Fourth Order Block-by-block Method to Solve System of Non-linear Volterra Integral Equations of the Second Kind

Borhan F. Jumaa
College of Science - University of Kirkuk

Abstract

In this paper we consider non-linear system of Volterra integral equations of the second kind (**NSVIEK2**). Fourth order block-by-block is modified and applied to solve **NSVIEK2**. A comparison between approximate and exact results for two numerical examples depending on the least-square error are given to show the accuracy of the results obtained by using this method. Programs are written in matlab program version 7.0.

Introduction

A block method is a class of self-starting methods which produce a block of values at a time (Delves & Mohamed, 1985).Second order block-by-block is described, modified, and used to solve Volterra integral equations of the second kind by (Linz, 1967). Also (AL-Asdi, 2002) used second and third order block-by-block for solving Hammersetien Volterra integral equations of the second kind, while (Saify, 2005) used second, third and fourth order block-by-block method for solving a system of linear Volterra integral equation of the second kind, and (Ahmed, 2007) used second and third order block-by-block method for solving a system of non-linear Volterra integral equation of the second kind. In this paper, the approaches of fourth order block-by-block method is applied for the first time to find the numerical solution for a **NSVIEK2** (Jumaa, 2005) which is defined by:

$$F(x) = G(x) + \int_0^x K(x,t,F(t))dt\,, \qquad \ldots(1)$$

where

$$F(x) = (f_1(x),\ldots, f_m(x))^T,\ F(t) = (f_1(t),\ldots, f_m(t)),$$

$$G(x) = (g_1(x),\ldots, g_m(x))^T,$$

$$K(x,t,F(t)) = (k_1(x,t,F(t),\ldots, k_m(x,t,F(t)))^T,$$

The method depends on the use of two three-points quadrature formulas:

$$\int_{x_0}^{x_n} f(x)dx = \frac{h}{3}[f_0 + 4f_1 + 2f_2 + \ldots + 2f_{n-2} + 4f_{n-1} + f_n] - \frac{h^5}{180} f^{(4)}(\varepsilon) \qquad \ldots(2)$$

and Newton-Gregory forward methods (Delves & Mohamed, 1985):

$$p_n(x) = f_0 + \binom{s}{1}\Delta f_0 + \binom{s}{2}\Delta^2 f_0 + \ldots + \binom{s}{n}\Delta^n f_0$$

$$= \sum_{j=0}^{n}\binom{s}{j}\Delta^j f_0 \qquad \ldots(3)$$

where $s = \frac{x - x_0}{h}$, $\Delta^{n+1} f_i = \Delta^n f_{i+1} - \Delta^n f_i$, $n \ge 0$

Hence

$$\int_{x_0}^{x_1} f(x)\,dx = h\left(f_0 + \frac{1}{2}\Delta f_0 - \frac{1}{12}\Delta^2 f_0 + \frac{1}{24}\Delta^3 f_0 - \ldots\right) \qquad \ldots(4)$$

Using five terms of equation (4) we obtain:

$$\int_{x_0}^{x_1} f(x)dx = \frac{h}{720}\left[251 f_0 + 646 f_1 - 264 f_2 + 106 f_3 - 19 f_4\right] + \frac{h^6}{1440} f_0^{(5)}(\varepsilon) \qquad \ldots(5)$$

Newton-Raphson Method(NRM)(Linz, 1969)

The quadrature formulas, when written for a system (1), becomes

$$F_n = G(x_n) + h\sum_{j=0}^{n} w_{nj} K(x_n, t_j, F_j)$$

$$= G(x_n) + h\sum_{j=0}^{n-1} w_{nj} K(x_n, t_j, F_j) + h w_{nn} K(x_n, t_n, F_n) \qquad \ldots(6)$$

where F_n is an m-component vector

$$F_n = ((f_n)_1, (f_n)_2 \ldots, (f_n)_m)^T$$

with $F_0 = G(x_0)$, equation (6) is a nonlinear system to be solved in a stepwise fashion for F_1, F_2, …,F_n such a system can be solved in various ways. Here we will use Newton'-Raphson method.

