Dhaka Univ. J. Sci. 63(1): 15-18, 2015 (January)

Approximate Solution of Systems of Volterra Integral Equations of Second Kind by Adomian Decomposition Method

Md. Shariful Islam[1], Mir Shariful Islam[*2], Md. Zavid Iqbal Bangalee[3], A.F.M. Khodadad Khan[4] and Amal Halder[5]

[1]*Department of Mathematics,Texas Christian University, USA*

[2]*Department of Oceanography, Dhaka University, Dhaka-1000, Bangladesh*

[3]*Department of Applied Mathematics, Dhaka University, Dhaka-1000, Bangladesh*

[4]*Independent University, Dhaka-1212, Bangladesh*

[5]*Department of Mathematics, Dhaka University, Dhaka-1000, Bangladesh*

(Received: 23 January 2014; Accepted: 05 August 2014)

Abstract

Real life problems that arise in different branches of science and social science, in the form of differential and integral equations are non-linear in nature. However, methods developed in Mathematics, usually, are suitable for the linear system. In this article, we talk on approximating solution of system of Volterra integral equations of second kind in an analytic way using Adomian decomposition method in Mathematica.

Keywords: Adomian polynomial, Volterra integral equations, Mathematica.

I. Introduction

Most of the phenomena that arise in real world are described by non-linear differential, integral equations and integro-differential equations. However, most of the methods developed in mathematics are usually used in solving linear differential and integral equations. The recently developed decomposition methods proposed by American Mathematician, George Adomian 1923-1996[1] have been receiving much attention in recent years in applied and computational mathematics. The Adomian decomposition method has the advantage of converging to the exact solution and this method can be applied directly for all types of differential and integral equations, linear or non linear, homogeneous or inhomogeneous, with constant coefficients or with variable coefficients. These polynomials have been used to solve nonlinear system of Volterra integral equations of second kind[2], System of ordinary differential equations[3], System of integro-differential equations, Nonlinear Strum-liouville problems[6], and two point boundary value problems in nonlinear mechanics[4]. The crucial aspect of the method is employment of the "Adomian polynomials" which allow for solutions convergence of the nonlinear portion of the equation, without simply linearizing the system. These polynomials mathematically generalize to a Maclaurin series about an arbitrary external parameter; which gives the solution method more flexibly than direct Taylor series expansion. There are some analytical-numerical methods[4] to compute adomian polynomials and employ them in solving system of Volterra integral equations which involves tedious cumbersome computations, so it would be convenient to have a Mathematica Program to generate approximate solution to system of Volterra integral equations of second kind by computing this type of polynomials.

II. Adomian Polynomial

Consider the functional equation

$$u = u_0 + f(u) \qquad (1)$$

where u is an unknown function, u_0 is a known function and f is assumed to be a nonlinear operator.

$$u(\lambda) = \sum_{i=0}^{\infty} u_i \lambda^i \qquad (2)$$

is a series solution of (1). From (1), we get

$$\sum_{i=0}^{\infty} u_i \lambda^i = u_0 + f\left(\sum_{i=0}^{\infty} u_i \lambda^i\right) \qquad (3)$$

which may be written as

$$\sum_{i=0}^{\infty} u_i \lambda^i = u_0 + \sum_{i=0}^{\infty} A_i(u_0, \ldots, u_i)\, \lambda^i \qquad (4)$$

Differentiating (4) at $\lambda = 0$, we get $u_{i+1} = A_i$. From (2), (3) and (4), we get $f\big(u(\lambda)\big) = \sum_{i=0}^{\infty} A_i(u_0, \ldots, u_i)\, \lambda^i$

$$\Rightarrow A_i = \frac{1}{i!}\frac{d^i}{d\lambda^i}[f(u(\lambda))] \qquad (5)$$

which gives $A_0 = f(u_0)$. A_i's called Adomian polynomials. An analytic construction of them is referred to work of Baiser[3]. So, an approximation to the solution of (1) may be given by the partial sum $s_n = \sum_{i=0}^{n} u_i$.

