

IMPROVED PRECISE DIRECT INTEGRATION METHOD FOR TIME-VARYING RICCATI DIFFERENTIAL EQUATION

Xiaomei Liu^{1,†}, Xiaolin Zhan¹ and Xuanhao Zhang¹

Abstract It is always limitations of existing approximate and numerical approaches to obtaining a high precision numerical solution of time-varying Riccati differential equation in a large region. A novel precise direct integration method (VHPD-TR) is proposed to compute the transfer matrix of the transformed differential equations from Riccati differential equation, instead of solving nonlinear algebraic equations. By introducing changes of variables and expanding dimensions, the Riccati equation is equivalently converted to a system of first-order time-varying differential equations, then approximately transformed into a time-invariant differential system, the solution of which is proved to converge to the exact solution of the Riccati differential equation as the expanding dimensions approach infinity. Moreover, the transformed time-invariant differential system can be precisely computed with the precise direct integration method even when the transfer matrix is stiff. The results of five examples verify the validity and simplicity of the proposed technique, and demonstrate the superior performance of VHPD-TR over previous methods.

Keywords Riccati differential equation, precise direct integration method, time-varying system, stiff problem.

MSC(2010) 65L05, 65L04, 34A34.

1. Introduction

Riccati differential equations play a pivotal role in applied science [6, 11, 35]. In optimal control, they enable finite-horizon state-feedback design via the Linear Quadratic Regulator framework. In diffusion processes, Riccati-type ordinary differential equations serve as auxiliary tools for constructing traveling wave solutions. In financial mathematics, they govern the coefficient dynamics in the Cox-Ingersoll-Ross interest rate model. Consequently, efficient solution methods for these equations are of great practical importance. But it is very difficult to derive the analytical solution in an explicit form except for certain special equations. Generally, researchers design approximate approaches or numerical techniques to get its solution.

For approximate solutions, decomposition methods and iterative methods have been widely employed to solve various problems, including ordinary differential equations [14], partial differential equations [42], and integral equations [43]. In what follows, we review the progress of these methods in solving Riccati differential equations. El-Tawil et al. applied the multi-stage Adomian's decomposition method (ADM) to give the series solution of Riccati differential equations [41]. S. Abbasbandy et al. proposed the Homotopy perturbation method (HPM) [1, 2],

[†]The corresponding author.

¹School of Mathematics, Physics and Statistics, Shanghai Polytechnic University, Shanghai 201209, China

Email: xmlu@sspu.edu.cn(X. Liu), xlzhan@sspu.edu.cn(X. Zhan),
xhzhang@sspu.edu.cn(X. Zhang)

and homotopy analysis method (HAM) [39] to solve quadratic Riccati differential equation. Then a series of modified HPM and HAM were developed [5, 21, 28, 31] and applied in fractional-order Riccati equations [3, 7, 22]. Combining the variational iteration method with Adomian's polynomials, He's variational iteration method (VIM) was suggested for the numerical solution [9] and other modified variational iteration methods were designed [17, 18]. Sakar presented the iterative reproducing kernel Hilbert spaces method (IRKHSM) to represent the exact solution of Riccati differential equation through reproducing kernel space [36]. Tang et al. transformed the Riccati equation into a second-order ordinary differential equation and obtained the approximate solution of the Volterra integral equation using Taylor expansion [40]. Limitations and demerits of the above methods had been commented on, such as calculating Adomian's polynomials, inherent inaccuracy of Lagrange multiplier, choosing auxiliary parameter h , and truncation of the function [29]. Moreover, these methods are available for getting a good solution in the neighborhood of the initial position, but ineffective for a larger region. To address this limitation, Chakir recently proposed a global semi-analytical approach based on two-point Padé approximations to construct rational approximations for Riccati differential equations with constant coefficients [12]. However, this approach is not applicable to time-varying Riccati differential equations.

For numerical techniques, one common approach employs operational matrix methods based on orthogonal and non-orthogonal functions or polynomials to solve Riccati differential equations, the main aim of which is to reduce the problem to a set of nonlinear algebraic equations based on operational matrix approximation for the numerical solution. Cubic B-spline scaling functions and Chebyshev cardinal functions were initially introduced to express the solution for solving Riccati equations [4, 23]. Legendre wavelet method, Haar wavelet operational matrix method (HWOMM) and Jacobi wavelet operational matrix method (JWOM) have been developed for quadratic Riccati differential equations and related differential equations [8, 13, 24, 25]. Bezier curves were also applied to find the approximate solution using optimization methods [19]. Moreover, Parand et al. proposed the exact operational matrices method (EOM) based on Bernstein polynomials instead of repeated approximations before reaching the residual function of the system, avoiding a huge loss of dimension in the approximation [32]. Rabiei et al. defined fractional-order Boubaker wavelets and applied them to solve fractional Riccati differential equations [33]. Alternatively, Riccati differential equation is reduced to a linear or nonlinear algebraic system by approximating the derivatives with Taylor expansions, thereby achieving direct computation of the solution. Rehman designed an unconditionally stable nonstandard finite difference predictor-corrector (NSFD-PC) strategy for nonlinear quadratic Riccati differential equations, the results of which show that it achieves higher accuracy and maintains excellent consistency with the exact solutions [34]. Yirga compared several fourth-order Runge-Kutta families for solving a quadratic Riccati equation and found the England scheme to be the most efficient [44]. However, it is difficult to obtain high precision solutions due to solving nonlinear algebraic equations.

The motive of the present work is to establish a direct and high precision numerical method in a larger region for the time-varying differential Riccati equation. For homogeneous linear time-invariant differential equations, Zhong and Williams [45, 46] originally put forward a high precision numerical method – precise direct integration method (HPD), the numerical results of which approach machine precision at the integration points. The simplicity and reliability of the method has proved it to be a power tool for nonhomogeneous time-invariant linear system [20, 26, 47] and homogeneous time-varying linear system [15, 27]. Moreover, researchers have used the HPD method to solve some special Riccati equations. Gao et al. proposed an im-

proved precise integration method for solving matrix Riccati differential equation with constant coefficients [16]. Symplectic conservative perturbation method was presented to solve Riccati differential equation, where the system was decomposed into a zero-order system computed by HPD and a first-order time-varying perturbation system calculated by approximation method based on Taylor series expansion in each time interval, then the solution was determined through successive inverse transformation [38]. However, it is time consuming to recalculate the transfer matrix in every small interval. So the main purpose of our technique is to approximately reduce the nonlinear Riccati differential equation with time-varying coefficients to a first-order linear time-invariant system, so that it can be computed by a new improved HPD method and the transfer matrix can be generated for once and used forever.

This paper is organized as follows: In Section 2, we equivalently transform the Riccati differential equation to a system of first-order time-varying differential equations, by introducing a series of transformations and expanding dimensions. In Section 3, precise direct integration method is introduced. In Section 4, the formulation of the proposed method (VHPD-TR) is described. In this approach, the Riccati differential equation is approximately transformed into a time-invariant differential system by introducing new variables, expanding dimensions and the scaling time domain, which is solved by HPD. In Section 5, the convergence and error analysis of the proposed method are discussed. In section 6, we report our computational results and demonstrate the accuracy and validity of the proposed numerical scheme by presenting numerical examples. In section 7, we end this paper with a brief conclusion.

2. Transformations of the time-varying Riccati differential equation

The famous Weierstrass approximation theorem states that any continuous function can be approximated by a polynomial function. So considering the time-varying Riccati differential equation with polynomial coefficients, it is as follows

$$y'(t) = p(t) + q(t)y(t) + r(t)y^2(t), \quad t_0 \leq t \leq t_f, \quad (2.1)$$

with initial value $y(t_0) = y_0$, in which $r(t) \neq 0$, $p(t)$, $q(t)$ and $r(t)$ are polynomials.

Firstly, introducing the well-known change of variable

$$u(t) = \exp\left(-\int_{t_0}^t r(s)y(s)ds\right),$$

that is

$$y(t) = -\frac{u'(t)}{r(t)u(t)},$$

Eq. (2.1) is transformed into the following second-order time-varying differential equation

$$u''(t) - \left[\frac{r'(t)}{r(t)} + q(t)\right]u'(t) + [p(t)r(t)]u(t) = 0. \quad (2.2)$$

Next, assuming

$$v(t) = -\frac{u'(t)}{r(t)}, \quad (2.3)$$

then the derivative of $v(t)$ is simplified to the following relation by Eq. (2.2)

$$v'(t) = q(t)v(t) + p(t)u(t). \quad (2.4)$$

Lastly, let $\tilde{\mathbf{x}}(t) = [u(t), v(t)]^T$, then Eq. (2.3) and Eq. (2.4) are integrated into the following homogeneous first-order differential system with polynomial coefficients

$$\tilde{\mathbf{x}}'(t) = \begin{bmatrix} 0 & -r(t) \\ p(t) & q(t) \end{bmatrix} \tilde{\mathbf{x}}(t). \quad (2.5)$$

Accordingly, the initial value becomes to

$$\tilde{\mathbf{x}}(t_0) = [1, y(t_0)]^T.$$

So the final solution of Eq. (2.1) is obtained by $y(t) = \frac{v(t)}{u(t)}$.

In addition, if $p(t) = 0$ or there exists a special solution, Eq. (2.1) is simplified to the following Bernoulli differential equation

$$z'(t) = \tilde{q}(t)z(t) + r(t)z^2(t). \quad (2.6)$$

Here, $\tilde{q}(t)$ is a polynomial function of $q(t)$ and $r(t)$. Adopting the transformation

$$\tilde{x}(t) = [z(t)]^{-1}.$$

Eq. (2.6) is expressed as a first-order nonhomogeneous differential equation with polynomial coefficients

$$\tilde{x}'(t) = -\tilde{q}(t)\tilde{x}(t) - r(t). \quad (2.7)$$

Both Eq. (2.5) and Eq. (2.7) indicate that quadratic Riccati differential equation with polynomial coefficients can be converted to a first-order time-varying differential system with polynomial coefficients.

