Open Journal of Statistics, 2012, 2, 352-355
http://dx.doi.org/10.4236/ojs.2012.23043 Published Online July 2012 (http://www.SciRP.org/journal/ojs)

Numerical Solution of Integro-Differential Equations with Local Polynomial Regression

Liyun Su[1*], Tianshun Yan[1], Yanyong Zhao[1], Fenglan Li[2], Ruihua Liu[1]
[1]School of Mathematics and Statistics, Chongqing University of Technology, Chongqing, China
[2]Library, Chongqing University of Technology, Chongqing, China
Email: [*]cloudhopping@163.com

Received April 18, 2012; revised May 20, 2012; accepted June 3, 2012

ABSTRACT

In this paper, we try to find numerical solution of

$$y'(x)=p(x)y(x)+g(x)+\lambda\int_a^b K(x,t)y(t)\mathrm{d}t,\quad y(a)=\alpha.\ a\le x\le b,\quad a\le t\le b$$

or

$$y'(x)=p(x)y(x)+g(x)+\lambda\int_a^x K(x,t)y(t)\mathrm{d}t,\quad y(a)=\alpha.\ a\le x\le b,\quad a\le t\le b$$

by using Local polynomial regression (LPR) method. The numerical solution shows that this method is powerful in solving integro-differential equations. The method will be tested on three model problems in order to demonstrate its usefulness and accuracy.

Keywords: Integro-Differential Equations; Local Polynomial Regression; Kernel Functions

1. Introduction

In recent years, there has been a growing interest in the Integro-Differential Equations (IDEs) which are a combination of differential and Fredholm-Volterra integral equations. IDEs play an important role in many branches of linear and nonlinear functional analysis and their applications in the theory of engineering, mechanics, physics, chemistry, astronomy, biology, economics, potential theory and electrostatics. The mentioned integro-differential equations are usually difficult to solve analytically, so a numerical method is required. Many different methods are used to obtain the solution of the linear and nonlinear IDEs such as the successive approximations, A domain decomposition, Homotopy perturbation method, Chebyshev and Taylor collocation, Haar Wavelet, Tau and Walsh series methods [1-8]. Recently, the authors [9], have used local polynomial regression (LPR) method for the numerical solution of linear and non-linear Fredholm and Volterra integral equations.

In this paper, we consider the linear IDEs,

$$y'(x)=p(x)y(x)+g(x)+\lambda\int_a^x K(x,t)y(t)\mathrm{d}t,\quad y(a)=\alpha. \tag{1}$$

where the upper limit of the integral is constant or variable, λ,α,a are constants, $g(x),p(x)$ and the kernel $K(x,t)$ are given functions, whereas $y(x)$ needs to be determined. The subject of this paper is to try to find numerical solutions of integro-differential equations by means of local polynomial regression method which is presented firstly by Hikmat Caglar [9]. Finally, we show the method to achieve the desired accuracy. Details of the structure of the present method are explained in sections. We apply LPR method for IDEs. In Section 3, it's proved the efficiency of numerical method. Finally, Section 4 contains some conclusions and directions for future expectations and researches.

2. Numerical Method

In this section, we describe local polynomial regression method to find the approximating solution of Equation (1). The following is the mathematical formulation of the local polynomial regression.

2.1. Local Polynomial Regression

First, we introduce the mathematical thoughts of local polynomial regression. This idea was mentioned in [10-14]. Since the form of regression function is not specified, so the data points with long distance from x_0 provide

[*]Corresponding author.

little information to $y(x_0)$. Therefore, we can only use the local data points around x_0. We suppose that $y(x)$ has p + 1 derivative at x_0, for point x, located in the neighborhood of this point x_0, we can use the p-order multivariate polynomials to locally approximate $y(x)$, and the surrounding local point of x_0, so we model $y(x)$ as:

$$y(x) \approx \sum_{j=0}^{p} \beta_j (x - x_0)^j. \tag{2}$$

where parameter β_j depends on x_0, so called local pa-rameter. Obviously, the local parameter $\beta_j = y^{(j)}(x_0)j!$ fits the local model with local data and it can be minimized,

$$\sum_{i=1}^{n} \left[Y_i - \sum_{j=0}^{p} \beta_j (X_i - x_0)^j \right]^2 K\left(\frac{X_i - x}{h} \right) \tag{3}$$

where h controls the size of the bandwidth of local area. Using matrix notation to represent the local polynomial regression is more convenient. Below is the design matrix corresponding to (3) with X and Y:

$$X = \begin{pmatrix} 1 & X_1 - x_0 & \cdots & (X_1 - x_0)^p \\ \vdots & \vdots & \ddots & \vdots \\ 1 & X_n - x_0 & \cdots & (X_n - x_0)^p \end{pmatrix}, Y = \begin{pmatrix} Y_1 \\ Y_2 \\ \vdots \\ Y_n \end{pmatrix}. \tag{4}$$