Example1: If $f(u) = u^3$ then $A_0 = u_0^3$, $A_1 = 3u_0^2 u_1$, $A_2 = 3u_1^2 u_0 + \frac{3}{2}u_0^2 u_2$, $A_3 = u_1^3 + 3u_0 u_1 u_2 + \frac{1}{2}u_0^2 u_3$

III. Adomian Polynomial and Approximate Solution of System of Volterra Integral Equations

Consider the system of Volterra integral equations of second kind $u_i = g_i + \int_0^t G_i(t, s, u_1, \ldots, u_n)ds$ (6)

where u_i's are unknown functions, g_i's known functions and G_i's in general non-linear operator.

Let $u_{i0} = g_i$ and $u_i = \sum_{j=0}^{\infty} u_{ij}$ (7)

Decomposing G_i in Adomian polynomial, as it is for f in (1), (6) can be written as

$$\sum_{j=0}^{\infty} u_{ij} = g_i + \int_0^t \sum_{j=0}^{\infty} A_{ij}\, ds = g_i + \sum_{j=0}^{\infty} \int_0^t A_{ij}\, ds \qquad (8)$$

On equating both sides $u_{ij} = \int_0^t A_{ij-1} ds$ (9)

* Author for correspondence. *e-mail:*mirsharifuldu77@yahoo.com

for $j \geq 1$. To compute the Adomian polynomial A_{ij}'s, suppose $u_i(\lambda) = \sum_{j=0}^{\infty} u_{ij}\,\lambda^j$, where λ is a parameter. Now G_i may be written as

$$G_i\left(t, s, \sum_{j=1}^{\infty} u_{1j}\,\lambda^j, \dots, \sum_{j=1}^{\infty} u_{nj}\,\lambda^j\right) = \sum_{j=0}^{\infty} A_{ij}\,\lambda^j \tag{10}$$

Differentiating (10) at $\lambda = 0$, we get

$$A_{ij} = \frac{1}{j!}\left[\frac{d^j}{d\lambda^j} G_i(t, s, u_1(\lambda), \dots, u_n(\lambda))\right]_{\lambda=0} \tag{11}$$

It is the identity by which Adomian himself computed his polynomial. From (11) it can be shown that

$$A_{i0} = G_i(t, s, u_{10}, \dots, u_{n0}) \tag{12}$$

and $A_{ij} = A_{ij}(t, s, u_{10}, \dots, u_{1j}, \dots, u_{n0}, \dots, u_{nj})$ (13)

Equation (7), (9) and (11) define a recurrence relations of u_{ij} and A_{ij}. By the recurrence relations, we construct an approximate solution of system (6).

IV. A Simple Algorithm to Compute Adomian Polynomial Suggested by Baiser[5]

Define $A_{i0} = G_i(t, s, u_{10}, \dots, u_{n0})$ then

$$A_{ij} = \frac{1}{j}\frac{d}{d\lambda}\left[A_{ij-1}\left(t, s, u_{10} + \lambda u_{11}, \dots, u_{1j-10} + \lambda u1j, \dots, un0 + \lambda un1, \dots, unj-1 + \lambda unj\right)\right]_{\lambda=0} \tag{14}$$

We employed this algorithm in Mathematica to generate Adomian polynomials and approximate solution to system (6).

Example 2: Let $G(u_1, u_2) = u_1 u_2$. By Baiser's algorithm

$$A_0 = u_{10}u_{20} A_1 = u_{10}u_{21} + u_{11}u_{20} A_2 = \frac{1}{2}(u_{10}u_{22} + 2u_{11}u_{21} + u_{12}u_{20})$$

Example 3: Consider Volterra Integral equation of second kind $u(t) = t + \int_0^t (t-s)u(s)ds$, Comparing this with (6)

$G(t, s, u) = (t-s)u$ and $u_0 = t$. $A_0 = G(t, s, u_0) = (t-s)u_0 u_1 = \int_0^t A_0 ds = \int_0^t s(t-s)ds = -\frac{t^3}{3!}$.

By induction, $A_n = (t-s)u_n$ and $u_n = (-1)^n \frac{t^{2n+1}}{(2n+1)!}$

Therefore, $u = \sum_{j=0}^{\infty} u_i = \sum_{j=0}^{\infty} (-1)^{j+1} \frac{t^{2j+1}}{(2j+1)!} = Sint.$

This example indicates the exactness of the solution of Volterra Integral equation of second kind by Adomian decomposition method.