3. Precise direct integration method for homogeneous time-invariant differential equations

As for homogeneous time-invariant differential equations

$$\mathbf{Y}'(t) = \mathbf{C}\mathbf{Y}(t), t \geq t_0,$$

the exact solution is expressed as

$$\mathbf{Y}(t) = e^{\mathbf{C}(t-t_0)}\mathbf{Y}(t_0).$$

Note that τ is taken as the time step, i.e. $\tau = t_{k+1} - t_k (k = 0, 1, 2, \dots)$, and $\mathbf{Y}_k = \mathbf{Y}(t_k)$. The solutions at the integration points are

$$\mathbf{Y}_{k+1} = e^{\tau\mathbf{C}} \cdot \mathbf{Y}_k \triangleq \mathbf{T}(\tau) \cdot \mathbf{Y}_k.$$

So the key to obtain high precision solution is to compute the transfer matrix $\mathbf{T}(\tau)$ accurately and efficiency. But the round-off error difficulties and the computing costs of the Taylor approximants

increase as $\tau \| \mathbf{C} \|$ increases, or as the spread of the eigenvalues of $\mathbf{C}$ increases [30]. So the scaling and squaring method as follows was proposed for solving the above difficulties,

$$\mathbf{T}(\tau) = e^{\tau \mathbf{C}} = (e^{\tau \mathbf{C}/2^N})^{2^N}.$$

Generally, the matrix $e^{\tau \mathbf{C}/2^N}$ is approximated by h -order Taylor expansion $\mathbf{T}_h$,

$$\mathbf{T}_h = \mathbf{I} + \mathbf{T}_{h,0} = \mathbf{I} + \tau \mathbf{C} + \frac{\tau^2 \mathbf{C}^2}{2!} + \cdots + \frac{\tau^h \mathbf{C}^h}{h!}.$$

Here, τ should be selected to be very small in order to improve precision in theory, but the numerical accuracy is completely lost when $\mathbf{T}_{h,0}$ is added into the unit matrix $\mathbf{I}$, because $\mathbf{T}_{h,0}$ is the same order as the round-off error in the floating point operations of computers. Moreover, the method is not efficient for calculations especially when the matrix $(\tau \mathbf{C})$ is badly stiff.

In practice, the process of computing the transfer matrix $\mathbf{T}(\tau)$ approximately by repeated squaring

$$\mathbf{T}(\tau) \approx (\mathbf{T}_h)^{2^N} = (((\mathbf{T}_h^2)^2) \cdots)^2,$$

is desired to write the following form

$$\begin{aligned} (\mathbf{I} + \mathbf{T}_{h,0})^2 &= \mathbf{I} + 2\mathbf{T}_{h,0} + \mathbf{T}_{h,0} \times \mathbf{T}_{h,0} = \mathbf{I} + \mathbf{T}_{h,1}, \\ (\mathbf{I} + \mathbf{T}_{h,1})^2 &= \mathbf{I} + 2\mathbf{T}_{h,1} + \mathbf{T}_{h,1} \times \mathbf{T}_{h,1} = \mathbf{I} + \mathbf{T}_{h,2}, \\ &\dots \\ (\mathbf{I} + \mathbf{T}_{h,N-1})^2 &= \mathbf{I} + 2\mathbf{T}_{h,N-1} + \mathbf{T}_{h,N-1} \times \mathbf{T}_{h,N-1} = \mathbf{I} + \mathbf{T}_{h,N}. \end{aligned}$$

Here, all small amounts are first added up to the incremental part $\mathbf{T}_{h,N}$ by performing the following operation N times,

$$\mathbf{T}_h \leftarrow \mathbf{T}_h + \mathbf{T}_h^2,$$

and then added to the identity matrix $\mathbf{I}$. Therefore, the transfer matrix $\mathbf{T}(\tau)$ is split into the two parts, $\mathbf{I}$ and $\mathbf{T}(\tau)$. Only in this way can the transfer matrix $\mathbf{T}(\tau)$ avoid round-off errors significantly. The above technique is the key to the precise direct integration method (HPD). The process of the HPD method is summarized in Algorithm 1.

Algorithm 1 HPD method for homogeneous invariant differential equation

Input: Initial value $Y_1 = Y(t_0)$, coefficient matrix C , time-step size τ , steps m and precision parameter N ;

1: Compute $d\tau = 2^{-N}\tau$, $T = d\tau * C$;

2: **for** $i=1$ to N **do**

3: $T = 2 * T + T^2$;

4: **end for**

5: $T = I + T$.

6: **for** $j=1$ to m **do**

7: $Y_{j+1} = T * Y_j$;

8: **end for**

Output: Y .

From the above approach, we deduce that provided N is appropriately chosen, the truncation error of $\mathbf{T}(\tau)$ is negligible and a high-precision solution is obtained, regardless of the time-step size. Furthermore, once the transfer matrix $\mathbf{T}(\tau)$ is generated at the first step, the result of $\mathbf{T}(\tau)$ can be used forever in the whole computation process. Thus, the HPD method is suitable for long time and large time-step size computations, making it an efficient algorithm.

Next, we describe an improved HPD method for Riccati differential equations in details.

4. Precise direct integration method (VHPD-TR) for Riccati differential equations

From the above, the Riccati differential equation can be expressed as the following first-order time-varying differential equations with polynomial coefficients by Eqs. (2.5) and (2.7)

$$\begin{cases} \tilde{\mathbf{x}}'(t) = \tilde{\mathbf{A}}(t)\tilde{\mathbf{x}}(t) + \tilde{\mathbf{B}}(t), & t_0 \leq t \leq t_f, \\ \tilde{\mathbf{x}}|_{t=t_0} = \tilde{\mathbf{x}}(t_0), \end{cases} \quad (4.1)$$

where $\tilde{\mathbf{x}}$ is an n dimensional vector, $\tilde{\mathbf{A}}(t)$ and $\tilde{\mathbf{B}}(t)$ are polynomial matrices,

$$\tilde{\mathbf{A}}(t) = \sum_{i=0}^{r_1} \tilde{\mathbf{A}}_i t^i, \tilde{\mathbf{B}}(t) = \sum_{i=0}^{r_2} \tilde{\mathbf{B}}_i t^i,$$

in which $\tilde{\mathbf{A}}_i$ is an $n \times n$ constant coefficient matrix, $\tilde{\mathbf{B}}_i$ is an $n \times 1$ constant matrix, r_1 and r_2 are constants.

4.1. Unit time-varying differential system

Definition 4.1.

$$\begin{cases} \mathbf{x}'(s) = \mathbf{A}(s)\mathbf{x}(s) + \mathbf{B}(s), & 0 \leq s \leq \rho, \\ \mathbf{x}|_{s=0} = \mathbf{x}(0), \end{cases} \quad (4.2)$$

is called unit time-varying differential system. The normalized parameter ρ is satisfied with $0 < \rho < 1$. $\mathbf{A}(s) = \sum_{i=0}^{r_1} \mathbf{A}_i s^i$, $\mathbf{B}(s) = \sum_{i=0}^{r_2} \mathbf{B}_i s^i$ ($\mathbf{A}_i$ is an $n \times n$ coefficient matrix and $\mathbf{B}_i$ is an $n \times 1$ constant matrix).

For the convergence of the proposed method, Eq. (4.1) must be firstly transformed into Eq. (4.2) by

$$s = \rho \frac{t - t_0}{t_f - t_0} \triangleq \frac{\rho}{l}(t - t_0).$$

Then $s(t_0) = 0$, and

$$\begin{aligned} \mathbf{x}(s) &= \tilde{\mathbf{x}}(t) = \tilde{\mathbf{x}}(t_0 + \frac{l}{\rho}s), \\ \mathbf{A}(s) &= \frac{l}{\rho} \tilde{\mathbf{A}}(t) = \frac{l}{\rho} \tilde{\mathbf{A}}(t_0 + \frac{l}{\rho}s), \\ \mathbf{B}(s) &= \frac{l}{\rho} \tilde{\mathbf{B}}(t) = \frac{l}{\rho} \tilde{\mathbf{B}}(t_0 + \frac{l}{\rho}s). \end{aligned}$$

Obviously, Eq. (4.1) and Eq. (4.2) are equivalent.

4.2. Expanding dimensions

Introducing new variables

$$\mathbf{x}_i(s) = s^i \mathbf{x}(s), (i = 0, 1, \dots, r_1, \dots, m, \dots, \alpha; \alpha > \max(r_1, r_2)),$$

an $(n+1)(\alpha+1)$ dimensional vector is constructed by

$$\mathbf{X}(s) = \left(\mathbf{x}^T \mathbf{x}_1^T \cdots \mathbf{x}_{r_1}^T \cdots \mathbf{x}_m^T \mathbf{x}_{m+1}^T \cdots \mathbf{x}_\alpha^T 1 s \cdots s^{r_2} \cdots s^\alpha \right)^T,$$

in which m and α are preselected positive integers.