The weighted least squares problem (3) can be written as

$$\min (Y - X\beta)^T W (Y - X\beta),$$

where,

$$W = diag\left(K_h (X_1 - x_0), \cdots, K_h (X_n - x_0) \right),$$

so the solution vector can be written as

$$\beta = \left(X^T W X \right)^{-1} X^T W Y. \tag{5}$$

Furthermore, we can get the estimation $\hat{y}(x_0)$,

$$\hat{y}(x_0) = E_1 \left(\boldsymbol{X}^T \boldsymbol{W} \boldsymbol{X} \right)^{-1} \boldsymbol{X}^T \boldsymbol{W} \boldsymbol{Y},$$

where E_1 is a column vector (the same size of β) with the first element equal to 1, and the rest equal to zero, that is, $E_1 = (1, 0, \cdots, 0)_{1 \times (p+1)}$. The selection of K does not influence the results much. We selected the quadratic kernel as follows:

$$K(u) = \begin{cases} \frac{3}{4}\left(1 - u^2\right)_+, & \text{if } |u| \le 1; \\ 0, & \text{otherwise.} \end{cases} \tag{6}$$

2.2. Illustration of Numerical Method

In this section, the LPR method for solving Equation (1) is outlined. Let Equation (2) be an approximate solution of IDEs (1):

$$y(x) \approx \sum_{j=0}^{p} \beta_j (x - x_0)^j. \tag{7}$$

where, $X_1 = a < X_2 < \cdots < X_n = b,$ and it is required that the approximate solution (7) satisfies the IDEs at the point $x = X_i$. Putting (7) in (1), it follows that

$$\begin{aligned} &\sum_{j=1}^{p} j\beta_j (x - x_0)^{j-1} - p(x) \cdot \sum_{j=0}^{p} \beta_j (x - x_0)^j \\ &-\lambda \sum_{j=0}^{p} \int_a^x K(x,t)(t - x_0)^j \mathrm{d}t = g(x) \end{aligned} \tag{8}$$

This leads to the system

$$\begin{cases} i = 1, \quad a_{1j} = (X_i - x_0)^j, \quad j = 0, \cdots, p, \\ \qquad\qquad y_1 = y(a) \\ i = 2, \cdots, n \quad a_{ij} = j(X_i - x_0)^{j-1}, \quad j = 1, \cdots, p \\ \qquad b_{ij} = -p(X_i)(X_i - x_0)^j, \\ \qquad c_{ij} = -\lambda \int_a^{X_i} K(X_i, t)(t - x_0)^j \mathrm{d}t \\ \qquad y_i = g(X_i) \end{cases} \tag{9}$$

Consequently, the matrix form (4) can be written as follows by using expression (9).

$$X = \begin{bmatrix} a_{10} & a_{11} & \cdots & a_{1p} \\ a_{20} + b_{20} + c_{20} & a_{21} + b_{21} + c_{21} & \cdots & a_{2p} + b_{2p} + c_{2p} \\ a_{30} + b_{30} + c_{30} & a_{31} + b_{31} + c_{31} & \cdots & a_{3p} + b_{3p} + c_{3p} \\ \vdots & \vdots & \vdots & \vdots \\ a_{n,0} + b_{n,0} + c_{n,0} & a_{n,1} + b_{n,1} + c_{n,1} & \cdots & a_{n,p} + b_{n,p} + c_{n,p} \end{bmatrix} \tag{10}$$

$$Y = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ \vdots \\ y_{n-1} \\ y_n \end{bmatrix} \tag{11}$$

Putting expression (10) and expression (11) in Equation (5), then estimated set of coefficients β_i are obtained by solving matrix system solution. Therefore, approximate solution (7) can be obtained.

3. Simulation and Analysis

In this section, we consider some examples of IDEs. To show the efficiency of the present method for our problem in comparison with the exact solution, we report absolute error which is defined by:

$$\left|Ey_{LPR}\right| = \left|y_{exact} - y_{LPR}\right|,$$

where $\left|Ey_{LPR}\right|$ is absolute error. y_{LPR} is LPR solution. y_{exact} is exact solution. Calculations were all performed by using MATLAB 7.0.

Example 1: First we consider the integro-differential equation:

$$y'(x) = 3e^{3x} - \frac{1}{3}\left(2e^3 + 1\right)x + \int_0^1 3xty(t)\,\mathrm{d}t$$
$$y(0) = 1,$$

For which the exact solution is $y(x) = e^{3x}$.