Theorem: If the series solution $u_i = \sum_{j=0}^{\infty} u_{ij}$ of the system (6) converges, it converges to the exact solution.

Proof: From (7), we get $\sum_{j=0}^{N} u_{ij} = g_i + \int_0^t \sum_0^N A_{ij}\, ds$ (15)

If partial sum on the left hand side of (15) converges to $u_i(t)$ then the partial sum under the integral sign on the right hand side converges to $G_i(t, s, u_1, \dots, u_n)$.

Example 4: Consider the system of Volterra Integral equation of second kind $x(t) = Sin(t) - t + \int_0^t x^2(s) + y^2(s)\, ds$, $y(t) = Cos(t) - \frac{1}{2}Sin^2(t) + \int_0^t x(s)y(s)\, ds$

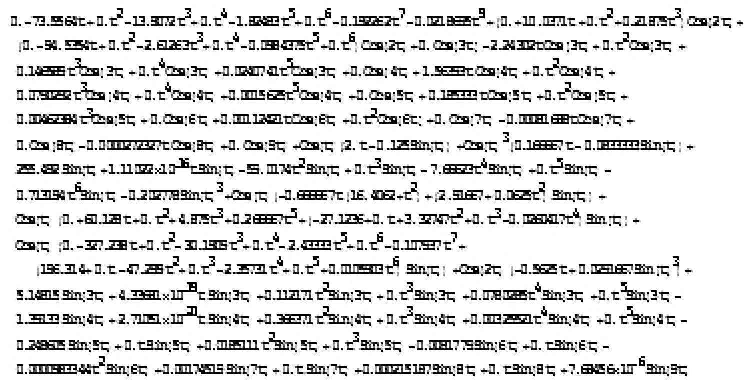
With the exact solution $x(t) = Sin(t)$ and $y(t) = Cos(t)$.

A four term Adomian approximation has been computed in Mathematica from the following programs.

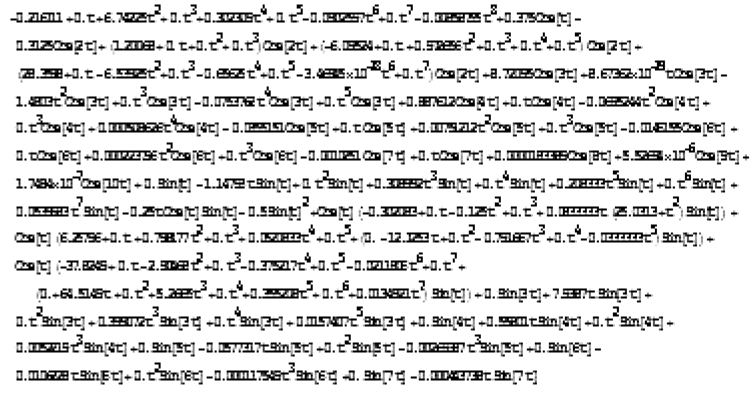
```
G2[x_,y_]:=x^2+y^2;G3[x_,y_]:=x*y;

u[1,0]=Sin[t]-t;u[2,0]=Cos[t]-(1/2.) (Sin[t])^2;

A[1,0]=G2[u[1,0],u[2,0]];

A[2,0]=G3[u[1,0],u[2,0]];

Array[K,{2}];K[1]=1;k[2]=1;m=3;n=2;

Do[yt[i]=u[i,0],{i,1,n}];For[k=1,k<=m, k++,

Do[A[i,k-1]=A[i,k-1]/.t→s,{i,1,n}];
Do[u[i,k]=Integrate[K[i];

A[i,k-1],{s,0,t}],{i,1,n}];

Do[yt[i]=yt[i]+u[i,k],{i,1,n}];

Do[A[i,k-1]=A[i,k-1]/.s→t,{i,1,n}];

Do[Do[A[i,k-1]=A[i,k-1]/.Table[u[l,j] → u[l, j] + (j +
1)V u[l,j+1],{j,0,k}],{l,1,n}],{i,1,n}];

Do[A[i,k]=(D[A[i,k-1],V]/.V→0)/k, {i,1,n}]]

Do[A[i,m]=A[i,m]/.t→s,
{i,1,n}];Do[u[i,m+1]=Integrate[K[i]

A[i,m],{s,0,t}],{i,1,n}];

Do[yt[i]=yt[i]+u[i,m+1],{i,1,n}]
```

A fourth term approximation of $x(t)$ and $y(t)$ are shown below.