If we write (6) as

$$F_n - h w_{nn} K(x_n, t_n, F_n) - Q_n = 0 \qquad \ldots(7)$$

where

$$Q_n = G(x_n) + h\sum_{j=0}^{n-1} w_{nj} K(x_n, t_j, F_j) \qquad \ldots(8)$$

Then the Newton iterates are

$$F_n^{(i+1)} = F_n^{(i)} - J^{-1}(F_n^{(i)})\{F_n^{(i)} - h w_{nn} K(x_n, t_n, F_n^{(i)}) - Q_n\} \qquad \ldots(9)$$

Here $F_n^{(i)}$ stands for the value of F_n at the ith iteration, and J is the Jacobian matrix with elements

$$[J(u)]_{ij} = \delta_{ij} - h w_{nn} \frac{\partial k_i(x_n, t_n, u)}{\partial u_i} \qquad \ldots(10)$$

The iterations (9) will be carried out until

$$\left\| F_n^{(i+1)} - F_n^{(i)} \right\| \le \varepsilon$$

where ε is some assigend tolerance.

Block-by-block method

The basic interval [a, b] is divided into mesh points of width *h*, such as $x_j = jh$, $j = 0,1,\ldots,n$ and $nh = b - a$. The approximate solution of $f_i(x)$ will be defined at mesh-points x_j and denoted by f_{ij}; $j = 0,1,\ldots,n$ such as f_{ij} is an approximation to $f_i(x_j)$.For solving system of non-linear Volterra integral equations, rewrite equation (1) as follows:

$$f_i(x_k) = g_i(x_k) + \int_0^{x_{pl}} k_i(x_k, t, F(t))dt + \int_{x_{pl}}^{x_k} k_i(x_k, t, F(t))dt, \qquad \ldots(11)$$

where *p* is some integer and *l* is $\left[\left| k/p \right|\right]$. If the values f_{i0}, f_{i1}, ..., $f_{i,pl}$ are known, then the first integral can be approximated by standard quadrature methods. The second integral is estimated by a quadrature rule using values of the integrand at $t = x_{pl}, x_{pl+1}, \ldots, x_{p(l+1)}$. Since the values of f_i at these points are unknown, we have a system of *mp* non-linear simultaneous equations

$$f_{ik} = g_i(x_k) + h\sum_{j=0}^{lp} w_{kj} k_i(x_k, t_j, f_{1j}, \ldots, f_{mj})$$

$$+ h\sum_{j=0}^{p} w'_{kj} k_i(x_k, t_{lp+j}, f_{1,lp+j}, \ldots, f_{m,lp+j}), \qquad \ldots(12)$$

for $k = lp+1, lp+2, \ldots, (l+1)p$,

where w_{kj}, w'_{kj} depend on the quadrature rule used. Thus, a 'block' of *p* values of f_i is obtained simultaneously.

Modified Method of Fourth-order Block-by-Block:

For this method, we take $p = 4$ the integration over $[0, x_{4l}]$ can be accomplished by Simpson's rule, and the integral over $[x_{4l}, x_k]$ by using a quadratic interpolation of the integrand at the point $x_{4l}, x_{4l+1}, x_{4l+3}, x_{4l+4}$.

Then through the use of equation (11), equation (1) can be written as:

$$f_{i,4l+1} = g_i(x_{4l+1}) + \int_0^{4lh} k_i(x_{4l+1}, t, F(t))dt + \int_{4lh}^{(4l+1)h} k_i(x_{4l+1}, t, F(t))dt \qquad \ldots(13)$$

$$f_{i,4l+2} = g_i(x_{4l+2}) + \int_0^{4lh} k_i(x_{4l+2}, t, F(t))dt + \int_{4lh}^{(4l+2)h} k_i(x_{4l+2}, t, F(t))dt \qquad \ldots(14)$$

$$f_{i,4l+3} = g_i(x_{4l+3}) + \int_0^{4lh} k_i(x_{4l+3}, t, F(t))dt + \int_{4lh}^{(4l+3)h} k_i(x_{4l+3}, t, F(t))dt \quad \ldots(15)$$

$$f_{i,4l+4} = g_i(x_{4l+4}) + \int_0^{4lh} k_i(x_{4l+4}, t, F(t))dt + \int_{4lh}^{(4l+4)h} k_i(x_{4l+4}, t, F(t))dt \quad \ldots(16)$$

