X.: The spread of computer viruses under the influence of removable storage devices. Appl. Math. Comput. **219**(8), 3914–3922 (2012)
8. Zhu, Q., Yang, X., Yang, L.X., Zhang, X.: A mixing propagation model of computer viruses and countermeasures. Nonlinear Dyn. **73**(3), 1433–1441 (2013)
9. Mishra, B.K., Saini, D.K.: Seirs epidemic model with delay for transmission of malicious objects in computer network. Appl. Math. Comput. **188**(2), 1476–1482 (2007)
10. Yang, X., Yang, L.X.: Towards the epidemiological modeling of computer viruses. Discrete Dyn. Nat. Soc. (2012). https://doi.org/10.1155/2012/259671
11. Billings, L., Spears, W.M., Schwartz, I.B.: A unified prediction of computer virus spread in connected networks. Phys. Lett. A **297**(3–4), 261–266 (2002)
12. Zhu, Q., Yang, X., Ren, J.: Modeling and analysis of the spread of computer virus. Commun. Nonlinear Sci. Numer. Simul. **17**(12), 5117–5124 (2012)
13. Gan, C., Yang, X., Liu, W., Zhu, Q., Zhang, X.: An epidemic model of computer viruses with vaccination and generalized nonlinear incidence rate. Appl. Math. Comput. **222**, 265–274 (2013)
14. Gan, C., Yang, X., Zhu, Q., Jin, J., He, L.: The spread of computer virus under the effect of external computers. Nonlinear Dyn. **73**(3), 1615–1620 (2013)
15. Yuan, H., Chen, G.: Network virus-epidemic model with the point-to-group information propagation. Appl. Math. Comput. **206**(1), 357–367 (2008)
16. Mishra, B.K., Pandey, S.K.: Dynamic model of worms with vertical transmission in computer network. Appl. Math. Comput. **217**(21), 8438–8446 (2011)
17. Yang, L.X., Yang, X.: A new epidemic model of computer viruses. Commun. Nonlinear Sci. Numer. Simul. **19**(6), 1935–1944 (2014)
18. Yang, L.X., Yang, X.: The effect of infected external computers on the spread of viruses: a compartment modeling study. Phys. A Stat. Mech. Appl. **392**(24), 6523–6535 (2013)
19. Ren, J., Yang, X., Yang, L.X., Xu, Y., Yang, F.: A delayed computer virus propagation model and its dynamics. Chaos Solitons Fractals **45**(1), 74–79 (2012)
20. Zhang, C., Zhao, Y., Wu, Y.: An impulse model for computer viruses. Discrete dynamics in nature and society. Discrete Dyn. Nat. Soc. 1–13 (2012)
21. Barthélemy, M., Barrat, A., Pastor-Satorras, R., Vespignani, A.: Velocity and hierarchical spread of epidemic outbreaks in scale-free networks. Phys. Rev. Lett. **92**(17), 178701 (2004)
22. Shi, H., Duan, Z., Chen, G.: An sis model with infective medium on complex networks. Phys. A Stat. Mech. Appl. **387**(8–9), 2133–2144 (2008)

23. d'Onofrio, A.: A note on the global behaviour of the network-based sis epidemic model. Nonlinear Anal. Real World Appl. **9**(4), 1567–1572 (2008)
24. Draief, M., Ganesh, A., Massoulié, L.: Thresholds for virus spread on networks. Ann. Appl. Probab. **18**(2), 359–378 (2008)
25. Griffin, C., Brooks, R.: A note on the spread of worms in scale-free networks. IEEE Trans. Syst. Man Cybern. Part B (Cybernetics) **36**(1), 198–202 (2006)
26. Yang, L.X., Yang, X., Liu, J., Zhu, Q., Gan, C.: Epidemics of computer viruses: a complex-network approach. Appl. Math. Comput. **219**(16), 8705–8717 (2013)
27. Kazeem, O., Samson, O., Ebenezer, B.: Optimal control analysis of computer virus transmission. Int. J. Netw. Secur. **20**(5), 971–982 (2018)
28. Sharma, S., Mondal, A., Pal, A., Samanta, G.: Stability analysis and optimal control of avian influenza virus a with time delays. Int. J. Dyn. Control **6**(3), 1351–1366 (2018)
29. Chen, L., Hattaf, K., Sun, J.: Optimal control of a delayed slbs computer virus model. Phys. A Stat. Mech. Appl. **427**, 244–250 (2015)
30. Darajat, P.P., Suryanto, A., Widodo, A.: Optimal control on the spread of slbs computer virus model. Int. J. Pure Appl. Math. **107**(3), 749–758 (2016)
31. Ren, J., Xu, Y., Zhang, C.: Optimal control of a delay-varying computer virus propagation model. Discrete Dyn. Nat. Soc. (2013). https://doi.org/10.1155/2013/210219
32. Gan, C., Yang, M., Zhang, Z., Liu, W.: Global dynamics and optimal control of a viral infection model with generic nonlinear infection rate. Discrete Dyn. Nat. Soc. (2017). https://doi.org/10.1155/2017/7571017
33. Zhu, Q., Yang, X., Yang, L.X., Zhang, C.: Optimal control of computer virus under a delayed model. Appl. Math. Comput. **218**(23), 11613–11619 (2012)
34. Bi, J., Yang, X., Wu, Y., Xiong, Q., Wen, J., Tang, Y.Y.: On the optimal dynamic control strategy of disruptive computer virus. Discrete Dyn. Nat. Soc. (2017). https://doi.org/10.1155/2017/8390784
35. De Groote, F., Kinney, A.L., Rao, A.V., Fregly, B.J.: Evaluation of direct collocation optimal control problem formulations for solving the muscle redundancy problem. Ann. Biomed. Eng. **44**(10), 2922–2936 (2016)
36. Jiang, G., Yang, Q.: Bifurcation analysis in an sir epidemic model with birth pulse and pulse vaccination. Appl. Math. Comput. **215**(3), 1035–1046 (2009)
37. Shi, R., Jiang, X., Chen, L.: The effect of impulsive vaccination on an sir epidemic model. Appl. Math. Comput. **212**(2), 305–311 (2009)
38. Cesari, L.: An existence theorem in problems of optimal control. J. Soc. Ind. Appl. Math. Ser. Control **3**(1), 7–22 (1965)