Then Eq. (4.2) is expressed by $\mathbf{X}(s)$ as follows

$$\begin{aligned} \mathbf{x}'(s) &= \mathbf{x}'_0(s) \\ &= \mathbf{A}(s)\mathbf{x}(s) + \mathbf{B}(s) \\ &= \sum_{i=0}^{r_1} \mathbf{A}_i s^i \mathbf{x}(s) + \sum_{j=0}^{r_2} \mathbf{B}_j s^j \\ &= \sum_{i=0}^{r_1} \mathbf{A}_i \mathbf{x}_i(s) + \sum_{j=0}^{r_2} \mathbf{B}_j s^j \\ &= (\mathbf{A}_0 \ \mathbf{A}_1 \ \cdots \ \mathbf{A}_{r_1} \ \mathbf{0} \ \cdots \ \mathbf{0} \ \mathbf{B}_0 \ \mathbf{B}_1 \ \mathbf{0} \ \cdots \ \mathbf{B}_{r_2} \ \cdots \ \mathbf{0}) \mathbf{X}(s). \end{aligned} \quad (4.3)$$

By derivative properties of matrices, when $i = 1, 2, \dots, m$, $\mathbf{x}'_i(s)$ is expressed as

$$\begin{aligned} \mathbf{x}'_i(s) &= i s^{i-1} \mathbf{x}(s) + s^i \mathbf{x}'(s) \\ &= i s^{i-1} \mathbf{x}(s) + \mathbf{A}(s)(s^i \mathbf{x}(s)) + \mathbf{B}(s)s^i \\ &= i \mathbf{I}_n \mathbf{x}_{i-1}(s) + \mathbf{A}(s) \mathbf{x}_i(s) + \mathbf{B}(s)s^i. \end{aligned} \quad (4.4)$$

In addition, if $i = m+1, \dots, \alpha$, then $\mathbf{x}'_i(s)$ may be written as another form

$$\mathbf{x}'_i(s) = i s^{i-1} \mathbf{x}(s) + s^i \mathbf{x}'(s) = i \mathbf{I}_n \mathbf{x}_{i-1}(s) + s^i \mathbf{I}_n \mathbf{x}'(s). \quad (4.5)$$

Therefore, substituting Eq. (4.3) to Eqs.(4.4) and (4.5), Eq. (4.2) becomes

$$\begin{cases} M \mathbf{X}'(s) = \hat{\mathbf{C}} \mathbf{X}(s), & 0 \leq s \leq \rho, \\ \mathbf{X}|_{s=0} = (\mathbf{X}(0)^T, \mathbf{0}, \dots, \mathbf{0}, 1, 0, \dots, 0)^T, \end{cases} \quad (4.6)$$

with

$$M = \begin{bmatrix} \mathbf{I} & & & & & & & & & & \\ & \mathbf{I} & & & & & & & & & \\ & & \ddots & & & & & & & & \\ & & & \mathbf{I} & & & & & & & \\ -s^{m+1} \mathbf{I} & & & & \mathbf{I} & & & & & & \\ \vdots & & & & & \ddots & & & & & \\ -s^\alpha \mathbf{I} & & & & & & \mathbf{I} & & & & \\ & & & & & & & 1 & & & \\ & & & & & & & & 1 & & \\ & & & & & & & & & \ddots & \\ & & & & & & & & & & 1 \end{bmatrix},$$

4.5. Back substitution

The solution of Eq. (4.1) in the original interval $[t_0, t_f]$ is given by the inverse transformation

$$\tilde{\mathbf{x}}(t) = \mathbf{x}(s) = \mathbf{x}\left(\frac{\rho}{l}(t - t_0)\right).$$

Furthermore, for general Riccati differential equation, the final solution $y(t)$ of Eq. (2.5) is easily derived from

$$y(t) = \frac{v(t)}{u(t)}.$$

The computing process is described in Algorithm 2.

Algorithm 2 VHPD-TR method for general time-varying Riccati differential equation

Input: time interval $[t_0, t_f]$, initial value $y(t_0) = y_0$, time-step size τ , the extended dimension m, α , scale parameter ρ and $p(t), q(t), r(t)$.

- 1: Form $\tilde{A}(t) = \begin{bmatrix} 0 & -r(t) \\ p(t) & q(t) \end{bmatrix} \triangleq \sum_{i=0}^{r_1} \tilde{A}_i t^i$;
 - 2: Compute $A(s) = \sum_{i=0}^{r_1} \tilde{A}_i s^i$ by $s = \rho \frac{t-t_0}{t_f-t_0}$;
 - 3: Form $\hat{C}$;
 - 4: Compute the transfer matrix $\mathbf{T} = e^{\hat{C}\tau}$ by HPD method;
 - 5: Give the initial value $x(s_0) = [1, y(s_0)]$, $X(1) = X(s_0) = [1, y_0, 0, \dots, 1, 0, \dots, 0]^T$;
 - 6: **for** $i=1$ to $\lceil \frac{\rho}{\tau} \rceil$ **do**
 - 7: $X(i+1) = \mathbf{T} * X(i)$;
 - 8: **end for**
 - 9: Compute the solution $y(s)$ at the integration points $s_k = k\tau$ by $y(s) = \frac{X_1}{X_2}$, where X_1 and X_2 denote the first element and the second element of the solution X , respectively;
 - 10: Give the solution $y(t) = y(s)$ at the points $t_k = \frac{s_k(t_f-t_0)}{\rho} + t_0$;
- Output:** $y(t)$.
-

For special case, the solution of Eq. (2.6) is

$$z(t) = \frac{1}{\tilde{\mathbf{x}}(t)},$$

then the final solution of Riccati differential equation is obtained by the special conditions. The detailed process is listed in Algorithm 3.

Algorithm 3 VHPD-TR method for special Riccati differential equation

Input: time interval $[t_0, t_f]$, initial value $z(t_0) = z_0$, time-step size τ , the extended dimension m, α , scale parameter ρ and $\tilde{q}(t), r(t)$.

- 1: Form $\tilde{q}(t) = \sum_{i=0}^{r_1} \tilde{A}_i t^i$, $r(t) = \sum_{i=0}^{r_2} \tilde{B}_i t^i$;
 - 2: Compute $q(s) = \sum_{i=0}^{r_1} \tilde{A}_i s^i$, $r(s) = \sum_{i=0}^{r_2} \tilde{B}_i s^i$ by $s = \rho \frac{t-t_0}{t_f-t_0}$;
 - 3: Form $\hat{C}$;
 - 4: Compute the transfer matrix $\mathbf{T} = e^{\hat{C}\tau}$ by HPD method;
 - 5: Give the initial value $x(s_0) = \frac{1}{z_0}$, $X(1) = X(s_0) = [z_0, 0, \dots, 1, 0, \dots, 0]^T$;
 - 6: **for** $i=1$ to $\lceil \frac{\rho}{\tau} \rceil$ **do**
 - 7: $X(i+1) = \mathbf{T} * X(i)$;
 - 8: **end for**
 - 9: Compute the solution $z(s)$ at the integration points $s_k = k\tau$ by $z(s) = \frac{1}{X_1}$, where X_1 denotes the first element of the solution X ;
 - 10: Give the solution $z(t) = z(s)$ at the points $t_k = \frac{s_k(t_f-t_0)}{\rho} + t_0$;
- Output:** $z(t)$.
-

5. Convergence and error analysis

5.1. Convergence

Theorem 5.1. *When $m \rightarrow \infty$, the solution of approximate unit time-invariant differential system (4.7) converges to the solution of unit time-varying differential system (4.6).*

Proof. Partition matrices M and $\hat{C}$, they are noted as

$$M = \begin{bmatrix} I_{n(m+1)} & \mathbf{0} & \mathbf{0} \\ -U_s & I_{n(\alpha-m)} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & I_{(\alpha+1)} \end{bmatrix}, \quad \hat{C} = \begin{bmatrix} \hat{A} & \hat{D} & \hat{B} \\ \mathbf{0} & \hat{E} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \hat{F} \end{bmatrix}.$$

Obviously, the inverse of M is

$$M^{-1} = \begin{bmatrix} I_{n(m+1)} & \mathbf{0} & \mathbf{0} \\ U_s & I_{n(\alpha-m)} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & I_{(\alpha+1)} \end{bmatrix},$$

multiplying $\hat{C}$, we have

$$M^{-1}\hat{C} = \begin{bmatrix} \hat{A} & \hat{D} & \hat{B} \\ U_s \hat{A} U_s \hat{D} + \hat{E} U_s \hat{B} \\ \mathbf{0} & \mathbf{0} & \hat{F} \end{bmatrix}.$$

Subtracting Eq. (4.7) from Eq. (4.6), we find

$$\begin{cases} \mathbf{X}'(s) - \hat{\mathbf{X}}'(s) = M^{-1}\hat{C}\mathbf{X}(s) - \hat{C}\hat{\mathbf{X}}(s), & 0 \leq s \leq \rho, \\ (\mathbf{X} - \hat{\mathbf{X}})|_{s=0} = (\mathbf{0}, \mathbf{0}, \dots, \mathbf{0})^T, \end{cases}$$

from which we arrive at

$$\begin{aligned} \mathbf{X}'(s) - \hat{\mathbf{X}}'(s) &= M^{-1}\hat{C}\mathbf{X}(s) - \hat{C}\hat{\mathbf{X}}(s) \\ &= \begin{bmatrix} \hat{A} & \hat{D} & \hat{B} \\ U_s \hat{A} U_s \hat{D} + \hat{E} U_s \hat{B} \\ \mathbf{0} & \mathbf{0} & \hat{F} \end{bmatrix} \mathbf{X}(s) - \begin{bmatrix} \hat{A} & \hat{D} & \hat{B} \\ \mathbf{0} & \hat{E} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \hat{F} \end{bmatrix} \hat{\mathbf{X}}(s) \\ &= \begin{bmatrix} \hat{A} & \hat{D} & \hat{B} \\ \mathbf{0} & \hat{E} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \hat{F} \end{bmatrix} (\mathbf{X}(s) - \hat{\mathbf{X}}(s)) + \begin{bmatrix} \mathbf{0} & \mathbf{0} & \mathbf{0} \\ U_s \hat{A} U_s \hat{D} U_s \hat{B} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} \end{bmatrix} \mathbf{X}(s). \end{aligned}$$

When m is a relatively big positive integer and $0 \leq s \leq \rho < 1$, it is clear that

$$\left\| \begin{bmatrix} \mathbf{0} & \mathbf{0} & \mathbf{0} \\ U_s \hat{A} U_s \hat{D} U_s \hat{B} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} \end{bmatrix} \right\|_1 = s^{m+1} \max(\|\hat{A}_i, \hat{B}_j\|_1) \rightarrow 0.$$

Consequently, when $m \rightarrow \infty$, we get

$$\mathbf{X}'(s) - \hat{\mathbf{X}}'(s) = \begin{bmatrix} \hat{\mathbf{A}} & \hat{\mathbf{D}} & \hat{\mathbf{B}} \\ \mathbf{0} & \hat{\mathbf{E}} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \hat{\mathbf{F}} \end{bmatrix} (\mathbf{X}(s) - \hat{\mathbf{X}}(s)).$$

According to the initial condition

$$(\mathbf{X} - \hat{\mathbf{X}})|_{s=0} = (\mathbf{0}, \mathbf{0}, \dots, \mathbf{0})^T,$$

it is evidently that

$$\hat{\mathbf{X}}(s) \rightarrow \mathbf{X}(s).$$

□

5.2. Error analysis

By the inverse error analysis of the scaling and squaring method [30], the error analysis of VHPD-TR method is as follows.