$x(t) =$



$y(t) =$



```
For [i=-10,i<=10,i++,
rr[i]=.1  i;
hh[i]=yt[1]/.t→rr[i];
nn[i]=Sin[rr[i]];
er[i]=Abs[hh[i]-nn[i]]]
xx=Table[{rr[i],nn[i],hh[i],er[i]},{i,-10,10}];
yy=Table[{rr[i],hh[i]},{i,-10,10}];
xx//TableForm
```

t	Exact value	A.approx	error
-1.	-0.841471	-0.611083	0.230388
-0.9	-0.783327	-0.629977	0.15335
-0.8	-0.717356	-0.62257	0.0947862
-0.7	-0.644218	-0.590738	0.0534794
-0.6	-0.564642	-0.537771	0.0268717
-0.5	-0.479426	-0.467848	0.0115777
-0.4	-0.389418	-0.385403	0.00401511
-0.3	-0.29552	-0.294525	0.000995558
-0.2	-0.198669	-0.198534	0.000135259
-0.1	-0.0998334	-0.0998291	4.3065×10^{-6}
0	0	0.	0.
0.1	0.0998334	0.0998291	4.3065×10^{-6}
0.2	0.198669	0.198534	0.000135259
0.3	0.29552	0.294525	0.000995558
0.4	0.389418	0.385403	0.00401511
0.5	0.479426	0.467848	0.0115777
0.6	0.564642	0.537771	0.0268717
0.7	0.644218	0.590738	0.0534794
0.8	0.717356	0.62257	0.0947862
0.9	0.783327	0.629977	0.15335
1.	0.841471	0.611083	0.230388

$x(t)$ is compared with it'sAdomian approximation

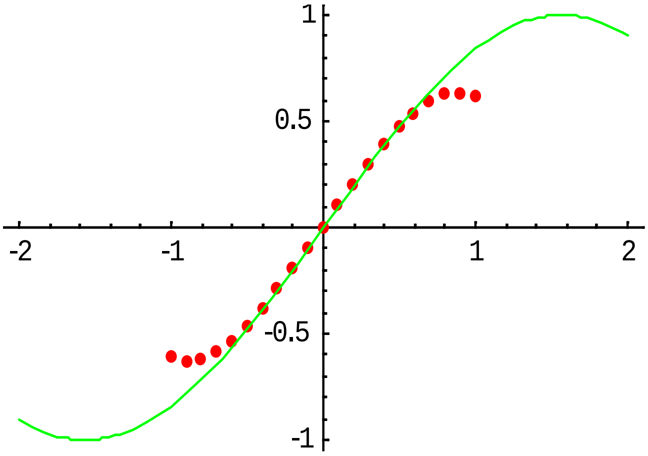
```
For [i=-10,i<=10,i++,
rr[i]=.1  i;
hh[i]=yt[2]/.t->rr[i];
nn[i]=Cos[rr[i]];
er[i]=Abs[hh[i]-nn[i]]]
xxx=Table[{rr[i],nn[i],hh[i],er[i]},{i,-10,10}];
yyy=Table[{rr[i],hh[i]},{i,-10,10}];
xxx//TableForm
```

t	Exact value	A. approx	error
-1.	0.540302	0.477475	0.0628275
-0.9	0.62161	0.581034	0.0405763
-0.8	0.696707	0.672868	0.0238386
-0.7	0.764842	0.752355	0.0124875
-0.6	0.825336	0.819672	0.005664
-0.5	0.877583	0.875457	0.00212567
-0.4	0.921061	0.920449	0.000611756
-0.3	0.955336	0.955219	0.000117092
-0.2	0.980067	0.980056	0.0000108284
-0.1	0.995004	0.995004	1.74562×10^{-7}
0	1	1.	7.10543×10^{-15}
0.1	0.995004	0.995004	1.74562×10^{-7}
0.2	0.980067	0.980056	0.0000108284
0.3	0.955336	0.955219	0.000117092
0.4	0.921061	0.920449	0.000611756
0.5	0.877583	0.875457	0.00212567
0.6	0.825336	0.819672	0.005664
0.7	0.764842	0.752355	0.0124875
0.8	0.696707	0.672868	0.0238386
0.9	0.62161	0.581034	0.0405763
1.	0.540302	0.477475	0.0628275