Some numerical results of these solutions are shown in **Table 1**. We solve example 1 with n = 20, 30, 50 by choosing p = 3 and various values of parameters h presented in **Table 1**. $\left|Ey_{LPR}\right|$ gets up to value 0 which is very accurate at point x = 0 given h = 0.04, n = 50. $\left|Ey_{LPR}\right|$ also gets up to value 1.75 × 10^{-7} which is very small at point x = 0.0 given h = 0.08, n = 20. Moreover, it's showed small absolute error at other point x given different parameters h and n. More importantly, the tabulated results indicate that the absolute errors present decreases more rapidly when parameter n increases.

Example 2: Consider the FIDE:

$$y'(x) = y(x) - \frac{1}{2}x + \frac{1}{1+x} - \ln(1+x) + \frac{1}{\ln^2 2}\int_0^1 \frac{x}{1+t}y(t)\,\mathrm{d}t,$$
$$y(0) = 0,$$

For which the exact solution is $y(x) = \ln(1+x)$.

Some numerical results of these solutions are revealed in **Table 2**. We solve example 2 with n = 20, 30, 50 by choosing p = 3 and various values of parameters h presented in **Table 2**. $\left|Ey_{LPR}\right|$ achieves value 0 which is so accurate at point x = 0 given h = 0.08, n = 20. $\left|Ey_{LPR}\right|$ gets up to value 2.87×10^{-12} which is very small at point x = 0.0 given h = 0.04, n = 50. It's also represented small absolute errors at other point x given kinds of parameters h and n. Further, the tabulated results indicate that the absolute errors reduce rapidly when parameter n increases approximately.

Example 3: At last, we consider the FIDE:

$$y'(x) = y(x) - \cos(2\pi x) - 2\pi\sin(2\pi x) - \frac{1}{2}\sin(4\pi x) + \int_0^1 \sin(4\pi x + 2\pi t)\,y(t)\,\mathrm{d}t,$$
$$y(0) = 1,$$

For which the exact solution is $y(x) = \ln(1+x)$. Some numerical results of these solutions are also shown in **Table 3** which is similar to **Tables 1** and **2**.

Table 1. Absolute errors at point x with p = 3, different h, Example 1.

x	$\left\|Ey_{LPR}\right\|$ $h = 0.08, n = 20$	$\left\|Ey_{LPR}\right\|$ $h = 0.055, n = 30$	$\left\|Ey_{LPR}\right\|$ $h = 0.04, n = 50$
0.0	1.75×10^{-7}	5.01×10^{-10}	0
0.2	3.91×10^{-5}	5.28×10^{-6}	5.11×10^{-10}
0.4	8.12×10^{-4}	1.18×10^{-6}	8.87×10^{-10}
0.6	2.11×10^{-5}	6.22×10^{-6}	5.21×10^{-11}
0.8	7.32×10^{-5}	3.58×10^{-5}	3.64×10^{-9}
1.0	6.59×10^{-4}	8.83×10^{-5}	6.39×10^{-9}

Table 2. Absolute errors at point x with p = 3, different h, Example 2.

x	$\left\|Ey_{LPR}\right\|$ $h = 0.08, n = 20$	$\left\|Ey_{LPR}\right\|$ $h = 0.055, n = 30$	$\left\|Ey_{LPR}\right\|$ $h = 0.04, n = 50$
0.0	0	3.49×10^{-9}	2.87×10^{-12}
0.2	5.12×10^{-4}	4.87×10^{-5}	9.34×10^{-9}
0.4	8.32×10^{-5}	1.72×10^{-5}	6.11×10^{-8}
0.6	6.74×10^{-5}	5.96×10^{-6}	8.27×10^{-10}
0.8	5.12×10^{-5}	4.12×10^{-6}	9.51×10^{-10}
1.0	3.55×10^{-3}	8.85×10^{-5}	3.06×10^{-9}

Table 3. Absolute errors at point x with p = 3, different h, Example 3.

x	$\left\|Ey_{LPR}\right\|$ $h = 0.08, n = 20$	$\left\|Ey_{LPR}\right\|$ $h = 0.055, n = 30$	$\left\|Ey_{LPR}\right\|$ $h = 0.04, n = 50$
0.0	0	0	1.75×10^{-13}
0.2	5.23×10^{-5}	7.19×10^{-6}	2.87×10^{-10}
0.4	1.87×10^{-5}	8.71×10^{-5}	8.76×10^{-10}
0.6	5.62×10^{-4}	7.32×10^{-6}	9.33×10^{-11}
0.8	9.11×10^{-5}	1.58×10^{-6}	3.18×10^{-9}
1.0	4.96×10^{-5}	2.19×10^{-5}	6.82×10^{-10}

In **Table 3**, some numerical results of these solutions are also opened up. We solve example 2 with n = 20, 30, 50 by choosing p = 3 and different values of parameters h presented in **Table 3**. $\left|Ey_{LPR}\right|$ achieves value 0 at point x = 0 given h = 0.08, n = 20. The equivalent result applys to h = 0.0055, n = 30. $\left|Ey_{LPR}\right|$ gets up to value 1.75×10^{-13} which is very small at point x = 0.0 given h = 0.04, n = 50. It's also showed that small absolute errors at other point x given different parameters h and n. From **Table 3**, we can conclude that the absolute errors reduce approximately when parameter n increases.