Theorem 5.2. *If $\frac{\|\hat{\mathbf{C}}\|}{2^N} \leq \frac{1}{2}$, then*

$$[T_h(\frac{\hat{\mathbf{C}}}{2^N})]^{2^N} = e^{\hat{\mathbf{C}} + \mathbf{G}},$$

where $\frac{\|\mathbf{G}\|}{\|\hat{\mathbf{C}}\|} \leq 8(\frac{\|\hat{\mathbf{C}}\|}{2^N})^h \frac{1}{h+1} \leq (\frac{1}{2})^{h-3} \frac{1}{h+1}$.

If ϵ is any error tolerance, the parameters (h, N) can be chosen for which the above inequality implies

$$\frac{\|\mathbf{G}\|}{\|\hat{\mathbf{C}}\|} \leq \epsilon.$$

So the relative error bounds can be derived

$$\frac{\|[T_h(\frac{\hat{\mathbf{C}}}{2^N})]^{2^N} - e^{\hat{\mathbf{C}}}\|}{\|e^{\hat{\mathbf{C}}}\|} = \frac{\|e^{\hat{\mathbf{C}}}(e^{\mathbf{G}} - \mathbf{I})\|}{\|e^{\hat{\mathbf{C}}}\|} \leq \|\mathbf{G}\|e^{\|\mathbf{G}\|} \leq \epsilon\|\hat{\mathbf{C}}\|e^{\epsilon\|\hat{\mathbf{C}}\|}.$$

For convenience, we usually choose $h = 1$ when using HPD method, and the parameter N is determined by a specific issue.

6. Numerical examples

In this section, five numerical experiments are given to demonstrate the performances of the proposed method based on the preceding sections, and to support our theoretical discussions.

6.1. Riccati differential equation with constant coefficients

Consider the following Riccati differential equation [1, 9, 10, 17, 23, 28, 31, 36, 37, 41]

$$\begin{cases} y'(t) = 1 + 2y(t) - y^2(t), & 0 \leq t \leq t_f, \\ y|_{t=0} = 0, \end{cases} \quad (6.1)$$

the exact solution is easily determined to be

$$y(t) = 1 + \sqrt{2} \tanh(\sqrt{2}t + \frac{1}{2} \log \frac{\sqrt{2}-1}{\sqrt{2}+1}).$$

Using the transformation $u(t) = e^{\int_0^t y(t)dt}$, the equation can be firstly transferred to

$$u''(t) - 2u'(t) - u(t) = 0.$$

Let $\tilde{\mathbf{x}}(t) = [u(t), v(t)]^T$, $v'(t) = -u'(t)$, we have

$$\tilde{\mathbf{x}}'(t) = \begin{bmatrix} 0 & 1 \\ 1 & 2 \end{bmatrix} \tilde{\mathbf{x}}.$$

The above time-invariant differential equations can be calculated directly by HPD. Here, we applied the VHPD-TR method to solve the system. Assuming the parameter $\rho = 0.5$, then

$$s = \frac{t}{2t_f}.$$

Eq. (6.1) is converted to

$$\begin{cases} \mathbf{x}'(s) = 2t_f \begin{bmatrix} 0 & 1 \\ 1 & 2 \end{bmatrix} \mathbf{x}(s), & 0 \leq s \leq 1/2, \\ \mathbf{x}|_{s=0} = (1, 0)^T. \end{cases}$$

Next, we discuss different intervals $[0,1]$ ($t_f = 1$) and $[0,4]$ ($t_f = 4$), respectively, to observe the validation of our method, compared with many methods as mentioned in section 1.

Case 1. $t_f = 1$.

Eq. (6.1) has been solved by Adomian's decomposition method (ADM) [41], Homotopy perturbation method (HPM) [1], modified homotopy perturbation method (MHPM) [31], optimal homotopy asymptotic method (OHAM) [28], variation iteration method (VIM) [9], iterative reproducing kernel Hilbert spaces method (IRKHS) [36], Jacobi collocation method [37] and polynomial least squares method (PLSM) [10]. We use the VHPD-TR method to solve it with the parameters ($N = 40, \tau = 0.02, m = 0$). Table 1 compares the absolute errors of different methods, demonstrating that the proposed method has higher precision than the other methods in the given domain, sufficiently accurate to attain machine precision.

Case 2. $t_f = 4$.

Figure 1 compares the absolute errors obtained from different methods in a large interval $[0,4]$. Figures 1(a) and 1(e) indicate that the ADM and HPM method give good approximations only in the neighborhood of the initial point, whereas the VHPD-TR method ($N = 40, \tau = 0.02, m = 0$) is efficient for the whole interval and the absolute error of y reaches 10^{-13} . Figures 1(b)-1(e) demonstrate that the proposed method is superior to B-spline method, multivariational integration method (MVIM) and cardinal method. Moreover, Table 2 shows the relative errors of the exact solution and the results of the VHPD-TR method for different N and τ , compared

Table 1. Comparisons of the absolute errors with different methods for Example 6.1 ($t_f = 1$).

t_i	ADM	HPM	MHPM	OHAM	VIM	IRKHSM	Jacobi	PLSM	VHPD-TR
0.1	2.72e-05	4.40e-10	1.00e-06	3.20e-05	1.98e-08	3.58e-05	7.45e-07	2.53e-07	1.39e-16
0.2	3.34e-04	8.77e-08	1.20e-05	2.90e-04	1.03e-06	7.58e-05	8.51e-07	4.14e-07	1.39e-16
0.3	1.18e-03	1.30e-06	1.00e-06	1.10e-03	8.85e-06	1.20e-04	9.30e-07	7.58e-07	2.22e-16
0.4	2.18e-03	9.55e-08	3.03e-04	2.50e-03	3.33e-05	1.66e-04	1.08e-06	9.01e-07	6.66e-16
0.5	1.85e-03	1.02e-04	1.55e-03	4.40e-03	7.26e-05	2.12e-04	1.14e-06	1.14e-06	1.11e-15
0.6	2.40e-03	9.88e-04	4.69e-03	5.50e-03	9.98e-05	2.52e-04	1.14e-06	1.55e-06	1.11e-15
0.7	1.37e-02	5.65e-03	1.05e-02	5.50e-03	8.84e-05	2.87e-04	1.21e-06	1.74e-06	2.00e-15
0.8	3.49e-02	2.41e-02	1.88e-02	3.80e-03	1.54e-05	3.40e-04	1.04e-06	2.31e-06	2.44e-15
0.9	6.74e-02	8.41e-02	2.80e-02	3.20e-03	4.99e-04	4.90e-04	1.13e-06	3.08e-06	2.66e-15
1.0	1.11e-01	2.53e-01	3.43e-02	3.40e-03	3.47e-03	9.22e-04	4.84e-07	4.39e-06	2.44e-15

with RK4. Firstly, we can see the numerical solution of VHPD-TR converges to the exact solution as the precision parameter N increases, which verifies the validity of the proposed method. Secondly, our method has higher precision than RK4 at the same time step τ , up to 7 orders of precision when $N = 40$. Thirdly, the results of our proposed method experience a negligibly small change as the time-step size increases, whereas the precision of RK4 gradually decreases. The transfer matrix needs to be computed only for once and used forever in the whole computation process, that is to say, it is suitable for large step computation. So we can get a highly precise solution with VHPD-TR if only the precision parameter N is appropriately selected.

Table 2. Comparisons of relative errors using VHPD-TR and RK4 at different time-step sizes for Example 6.1 ($t_f = 4$).

scheme	N	$\tau(s)$	time(s)				
			0.8	1.6	2.4	3.2	4
RK4		0.08	1.0474e-09	5.0362e-10	1.3862e-10	2.4256e-11	3.5540e-12
VHPD-TR	$N=20$	0.08	8.5288e-08	2.4279e-08	3.9639e-09	5.5268e-10	7.1930e-11
	$N=30$	0.08	8.3231e-11	2.3710e-11	3.8710e-12	5.3958e-13	7.0090e-14
	$N=40$	0.08	8.1801e-14	2.2933e-14	3.5219e-15	3.6819e-16	0
RK4		0.16	1.6623e-08	8.1302e-09	2.2565e-09	3.9586e-10	5.8061e-11
VHPD-TR	$N=20$	0.16	1.7046e-07	4.8558e-08	7.9278e-09	1.1054e-09	1.4386e-10
	$N=30$	0.16	1.6646e-10	4.7420e-11	7.7421e-12	1.0793e-12	1.4055e-13
	$N=40$	0.16	1.6294e-13	4.6657e-14	8.1561e-15	1.6569e-15	7.3585e-16
RK4		0.4	6.3368e-07	3.2645e-07	9.2896e-08	1.6417e-08	2.4156e-09
VHPD-TR	$N=20$	0.4	4.2614e-07	1.2139e-07	1.9820e-08	2.7634e-09	3.5965e-10
	$N=30$	0.4	4.1615e-10	1.1855e-10	1.9356e-11	2.6992e-12	3.5174e-13
	$N=40$	0.4	4.0686e-13	1.1565e-13	1.8536e-14	2.0250e-15	1.8396e-16
RK4		0.8	9.7301e-06	5.4814e-06	1.6256e-06	2.9070e-07	4.2993e-08
VHPD-TR	$N=20$	0.8	8.5228e-07	2.4279e-07	3.9639e-08	5.5268e-09	7.1309e-10
	$N=30$	0.8	8.3231e-10	2.3710e-10	3.8710e-11	5.3971e-12	7.0256e-13
	$N=40$	0.8	8.1306e-13	2.3170e-13	3.8000e-14	5.3388e-15	7.3585e-16