$i = 1, 2, \ldots, m$, $l = 0, 1, \ldots$

In practice this method depends on the use of three quadrature formula, Simpson's $\frac{1}{3}$ rule, trapezoidal rule and some quadrature interpolation formula. Therefore, the approximate solution is computed as follows:

$$\begin{aligned} f_{i,4l+1} &= g_i(x_{4l+1}) + \frac{h}{3}\sum_{j=0}^{4l} w_j k_i(x_{4l+1}, t_j, f_{1j}, \ldots, f_{mj}) \\ &+ \frac{h}{720}\big[251k_i(x_{4l+1}, t_{4l}, f_{1,4l}, \ldots, f_{m,4l}) + 646k_i(x_{4l+1}, t_{4l+1}, f_{1,4l+1}, \ldots, f_{m,4l+1}) \\ &- 264k_i(x_{4l+1}, t_{4l+2}, f_{1,4l+2}, \ldots, f_{m,4l+2}) + 106k_i(x_{4l+1}, t_{4l+3}, f_{1,4l+3}, \ldots, f_{m,4l+3}) \\ &- 19k_i(x_{4l+1}, t_{4l+4}, u_{1,4l+4}, \ldots, u_{m,4l+4})\big]. \end{aligned} \quad \ldots(17)$$

$$f_{i,4l+2} = g_i(x_{4l+2}) + \frac{h}{3}\sum_{j=0}^{4l+2} \overline{w}_j k_i(x_{4l+2}, t_j, f_{1j}, \ldots, f_{mj}) \quad \ldots(18)$$

$$\begin{aligned} f_{i,4l+3} &= g_i(x_{4l+3}) + \frac{h}{3}\sum_{j=0}^{4l+2} \overline{w}_j k_i(x_{4l+3}, t_j, f_{1j}, \ldots, f_{mj}) \\ &+ \frac{h}{720}\big[251k_i(x_{4l+3}, t_{4l}, f_{1,4l}, \ldots, f_{m,4l}) + 646k_i(x_{4l+2}, t_{4l+1}, f_{1,4l+1}, \ldots, f_{m,4l+1}) \\ &- 264k_i(x_{4l+3}, t_{4l+2}, f_{1,4l+2}, \ldots, f_{m,4l+2}) + 106k_i(x_{4l+3}, t_{4l+3}, f_{1,4l+3}, \ldots, f_{m,4l+3}) \\ &- 19k_i(x_{4l+3}, t_{4l+4}, u_{1,4l+4}, \ldots, u_{m,4l+4})\big]. \end{aligned} \quad \ldots(19)$$

$$f_{i,4l+4} = g_i(x_{4l+4}) + \frac{h}{3}\sum_{j=0}^{4l+4} z_j k_i(x_{4l+4}, t_j, f_{1j}, \ldots, f_{mj}) \quad \ldots(20)$$

It will have been noticed that in equation (17) the kernel has to be evaluated at the points $[x_{4l+1}, t_{4l+2}, f_{1,4l+2}, \ldots, f_{m,4l+2}]$, $[x_{4l+1}, t_{4l+3}, f_{1,4l+3}, \ldots, f_{m,4l+3}]$, $[x_{4l+1}, t_{4l+4}, f_{1,4l+4}, \ldots, f_{m,4l+4}]$, and in equation (19) the kernel has to be evaluated at the point $[x_{4l+3}, t_{4l+4}, f_{1,4l+4}, \ldots, f_{m,4l+4}]$. This may not be feasible. Thus, replace the last term in equations (17),(19) by formula is derived by using adaptive simpson' s 1/3 rule:

$$\int_{x_0}^{x_1} f(x)\,dx = \frac{h}{6}[f_0 + 4f_{1/2} + f_1] \quad \ldots(21)$$

where $f_{1/2}$ can be found as follows:

Putting s=1/2, n=4 in equation (3), we get:

$$f_{1/2}=\frac{h}{32}\left[\frac{35}{4}f_0+35f_1-\frac{70}{4}f_2+7f_3-\frac{5}{4}f_4\right] \qquad \ldots(22)$$

The resulting equations (17)-(20) become:

$$f_{i,4l+1}=g_i(x_{4l+1})+\frac{h}{3}\sum_{j=0}^{4l}w_j k_i(x_{4l+1},t_j,\,f_{1j},\ldots,\,f_{mj})+\frac{h}{6}\Big[k_i(x_{4l+1},t_{4l},\,f_{1,4l},\ldots,\,f_{m,4l})$$
$$+4k_i\Big(x_{4l+1},t_{4l+0.5},\frac{1}{32}(\frac{35}{4}f_{1,4l}+35f_{1,4l+1}-\frac{70}{4}f_{1,4l+2}+7f_{1,4l+3}-\frac{5}{4}f_{1,4l+4}),\ldots,\frac{1}{32}(\frac{35}{4}f_{1,4l}$$
$$+35f_{m,4l+1}-\frac{70}{4}f_{m,4l+2}+7f_{m,4l+3}-\frac{5}{4}f_{m,4l+4})\Big)+k_i(x_{4l+1},t_{4l+1},\,f_{1,4l+1},\ldots,\,f_{m,4l+1})\Big]. \qquad \ldots(23)$$

$$f_{i,4l+2}=g_i(x_{4l+2})+\frac{h}{3}\sum_{j=0}^{4l+2}\overline{w}_j k_i(x_{4l+2},t_j,\,f_{1j},\ldots,\,f_{mj}) \qquad \ldots(24)$$

$$f_{i,4l+3}=g_i(x_{4l+3})+\frac{h}{3}\sum_{j=0}^{4l+2}\overline{w}_j k_i(x_{4l+3},t_j,\,f_{1j},\ldots,\,f_{mj})+\frac{h}{6}\Big[k_i(x_{4l+3},t_{4l+2},\,f_{1,4l},\ldots,\,f_{m,4l})$$
$$+4k_i\Big(x_{4l+3},t_{4l+2.5},\frac{1}{32}(\frac{35}{4}f_{1,4l}+35f_{1,4l+1}-\frac{70}{4}f_{1,4l+2}+7f_{1,4l+3}-\frac{5}{4}f_{1,4l+4}),\ldots,\frac{1}{32}(\frac{35}{4}f_{1,4l}$$
$$+35f_{m,4l+1}-\frac{70}{4}f_{m,4l+2}+7f_{m,4l+3}-\frac{5}{4}f_{m,4l+4})\Big)+k_i(x_{4l+4},t_{4l+3},\,f_{1,4l+1},\ldots,\,f_{m,4l+1})\Big]. \qquad \ldots(25)$$

$$f_{i,4l+4}=g_i(x_{4l+4})+\frac{h}{3}\sum_{j=0}^{4l+4}Z_j k_i(x_{4l+4},t_j,\,f_{1j},\ldots,\,f_{mj}) \qquad \ldots(26)$$

where $w_0=w_{4l}=1$, $w_j=3-(-1)^j$, $j=1,2,\ldots,4l-1$

$\overline{w}_0=\overline{w}_{4l+2}=1$, $\overline{w}_j=3-(-1)^j$, $j=1,2,\ldots,4l+1$

$Z_0=Z_{4l+4}=1$, $Z_j=3-(-1)^j$, $j=1,2,\ldots,4l+3$

$i=1,2,\ldots,m$, $l=0,1,\ldots$

Therefore, at each step we construct 4m simultaneously non-linear equations from the equation (23)-(26) which can be solved for the unknown's $f_{i,4l+1}$, $f_{i,4l+2}$, $f_{i,4l+3}$ and $f_{i,4l+4}$, $i=1,2,\ldots,m$ by using modified Newton-Raphson method.

Algorithm of MBLM4:

Step (1): Fix $f_{i0}(0)=g_i(0)$, $i=1,2,\ldots,m$.

Step (2): Letting $h=\frac{b}{n}$, $n\in N$.