4. Conclusion

In this paper, we make use of LPR method to solve the linear integro-differential equations. It's showed that this method is very convergent for solving linear integro-differential equations. Moreover, the numerical results approximate the exact solution very well. The Method can be extended to different parameters p, h and kinds of kernel functions. LPR method can also solve nonlinear or integro-differential equations which can be researched and resolved.

5. Acknowledgements

This work was in part supported by Chongqing CSTC foundations of China (Grant No. CSTC, 2010BB2310, Grant No. CSTC, 2011jjA40033), Chongqing CMEC foundations of China (Grant No. KJ080614, Grant No. KJ100810, Grant No. KJ100818, KJ120829).

REFERENCES

[1] B. Asady and M. T. Kajani, "Direct Method for Solving Integro Differential Equations Using Hybrid Fourier and Block-Pluse Functions," *International Journal of Computer Mathematics*, Vol. 82, No. 7, 2007, pp. 889-895. doi:10.1080/00207160412331336044

[2] D. F. Han and X. F. Shang, "Numerical Solution of Integro-Differential Equations by Using CAS Wavelet Operational Matrix of Integration," Vol. 156, No. 2, *Applied Mathematics Computation*, 2007, pp. 460-466. doi:10.1016/j.amc.2007.04.048

[3] A. Golbabai and M. Javidi, "Application of He's Homotopy Perturbation Method for *n*th-Order Integro-Differential Equations," *Applied Mathematics Computation*, Vol. 190, No. 2, 2007, pp. 1409-1416. doi: 10.1016/j.amc.2007.02.018

[4] A. Karamete and M. Sezer, "A Taylor Collocation Method for the Solution of Linear Integro-Differential Equations," *International Journal of Computer Mathematics*, Vol. 79, No. 9, 2002, pp. 987-1000. doi:10.1080/00207160216116

[5] H. Jaradat, O. Alsayyed and S. Al-Shara, "Numerical Solution of Linear Integro-Differential Equations 1," *Journal of Mathematics Statistics*, Vol. 4, No. 4, 2008, pp. 250-254. doi: 10.1.1.152.8509

[6] K. Maleknejad and F. Mizaee, "Numerical Solution of Integro-Differential Equations by Using Rationalized Haar Functions Method," *International Journal of System Mathematics*, Vol. 35, No. 10, 2006, pp. 1735-1744. doi 10.1108/03684920610688694

[7] J. Pour-Mahmoud, M. Y. Rahimi-Ardabili and S. Shahmorad, "Numerical Solution of the System of Fredholm Integro-Differential Equations by the Tau Method," *Applied Mathematics Computation*, Vol. 168, No. 1, 2005, pp. 465-478. doi:10.1016/j.amc.2004.09.026

[8] M. T. Rashed, "Numerical Solution of Functional Differential, Integral and Integro-Differential Equations," *Applied Numerical Mathematics*, Vol. 156, No. 2, 2004, pp. 485-492. doi:10.1016/j.amc.2003.08.021

[9] H. Caglar and N. Caglar, "Numerical Solution of Integral Equations by Using Local Polynomial Regression," *Journal of Computational Analysis and Applications*, Vol. 10, No. 2, 2008, pp. 187-195.

[10] L. Y. Su, Y. Y. Zhao and T. S. Yan, "Two-Stage Method Based on Local Polynomial Fitting for a Linear Heteroscedastic Regression Model and Its Application in Economics," *Discrete Dynamics in Nature and Society*, Vol. 2012, 2012, Article ID 696927. doi:10.1155/2012/696927

[11] L. Y. Su, "Prediction of Multivariate Chaotic Time Series with Local Polynomial Fitting," *Computers Mathematics with Applications*, Vol. 59, No. 2, 2010, pp. 737-744. doi:10.1016/j.camwa.2009.10.019

[12] L. Y. Su, Y. J. Ma and J. J. Li, "Application of Local Polynomial Estimation in Suppressing Strong Chaotic Noise," *Chinese Physics B*, 2012, Vol. 21, No. 2, 2012. doi:10.1088/1674-1056/21/2/020508

[13] J. Fan and I. Gijbels, "Local Polynomial Modelling and Its Applications, Vol. 66 of Monographs on Statistics and Applied Probability," Chapman Hall, London, 1996.

[14] J. Fan and Q. Yao, "Nonlinear Time Series: Nonparametric and Parametric Methods, Springer Series in Statistics," Springer, Berlin, 2003.