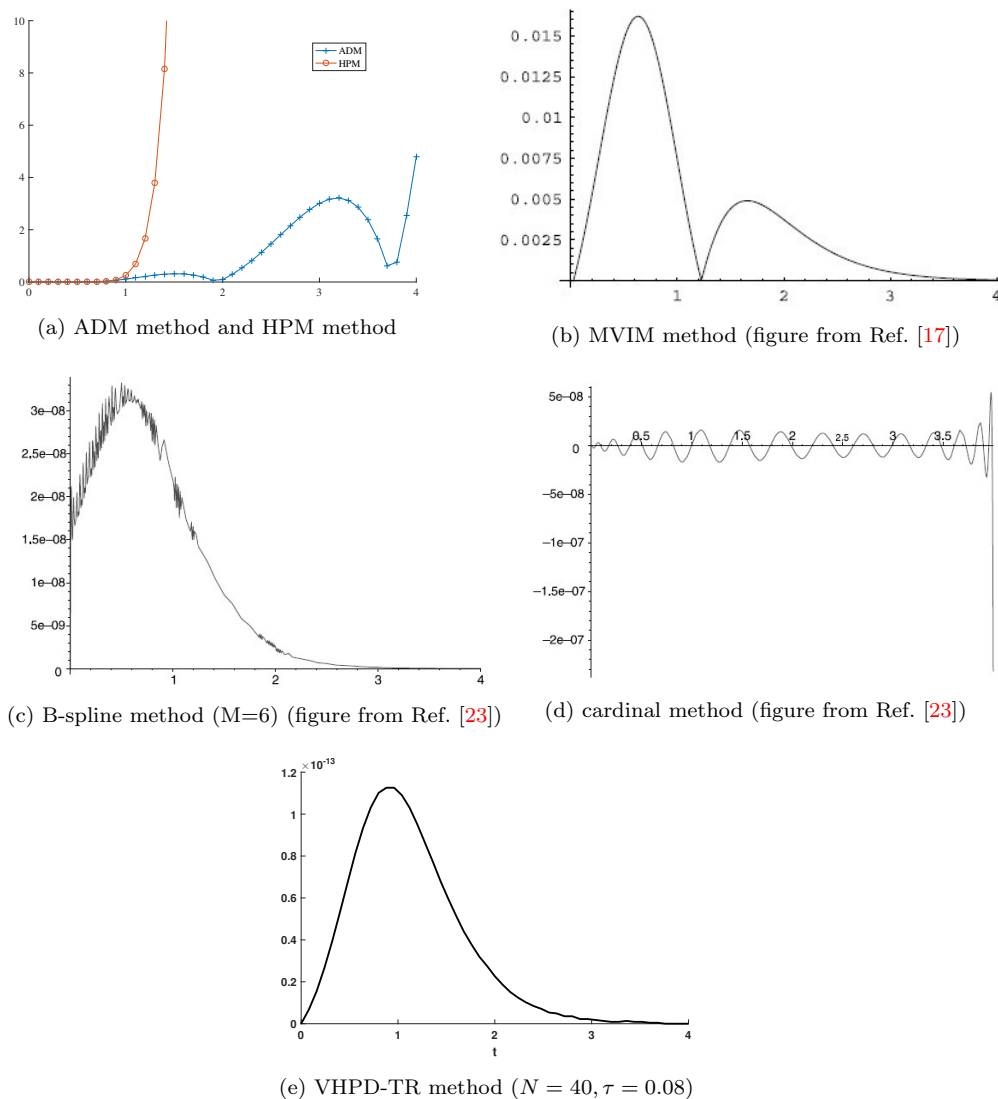


Figure 1. Comparisons of absolute errors with different methods for Example 6.1 ($t_f = 4$).

6.2. Riccati differential equation with a special solution

Consider the following Riccati differential equation [17, 18, 23]

$$\begin{cases} y'(t) = 1 + t^2 - y^2(t), & 0 \leq t \leq 4, \\ y(0) = 1, \end{cases} \quad (6.2)$$

the exact solution of this problem is

$$y(t) = t + \frac{e^{-t^2}}{1 + \int_0^t e^{-x^2} dx}.$$

There are two ways to transform it to the time-varying differential equations. One is transformed into the form of Eq. (2.5) by the transformation $u(t) = e^{\int_0^t y(t) dt}$, the other is converted to the

form of Bernoulli differential equation by a special solution $y = t$. Here, we only discuss the latter. According to $y(t) = t + \varphi(t)$, Eq. (6.2) can be written as

$$\begin{cases} \varphi'(t) = -\varphi^2(t) - 2t\varphi(t), & 0 \leq t \leq 4, \\ \varphi(0) = 1. \end{cases}$$

Next, by introducing a variable $\tilde{x}(t) = \frac{1}{\varphi(t)}$, the system is converted to the following nonhomogeneous time-varying differential equation,

$$\begin{cases} \tilde{x}'(t) = 1 + 2t\tilde{x}(t), & 0 \leq t \leq 4, \\ \tilde{x}(0) = (1, 1)^T. \end{cases}$$

Subsequently, let $s = \frac{\rho}{l}t$, the equation is transformed into the unit time-varying system

$$\begin{cases} x'(s) = \frac{l}{\rho} + 2\left(\frac{l}{\rho}\right)^2 sx(s), & 0 \leq s \leq \rho, \\ x|_{s=0} = (1, 1)^T. \end{cases}$$

We obtain the final results by means of expanding dimensions, approximate time-invariant system, solving by HPD and back substitution. Table 3 lists the results of the proposed method VHPD-TR ($\rho = 0.5$) at $t = 0.8s, 1.6s, 2.4s, 3.2s$ and $4.0s$, with $\tau = 0.04s, \tau = 0.08s, \tau = 0.16s$ and $\tau = 0.4s$, respectively. The data show that the numerical solution can approach machine precision, which verifies the convergence and efficiency of our method at the same time.

Table 3. The relative errors of y with VHPD-TR($N = 50, m = 35, \rho = 1/2$) for Example 6.2.

$\tau(s)$	time(s)				
	0.8	1.6	2.4	3.2	4
0.04	3.9718e-16	6.7637e-16	1.1095e-15	1.1102e-15	8.8818e-16
0.08	7.9437e-16	4.0582e-16	1.8491e-16	6.9389e-16	4.4409e-16
0.16	5.9578e-16	1.3527e-16	1.8491e-16	5.5511e-16	6.6613e-16
0.4	0	9.4692e-16	7.3963e-16	1.2490e-15	1.3323e-15

Furthermore, Figures 2 and 3 describe the influence of parameters m, N , and ρ under the condition of the same time-step size $\tau = 0.08s$. Figure 2 shows that the results are basically unchanged for different parameters ρ when $m = 35$, and $N = 50$, and the accuracy can reach 10^{-15} . So ρ is usually taken as 0.5. Subsequently, Figure 3 shows the maximum relative errors of y for various combinations of $m = 20, 25, 30, 35, 40, 45$ and $N = 20, 30, 40, 50, 60, 70$. It is easy to observe that the larger the parameters m , and N , the higher the precision of numerical solution. When $N = 20$, the maximum relative error of y does not decrease as m increases. Conversely, for fixed $m = 20$, increasing N does not enhance the precision of y either. Therefore, the parameters m and N are important for the precision of the numerical solution, and should be properly preselected.

6.3. Riccati differential equation with $r(t) = \text{constant}$

Consider the following Riccati differential equation

$$\begin{cases} y'(t) = -y^2(t) + (t^2 + t)y(t) - t - 1, & 1 \leq t \leq 3, \\ y(1) = 1, \end{cases} \quad (6.3)$$

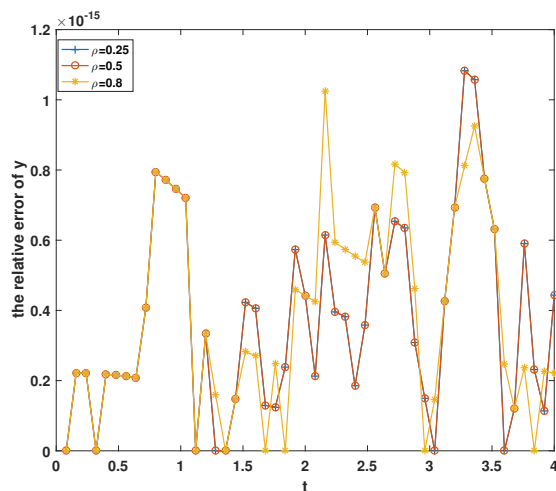


Figure 2. The results of VHPD-TR method for different values of ρ in Example 6.2.