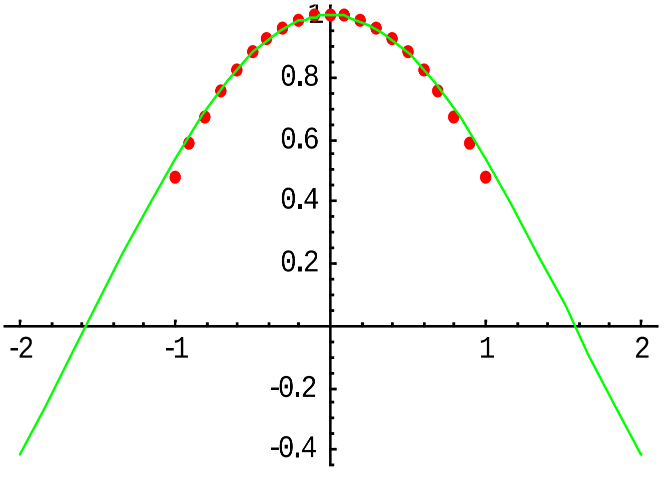
$y(t)$ is compared with it's Adomian approximation,

```
q1=ListPlot[yyy,PlotStyle→{PointSize[.02],Hue[0]},PlotJoined→False];
q2=ListPlot[yy,PlotStyle→{PointSize[.02],Hue[0]},PlotJoined→False];
q3=Plot[Sin[x],{x,-2,2}, PlotStyle→RGBColor[0,1,0]];
q4=Plot[Cos[x],{x,-2,2},PlotStyle→RGBColor[0,1,0]];
Show[q2,q3];
Show[q1,q4];
```

Exact and Adomian approximation of $x(t)$ is shown below:



Exact and Adomian approximation of $y(t)$ is shown.



V.Conclusion

Mathematica reduces huge efforts in approximating solution of system of Volterra Integral equation of second kind. The same program may be used for system of ordinary differential equations, system of integro-differential equations, system of Fredholm integral equations and some other system of functional equations with some sort of adjustment. The routine that we used to generate approximate solution is especially suitable in Mathematica-4. If someone wants to use the same routine, it may require necessary adoption.

Reference

1. Adomian G., G.E. Adomian, 1984. A global method for solution of complex system,*Math. Model,* **5**, 521-568.
2. Babolian E. and J. Baiser, 2000. Solution of system of non-linear Volterra integral equations of the second kind, *Far East J. Math. Sci.* **2 (6),** 935-945.
3. Baiser J., E. Babolian, R. Islam, 2004. Solution of system of ordinary differential equations with Adomian composition method, *Appl. Math. Comput.***147 (3)**, 731-739.
4. Ghosh, S. and D. Roy, 2007. Numeric-analytic form of the Adomian decomposition method for two-point boundary value problems in nonlinear mechanics, *Journal of engineering mechanics*. **133 (10)**, 1124-1133.
5. Baiser J.andS. M. Shafoif, 2007. A simple algorithm for calculating Adomian polynomials, *International journal contemporary math. Science*. **2 (20)**, 975-982.
6. Somali S., and G. Gokmen, 2007.Adomian decomposition method for nonlinear strum-liouville problems, *Surveys in mathematics and its applications*. **2**, 11-20.
7. Deeba, E. and Xie, S., 2000. Numerical approximation for integral equations, IJMMS. **20**,1057-1065.
8. Jerri A.J., 1985. Introduction to integral equations with applications, *New York and Basel: Marcel Dekker*inc., 102-150.
9. Biazar J., M. Ilie, and A. Khoshkenar, 2005. Animprovement to an alternate algorithm for computing adomian polynomials in special cases. Applied mathematics and computation., 523-529.
10. Corduneanu C.,1969. Principles of differential andintegral equations. Boston: Allyn and Beacon, 1-205.