Step (3): Calculate $g_i(x_{4l+1})$, $g_i(x_{4l+2})$, $g_i(x_{4l+3})$ and $g_i(x_{2l+4})$ for $i=1,2,\ldots,m$.

Step (4): Using equation (23),(24),(25)and (26) to find system of equations for the unknown's $f_{i,4l+1}$, $f_{i,4l+2}$, $f_{i,4l+3}$ and $f_{i,4l+4}$,.

Step (5): Find the value of $f_{i,4l+1}$, $f_{i,4l+2}$, $f_{i,4l+3}$ and $f_{i,4l+4}$, by using NRM.

Step (9): Repeat steps (3)-(5) for *l*=1, 2, ...

Illustrative Examples

In this section, two examples are presented for demonstrating the method and a comparison among the solutions obtained by this method against the exact solution which has been made depending on the least square errors.

Example 1:

As a first example, we study the following non-linear differential equation of the second order:

$$f'' + \sin(x)(\frac{\cos(x)}{f'} - \frac{1}{f}) = -1, \quad x \in (0,1), \; f(0) = 0, \quad f'(0) = 1$$

which can be written as a system of first order non-linear differential equations:

$$f_1^{'} = f_2 \qquad f_1(0) = 0 \qquad \dots(27)$$

$$f_2^{\prime} = \sin(x)(\frac{1}{f_1} - \frac{\cos(x)}{f_2}) - 1 \qquad f_2(0) = 1 \qquad \dots(28)$$

where the exact solution of this problem is:

$$f_1 = \sin(x), \qquad f_2 = \cos(x)$$

By integrating both sides of equations (27)and (28) over $[0,x]$, we obtained the following **NSVIEK2:**

$$f_1(x) = \int_0^x f_2(t)dt$$

$$f_2(x) = 1 - x + \int_0^x \frac{\sin(t)}{f_1(t)}dt - \int_0^x \frac{\sin(t)\cos(t)}{f_2(x)}dt$$

After solving this system by fourth order block-by-block methods with h=0.1 in equations (23)-(26), we obtain the following numerical solution.

Table (1): comparison between the exact solution $\sin(x)$ and the numerical solution $f_1(x)$ and comparison between the exact solution $\cos(x)$ and the numerical solution $f_2(x)$ of Example 1 taking h=0.1.

x	$f_1(x)$		$f_2(x)$	
	Exact solution	MBLM4	Exact solution	MBLM4
0	0	0	1	1
0.1	0.099833	0.090783	0.995004	0.990222
0.2	0.198667	0.199978	0.980066	0.982766
0.3	0.295500	0.298865	0.955336	0.957442
0.4	0.389333	0.399948	0.921061	0.933644
0.5	0.479167	0.495445	0.877583	0.888552
0.6	0.564000	0.587733	0.825336	0.837411
0.7	0.642833	0.646445	0.764842	0.765898
0.8	0.714667	0.717322	0.696707	0.697887
0.9	0.778500	0.779222	0.621609	0.623221
1	0.833333	0.835387	0.540302	0.542661
L.S.E.		7.1174×10^{-3}		4.2070×10^{-4}
R.T		00:05:00		00:05:00

Table (2): shows the least square errors for $f_1(x)$ and $f_2(x)$ with different values of h for Example 1.

Numerical solution of	methods	least square errors		
		h=0.1	h=0.05	h=0.025
$f_1(x)$	MBLM4	7.1174×10^{-3}	1.4424×10^{-4}	1.0322×10^{-4}
$f_2(x)$	MBLM4	4.2070×10^{-4}	6.2225×10^{-5}	1.5022×10^{-5}

Example 2 (Jumaa, 2005):

Solve a system of non-linear VIEK2's:

$$f_1(x)=\frac{1}{4}-\frac{1}{4}e^{2x}+\int_0^x (x-t)\,f_2^2(t)dt$$

$$f_2(x)=-xe^x+2e^x-1+\int_0^x te^{-2f_1(t)}dt$$

The exact solution of this system is:

$$f_1(x)=-\frac{1}{2}x \quad \text{and} \quad f_2(x)=e^x$$

After solving this system by block-by-block methods with h=0.1 in equation (23)-(26), we obtain the following numerical solution.