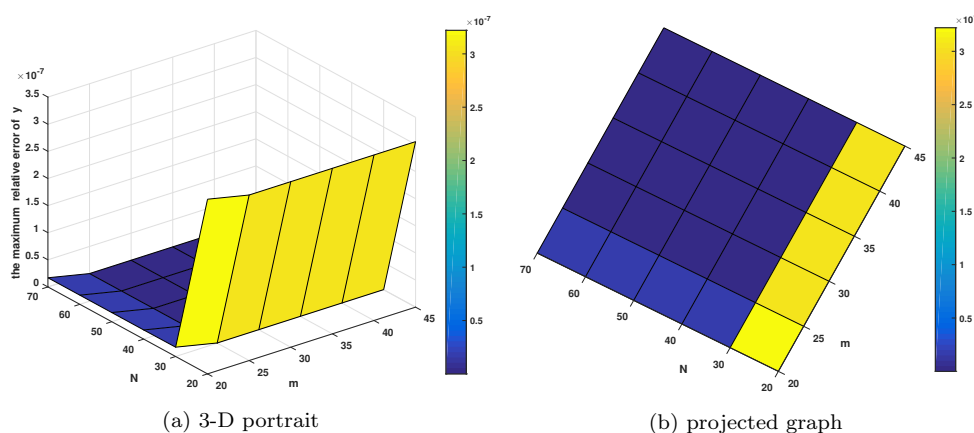


Figure 3. The maximum relative errors of VHPD-TR method for different values of m and N in Example 6.2.

the exact solution is

$$y(t) = \frac{1}{t}.$$

At first, introducing new variables $u(t) = e^{\int_1^t y(t)dt}$, $v(t) = u'(t)$ and $\tilde{\mathbf{x}}(t) = [u(t), v(t)]^T$, Eq. (6.3) can be transformed into

$$\tilde{\mathbf{x}}'(t) = \left(\begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix} + \begin{bmatrix} 0 & 0 \\ -1 & 1 \end{bmatrix} t + \begin{bmatrix} 0 & 0 \\ 0 & 1 \end{bmatrix} t^2 \right) \tilde{\mathbf{x}}(t)$$

with initial value

$$\tilde{\mathbf{x}}(1) = (1, 1)^T.$$

Next, the time interval $[1, 3]$ is changed into $[0, \rho]$ by $s = \rho \frac{t-1}{2}$, and the parameter ρ is chosen freely. The above homogeneous time-varying system becomes the unit time-varying system

$$\mathbf{x}'(s) = \left(\frac{\rho}{l} \begin{bmatrix} 0 & 1 \\ -2 & 2 \end{bmatrix} + \left(\frac{\rho}{l} \right)^2 \begin{bmatrix} 0 & 0 \\ -1 & 3 \end{bmatrix} s + \left(\frac{\rho}{l} \right)^3 \begin{bmatrix} 0 & 0 \\ 0 & 1 \end{bmatrix} s^2 \right) \mathbf{x}(s), \quad s \in [0, \rho].$$

If the time interval is long, we can calculate subsystem by subsystem to improve precision after dividing the interval. First, we choose $\rho = 0.5, m = 25, N = 50$, and time steps=100. Table 4 shows that the numerical results of the interval $([1, 2] \cup [2, 3])$ and the interval $([1, 1.5] \cup [1.5, 2] \cup [2, 2.5] \cup [2.5, 3])$ are better than the results of the interval $([1, 3])$ under the same number of steps. Second, the proposed VHPD-TR method is more accurate than RK4 with the same number of steps. Finally, Figure 4 shows the RMSE (root mean squared error) of VHPD-TR (steps=100, $\rho = 0.5$) with $m = 15, 16, \dots, 49$ and $N = 20, 21, \dots, 54$. RMSE becomes smaller and smaller as m and N increase, which also shows the importance of m is higher than that of N .

Table 4. Comparisons of relative errors using RK4 and VHPD-TR for Example 6.3.

time(s)	RK4 (steps=100)	VHPD-TR($m = 25, N = 50$, steps=100, $\rho=0.5$)		
		$[1, 3]$	$[1, 2] \cup [2, 3]$	$[1, 1.5] \cup [1.5, 2] \cup [2, 2.5] \cup [2.5, 3]$
1.2	1.2440e-9	3.3307e-16	3.3307e-16	3.3307e-16
1.4	2.4345e-9	2.2204e-16	2.2204e-16	2.2204e-16
1.6	2.3371e-9	8.8818e-16	8.8818e-16	2.2204e-16
1.8	9.9158e-10	1.7764e-15	1.7764e-15	4.4409e-16
2.0	1.3855e-8	4.4409e-15	4.4409e-15	1.4433e-15
2.2	6.1694e-8	1.8652e-14	1.2268e-14	5.9397e-15
2.4	2.6756e-7	9.6922e-14	4.6629e-14	2.2982e-14
2.6	1.3593e-6	1.3128e-13	2.3259e-13	1.1158e-13
2.8	8.7174e-6	3.3590e-10	1.4957e-12	7.1060e-13
3.0	7.3064e-5	1.5514e-7	1.2676e-11	5.9427e-12

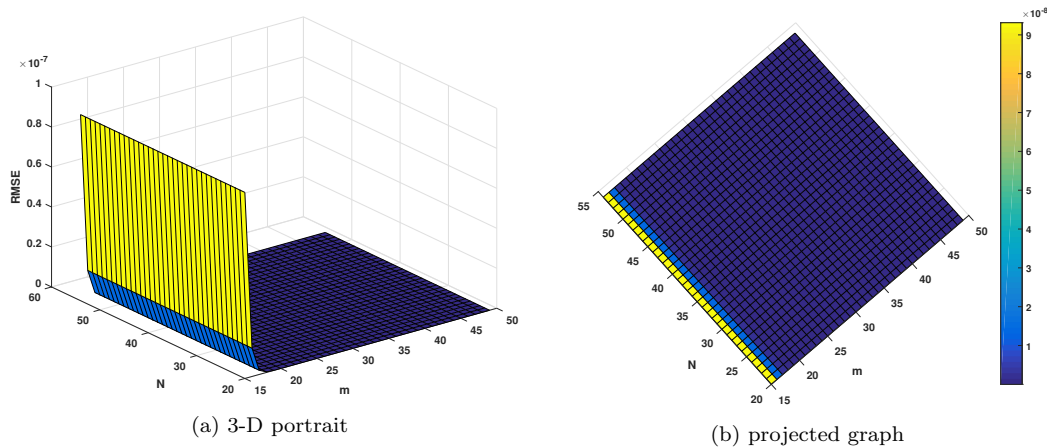


Figure 4. RMSE of VHPD-TR method for different m and N in Example 6.3.

6.4. General time-varying Riccati differential equation

Consider the following Riccati equation

$$\begin{cases} y' = (t-1)y^2 + (1-2t)y + t, & 1 \leq t \leq 3, \\ y(1) = 1 + \frac{1}{1-5e}, \end{cases}$$

the exact solution is

$$y = 1 + \frac{1}{t - 5e^t}.$$

According to Eq. (2.2), the above equation is transformed into

$$\tilde{\mathbf{x}}'(t) = \left(\begin{bmatrix} 0 & 1 \\ 0 & 1 \end{bmatrix} + \begin{bmatrix} 0 & -1 \\ 1 & -2 \end{bmatrix} t \right) \tilde{\mathbf{x}}(t) \quad (6.4)$$

with initial value

$$\tilde{\mathbf{x}}(1) = (1, 1 + \frac{1}{1 - 5e})^T.$$

The entire time interval $[1, 3]$ is cut into two subintervals $([1, 2] \cup [2, 3])$. In $[1, 2]$, let $s = \frac{\rho}{l}(t - 1)$, Eq. (6.4) is transformed into the unit time-varying system.

$$\mathbf{x}'(s) = \left(\frac{\rho}{l} \begin{bmatrix} 0 & 0 \\ 1 & -1 \end{bmatrix} + \left(\frac{\rho}{l} \right)^2 \begin{bmatrix} 0 & -1 \\ 1 & -2 \end{bmatrix} s \right) \mathbf{x}(s), \quad s \in [0, \rho].$$

In $[2, 3]$, let $s = \frac{\rho}{l}(t - 2)$, it is converted to

$$\mathbf{x}'(s) = \left(\frac{\rho}{l} \begin{bmatrix} 0 & -1 \\ 2 & -3 \end{bmatrix} + \left(\frac{\rho}{l} \right)^2 \begin{bmatrix} 0 & -1 \\ 1 & -2 \end{bmatrix} s \right) \mathbf{x}(s), \quad s \in [0, \rho].$$

The relative errors of y with VHPD-TR and RK4 are described in Table 5. It can be seen that when $m > 10$, the accuracy of VHPD-TR ($\rho = \frac{1}{2}, N = 40$) reaches 10^{-15} , which is much higher than that of RK4 at the same time-step size. Moreover, our method is basically not affected by the time-step size under certain conditions of m , whereas the RK4 method decreases rapidly as the time step increases. Subsequently, we choose $m = 10$ and $\tau = 0.01$. Figure 5 shows the logarithm values of the relative errors ($\log_{10} \Delta y = \log_{10} |(y_{\text{compute}} - y)/y|$) over the entire time interval with different N , indicating that the errors gradually become smaller as N increases. When $N = 40$, the precision becomes stable and approaches machine precision.

Next, we analyze the importance of the parameters m, N and time-step size τ , by the average relative errors at different levels of parameters:

- (1) m has four levels, 5, 10, 15, 20,
- (2) N has four levels, 20, 30, 40, 50,
- (3) τ has four levels, 0.01, 0.02, 0.05, 0.1.

Figure 6 demonstrates that both m and N are important for the results, which reach machine precision when $N \geq 40$, and $m \geq 10$. The results of the best choice ($N = 50, m = 15$), are shown in Table 6. The average relative errors are basically not affected by the change of time-step sizes, provided that m and N are properly selected. Consequently, the proposed method is applicable to long time calculations with large time-step sizes.

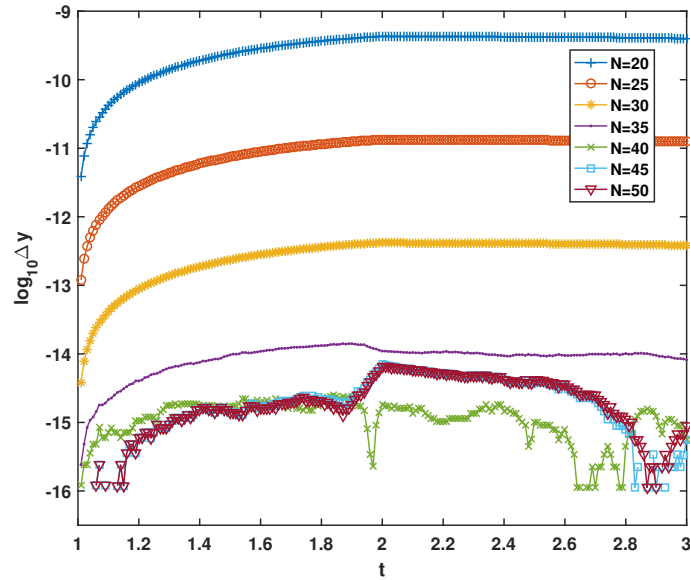
6.5. Stiff Riccati differential equation

Consider the following Riccati equation [38]

$$\begin{cases} y' = -9t^2 y^2 + 36t^2, & 0 \leq t \leq 2, \\ y(0) = 0, \end{cases}$$

Table 5. Comparisons of the relative errors with VHPD-TR ($N=40$) and RK4 for Example 6.4.

scheme	m	$\tau(s)$	time(s)			
			1.5	2	2.5	3
RK4		0.01	2.4278e-12	2.9012e-12	2.5662e-11	2.0140e-12
VHPD-TR	$m=5$	0.01	1.1211e-10	1.6894e-7	9.8558e-8	1.2338e-7
	$m=10$	0.01	1.7490e-15	1.8287e-15	1.0166e-15	5.6087e-16
	$m=15$	0.01	1.7490e-15	3.2002e-15	1.4684e-15	2.6922e-15
RK4		0.02	3.9157e-11	4.6806e-11	4.1401e-11	3.2495e-11
VHPD-TR	$m=5$	0.02	1.1211e-10	1.6894e-7	9.8558e-8	1.2238e-7
	$m=10$	0.02	1.1660e-15	1.9430e-15	7.9069e-16	2.2435e-16
	$m=15$	0.02	1.1660e-15	2.8573e-15	3.3887e-15	3.2530e-15
RK4		0.05	1.5665e-9	1.8738e-9	1.6580e-9	1.3015e-9
VHPD-TR	$m=5$	0.05	1.1211e-10	1.6894e-7	9.8558e-8	1.2238e-7
	$m=10$	0.05	1.6324e-15	1.8287e-15	2.2591e-16	1.4583e-15
	$m=15$	0.05	2.6324e-15	3.0859e-15	2.2591e-15	1.3461e-15
RK4		0.1	2.6082e-8	3.1232e-8	2.7650e-8	2.1711e-8
VHPD-TR	$m=5$	0.1	1.1211e-10	1.6894e-7	9.8558e-8	1.2238e-7
	$m=10$	0.1	1.5158e-15	1.3715e-15	3.3887e-16	1.7948e-15
	$m=15$	0.1	1.5158e-15	2.9716e-15	3.5016e-15	2.5800e-15

**Figure 5.** The results of VHPD-TR ($\tau = 0.01$, $m = 10$) with different N for Example 6.4.

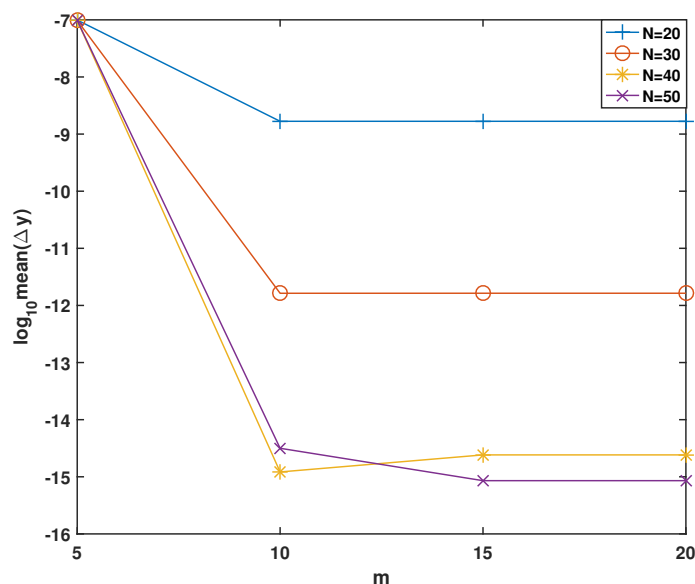
the exact solution is given by

$$y = \frac{2(1 - e^{-12t^3})}{1 + e^{-12t^3}}.$$

This is a stiff equation, the solution contains fast decaying component e^{-12t^3} and the slow decaying component 1, and quickly converges to a steady state solution $\lim_{t \rightarrow +\infty} y = 2$.

Table 6. The relative errors of VHPD-TR with different parameters for Example 6.4.

scheme	m	time(s)			
		$\tau=0.01$	$\tau=0.02$	$\tau=0.05$	$\tau=0.1$
N=20	$m=5$	9.7867e-08	9.8239e-08	9.9355e-08	1.0122e-07
	$m=10$	3.7207e-10	7.4415e-10	1.8604e-09	3.7208e-09
	$m=15$	3.7207e-10	7.4415e-10	1.8604e-09	3.7208e-09
	$m=20$	3.7207e-10	7.4415e-10	1.8604e-09	3.7208e-09
N=30	$m=5$	9.7496e-08	9.7496e-08	9.7497e-08	9.7500e-08
	$m=10$	3.6047e-13	7.2733e-13	1.8129e-12	3.6306e-12
	$m=15$	3.6369e-13	7.2929e-13	1.8154e-12	3.6334e-12
	$m=20$	3.6369e-13	7.2929e-13	1.8154e-12	3.6334e-12
N=40	$m=5$	9.7495e-08	9.7495e-08	9.7495e-08	9.7495e-08
	$m=10$	1.2888e-15	1.0310e-15	1.2863e-15	1.2552e-15
	$m=15$	2.2775e-15	2.6663e-15	2.0809e-15	2.6423e-15
	$m=20$	2.2775e-15	2.6663e-15	2.0809e-15	2.6423e-15
N=50	$m=5$	9.7495e-08	9.7495e-08	9.7495e-08	9.7495e-08
	$m=10$	3.2580e-15	3.2824e-15	2.1857e-15	3.9096e-15
	$m=15$	1.1393e-15	4.8407e-16	9.7275e-16	8.2569e-16
	$m=20$	1.1393e-15	4.8407e-16	9.7275e-16	8.2569e-16

**Figure 6.** Comparisons of the results with VHPD-TR ($\tau = 0.01$, $m = 10$) at different N for Example 6.4.

According to Eq. (2.2), the above equation is transformed into

$$\tilde{\mathbf{x}}'(t) = \begin{bmatrix} 0 & 9 \\ 36 & 0 \end{bmatrix} t^2 \tilde{\mathbf{x}}(t) \quad (6.5)$$

with initial value $\tilde{\mathbf{x}}(1) = (1, 0)^T$. Suppose that $s = \frac{\rho}{t}(t)$, Eq. (6.5) is transformed into the unit

time-varying system, that is used by VHPD-TR method

$$\mathbf{x}'(s) = \left(\frac{\rho}{l}\right)^3 \begin{bmatrix} 0 & 9 \\ 36 & 0 \end{bmatrix} s^2 \mathbf{x}(s), \quad s \in [0, \rho].$$

Stiffness readily leads to the instability of numerical calculation. Table 7 shows the comparisons in the transient region, the accuracy of VHPD-TR ($N = 40, m = 40, \rho = 0.5$) is much higher than that of the symplectic perturbation method [38] and RK4 method with the same time step ($\tau = 0.08s$), and the cumulative errors of RK4 make the solution tend to diverge at the end. Table 8 shows the comparisons of relative errors at different time-step sizes, RK4 method diverges rapidly when it tends to 2 as the step size increases, whereas VHPD-TR can still maintain good stability and accuracy. It demonstrates that our proposed method is suitable for large time step calculations once again and can effectively solve stiff problems.

Table 7. Comparisons of the results at the same time step ($\tau = 0.08s$) for Example 6.5.

scheme	time(s)				
	0.24	0.48	0.72	0.96	1.20
analytic solution	0.1655086	1.1614448	1.9551290	1.9999020	2.0000000
method in [38]	0.1655087	1.1614454	1.9551295	1.9999021	2.0000000
RK4	0.1655042	1.1613221	1.9506266	1.9973800	1.9704478
VHPD-TR	0.1655086	1.1614448	1.9551290	1.9999020	2.0000000

Table 8. Comparisons of the relative errors at different time-step sizes for Example 6.5.

scheme	$\tau(s)$	time(s)				
		0.24	0.72	1.20	1.68	1.92
RK4	0.04	2.0600e-06	9.3978e-05	3.2293e-08	1.2527e-06	2.1829e+00
RK4	0.08	2.6818e-05	2.3029e-03	1.4776e-02	Inf	Inf
RK4	0.12	9.9577e-05	1.6367e-02	6.5432e-01	Inf	Inf
RK4	0.24	1.1416e-04	2.5044e-01	2.1984e+19	Inf	Inf
VHPD-TR	0.04	4.5279e-13	1.4537e-14	2.2500e-11	1.1058e-13	8.8818e-16
VHPD-TR	0.08	9.0524e-13	3.0437e-14	2.2500e-11	1.1102e-13	2.2204e-16
VHPD-TR	0.12	1.3577e-12	4.6110e-14	2.2500e-11	1.124e-13	8.8818e-16
VHPD-TR	0.24	2.7164e-12	9.2560e-14	2.2500e-11	1.1102e-13	1.1102e-16

7. Conclusion

An effective high precise direct integration method (VHPD-TR) is proposed for solving the time-varying Riccati differential equation with polynomial coefficients. In the approach, the Riccati differential equation is approximately transformed into a system of first-order time-invariant differential equations by expanding dimensions and changes of variable. Its analytic solution approaches the exact solution of the Riccati differential equation, and the numerical solution is computed by the HPD method with high precision. All discussed examples show that the VHPD-TR method has higher precision than common methods if the parameters N and m

are properly selected. Moreover, it is suitable for long time computation with large time-step sizes and stiff problems. Future research will extend the proposed numerical framework to solve the Riccati differential equation with time-periodic coefficients and external forcing terms, with applications in physics, engineering, and biology.

Acknowledgments

The authors are grateful to the editors and reviewers for their insightful comments and valuable suggestions.