Table (3): comparison between the exact solution $-\frac{1}{2}x$ and the numerical solution $f_1(x)$ and comparison between the exact solution e^x and the numerical solution $f_2(x)$ of Example 2 taking h=0.1.

x	$f_1(x)$		$f_2(x)$	
	Exact solution	MBLM4	Exact solution	MBLM4
0	0	0	1	1
0.1	-0.050000	-0.049878	1.105170	1.103298
0.2	-0.100000	-0.106711	1.221402	1.216755
0.3	-0.150000	-0.150244	1.349858	1.350211
0.4	-0.200000	-0.213232	1.491824	1.478871
0.5	-0.250000	-0.259972	1.648721	1.628998
0.6	-0.300000	-0.303776	1.822118	1.824599
0.7	-0.350000	-0.359843	2.013752	1.999455
0.8	-0.400000	-0.416077	2.225540	2.245578
0.9	-0.450000	-0.464778	2.459603	2.488993
1	-0.500000	-0.553133	2.718281	2.798210
L.S.E.		3.7302×10^{-3}		6.5732×10^{-3}
R.T		00:04:17		00:04:17

Table (4): shows the least square errors for $f_1(x)$ and $f_2(x)$ with different values of h for Example 3.

Numerical solution of	methods	least square errors		
		h=0.1	h=0.05	h=0.025
$f_1(x)$	MBLM4	3.7302×10^{-3}	3.4578×10^{-5}	0.9884×10^{-6}
$f_2(x)$	MBLM4	6.5732×10^{-3}	4.5443×10^{-5}	8.5455×10^{-6}

Conclusions

According to the numerical results which obtaining from the illustrative examples, we conclude that the method of fourth order block-by-block given the good results. In this method, the error will be decreasing if we choose small values for h (step size) and it is the faster.

References

- Ahmed, CH. SH., (2007): Computational Methods for Solving system of Non-linear Volterra Integral Equations of the Second Kind, B.Sc. Thesis, University of Sulaimani / College of Science, Department of Mathematics.
- AL-Asdi, A. S., (2002): The Numerical Solution of Hammersetien-Volterra-Second Kind-Integral Equations, M.Sc. thesis, University of AL-Mustansiriya.
- Babolian, E. and Biazar, J., (2000): Solution of a System of Non-linear Volterra Integral Equations of the Second Kind, Far East J. Math. Sci. (FJMS), Vol. 2, pp. 935-946.
- Delves, L. M. and Mohamed, J. L., (1985):Computational Method for Integral Equations, Cambridge University.
- Delves, L. M. and Walsh, J. L., (1974): Numerical Solution of Integral Equations, Clarendon Press Oxford.
- Jumaa, B. F., (2005), On Approximate Solutions to a system of Non-linear Volterra Integral Equations, Ph.D. Thesis, University of Technology, Department of Applied Science.
- Linz, P., (1969): A Method for Solving Non-linear Volterra Integral Equations of the Second Kind, Mathematics of Computation,Vol. 23, pp. 595-599.
- Saify, S. A. A., (2005):Numerical Methods for a System of Linear Volterra Integral Equations, M.Sc. thesis, University of Technology.

كتلة–بطريقة–كتلة من الرتبة الرابعة لحل منظومة من معادلات فولتيرا اللاخطّية التكاملية من النوعِ الثانيِ

برهان فخرالدين جمعة
كلية العلوم ـــ جامعة كركوك

الخلاصة

في هذا العملِ، تم دراسة منظومة معادلات فولتيرا التكاملية اللاخطية من النوع الثاني، ثم طورت واستخدمت طريقة كتلة–بطريقة–كتلة من الرتبة الرابعة لحَل هذا المنظومة ، مقارنة بين الحلول التقريبية والمضبوطة لمثالين وبالاعتماد على أخطاء التربيعيات الصغرى قد أعطيت لتعزيز النتائج التي تم الحصول عليها باستخدام هذه الطريقة، تم استخدام برنامج (matlab version 7.0) لكتابة البرامج الخاصة بهذه الطريقة.