References

- [1] S. Abbasbandy, *Homotopy perturbation method for quadratic Riccati differential equation and comparison with Adomian's decomposition method*, Applied Mathematics and Computation, 2006, 172(1), 485–490.
- [2] S. Abbasbandy, *Iterated He's homotopy perturbation method for quadratic Riccati differential equation*, Applied Mathematics and Computation, 2006, 175(1), 581–589.
- [3] S. Ahmad, A. Ullah, A. Akgül and M. De la Sen, *A novel homotopy perturbation method with applications to nonlinear fractional order KdV and Burger equation with exponential-decay kernel*, Journal of Function Spaces, 2021. DOI: 10.1155/2021/8770488.
- [4] O. Ala'yed, B. Batiha, D. Alghazo and F. Ghanim, *Cubic B-spline method for the solution of the quadratic Riccati differential equation*, AIMS Mathematics, 2023, 8(4), 9576–9584.
- [5] H. Aminikhah and M. Hemmatnezhad, *An efficient method for quadratic Riccati differential equation*, Communications in Nonlinear Science and Numerical Simulation, 2010, 15(4), 835–839.
- [6] B. D. O. Anderson and J. B. Moore, *Optimal Control: Linear Quadratic Methods*, Dover Publications, Mineola, NY, 2007.
- [7] A. Arafa and A. Hagag, *A new semi-analytic solution of fractional sixth order Drinfeld–Sokolov–Satsuma–Hirota equation*, Numerical Methods for Partial Differential Equations, 2022, 38(3), 372–389.
- [8] R. N. Baghchehjoughi, B. N. Saray and M. Lakestani, *Numerical solution of Riccati differential equation using Legendre scaling functions*, Journal of Current Research in Science, 2014, 2(1), 149–153.
- [9] B. Batiha, M. S. M. Noorani and I. Hashim, *Application of variational iteration method to a general Riccati equation*, International Mathematical Forum, 2007, 2(56), 2759–2770.
- [10] C. Bota and B. Cruntu, *Analytical approximate solutions for quadratic Riccati differential equation of fractional order using the Polynomial Least Squares Method*, Chaos, Solitons & Fractals, 2017, 102, 339–345.
- [11] J. F. Cariñena, G. Marmo, A. M. Perelomov and M. Rañada, *Related operators and exact solutions of Schrödinger equations*, International Journal of Modern Physics A, 1998, 13(28), 4913–4929.
- [12] Y. Chakir, *Application of global rational approximants method to solve nonlinear differential equations: Riccati equations, Logistic growth model and drug consumption model*, Chaos, Solitons & Fractals, 2025. DOI: 10.1016/j.chaos.2025.116089.

- [13] M. Devi, S. Rawan, S. C. Rawan and V. K. Srivastava, *Numerical solutions of Hermite differential equations using Legendre multiwavelets*, BAREKENG: Journal of Mathematics and its Applications, 2026, 20(1), 0691–0710.
- [14] J. Duan, R. Rach, D. Baleanu and A. M. Wazwaz, *A review of the Adomian decomposition method and its applications to fractional differential equations*, Communications in Fractional Calculus, 2012, 3(2), 73–99.
- [15] M. Fu, L. Lan, K. Lu and W. Zhang, *The high order multiplication perturbation method for time-varying dynamic system*, Scientia Sinica: Physica, Mechanica & Astronomica, 2012, 42(2), 185–191.
- [16] Q. Gao, S. Tan, W. Zhong and H. Zhang, *Improved precise integration method for differential Riccati equation*, Applied Mathematics and Mechanics (English Edition), 2013, 34(1), 1–14.
- [17] F. Geng, *A modified variational iteration method for solving Riccati differential equations*, Computers & Mathematics with Applications, 2010, 60(7), 1868–1872.
- [18] F. Geng, Y. Lin and M. Cui, *A piecewise variational iteration method for Riccati differential equations*, Computers & Mathematics with Applications, 2009, 58(11–12), 2518–2522.
- [19] F. Ghomanjani and E. Khorram, *Approximate solution for quadratic Riccati differential equation*, Journal of Taibah University for Science, 2015, 11(2), 246–250.
- [20] Y. Gu, B. Chen, H. Zhang and Z. Guan, *Precise time-integration method with dimensional expanding for structural dynamic equations*, AIAA Journal, 2001, 39(12), 2394–2399.
- [21] M. M. Khader, A. M. Shloof and H. A. Hamead, *Numerical investigation based on the Chebyshev-HPM for Riccati/Logistic differential equations*, AIMS Mathematics, 2025, 10(4), 7906–7919.
- [22] S. R. Khirsariya, S. B. Rao and J. P. Chauhan, *A novel hybrid technique to obtain the solution of generalized fractional-order differential equations*, Mathematics and Computers in Simulation, 2023, 205, 272–290.
- [23] M. Lakestani and M. Dehghan, *Numerical solution of Riccati equation using the cubic B-spline scaling functions and Chebyshev cardinal functions*, Computer Physics Communications, 2010, 181(5), 957–966.
- [24] S. Lal and P. Kumari, *Approximation of functions with bounded derivative and solution of Riccati differential equations by Jacobi wavelet operational matrix*, Applied Mathematics and Computation, 2020, 394, 125834. DOI: 10.1016/j.amc.2020.125834.
- [25] Y. Li, N. Sun, B. Zheng, et al., *Wavelet operational matrix method for solving the Riccati differential equation*, Communications in Nonlinear Science and Numerical Simulation, 2014, 19(3), 483–493.
- [26] J. Lin, W. Shen and F. W. Williams, *A high precision direct integration scheme for structures subjected to transient dynamic loading*, Computers and Structures, 1995, 56(1), 113–120.
- [27] X. Liu, G. Zhou, S. Zhu, et al., *A modified highly precise direct integration method for a class of linear time-varying systems*, Science China - Physics, Mechanics & Astronomy, 2014, 57(7), 1382–1389.
- [28] F. Mabood, A. I. B. Ismail and I. Hashim, *Application of optimal homotopy asymptotic method for the approximate solution of Riccati equation*, Sains Malaysiana, 2013, 42(6), 863–867.

- [29] F. Mohammadi and M. M. Hosseini, *A comparative study of numerical methods for solving quadratic Riccati differential equations*, Journal of the Franklin Institute, 2011, 348(2), 156–164.
- [30] C. Moler and C. Van Loan, *Nineteen dubious ways to compute the exponential of a matrix, twenty-five years later*, SIAM Review, 2003, 45(1), 3–49.
- [31] Z. Odibat and S. Momani, *Modified homotopy perturbation method: Application to quadratic Riccati differential equation of fractional order*, Chaos, Solitons & Fractals, 2008, 36(1), 167–174.
- [32] K. Parand, S. A. Hossayni and J. A. Rad, *Operation matrix method based on Bernstein polynomials for the Riccati differential equation and Volterra population model*, Applied Mathematical Modelling, 2016, 40(2), 993–1011.
- [33] K. Rabiei and M. Razzaghi, *Fractional-order Boubaker wavelets method for solving fractional Riccati differential equations*, Applied Numerical Mathematics, 2021, 168, 221–234.
- [34] B. Rehman, M. A. Bin Iqbal, A. Khan, et al., *Nonstandard finite difference predictor corrector method for quadratic Riccati differential equation*, European Journal of Pure and Applied Mathematics, 2025, 18(2), 5703.
- [35] W. T. Reid, *Riccati Differential Equations*, Mathematics in Science and Engineering, Academic Press, New York, 1972.
- [36] M. G. Sakar, *Iterative reproducing kernel Hilbert spaces method for Riccati differential equations*, Journal of Computational and Applied Mathematics, 2017, 309, 163–174.
- [37] H. Singh and H. M. Srivastava, *Jacobi collocation method for the approximate solution of some fractional-order Riccati differential equations with variable coefficients*, Physica A: Statistical Mechanics and its Applications, 2019, 523, 1130–1149.
- [38] S. Tan and W. Zhong, *Numerical solution of differential Riccati equation with variable coefficients via symplectic conservative perturbation method*, Journal of Dalian University of Technology, 2006, 46(S1), S7–S13.
- [39] Y. Tan and S. Abbasbandy, *Homotopy analysis method for quadratic Riccati differential equation*, Communications in Nonlinear Science and Numerical Simulation, 2008, 13(3), 539–546.
- [40] B. Tang and X. Li, *A new method for determining the solution of Riccati differential equations*, Applied Mathematics and Computation, 2007, 194(2), 431–440.
- [41] M. A. El-Tawil, A. A. Bahnasawi and A. Abdel-Naby, *Solving Riccati differential equation using Adomian's decomposition method*, Applied Mathematics and Computation, 2004, 157(2), 503–514.
- [42] S. A. Wani, S. Patil, W. Ramírez and J. Hernández, *Some families of differential equations for multivariate hybrid special polynomials associated with Frobenius-Genocchi polynomials*, European Journal of Pure and Applied Mathematics, 2025, 18(1), 1–22.
- [43] S. A. Wani, T. U. R. Shah, W. Ramírez and C. Cesarano, *Exploring the properties of multivariable Hermite polynomials in relation to Apostol-type Frobenius-Genocchi polynomials*, Georgian Mathematical Journal, 2025, 32(3), 515–528.
- [44] W. S. Yirga, F. W. Gelu, W. G. Melesse and G. Duressa, *Efficient numerical method for solving a quadratic Riccati differential equation*, Abstract and Applied Analysis, 2024, 1433858. DOI: 10.1155/2024/1433858.

- [45] W. Zhong, *Precise computation for transient analysis*, Journal of Computational Structural Mechanics and Applications, 1995, 12(1), 1–6.
- [46] W. Zhong and F. W. Williams, *A precise time step integration method*, Proceedings of the Institution of Mechanical Engineers, Part C: Journal of Mechanical Engineering Science, 1994, 208(C6), 427–430.
- [47] G. Zhou, Y. Wang, G. Jia and J. Chen, *A homogenized high precise direct integration based on Taylor series*, Journal of Shanghai Jiao Tong University, 2001, 35(12), 1916–1919.

Received August 2025; Accepted March 2026; Available online September 2026.