*International Journal of Applied Mathematical Research*, 5 (4) (2016) 192-196

**International Journal of Applied Mathematical Research**

*Website: www.sciencepubco.com/index.php/IJAMR*
*doi: 10.14419/ijamr.v5i4.6174*
***Research paper***

# Solving Volterra Integral Equation via Nonlinear Programming

**Jamal A. Othman**[1*]

[1] *University of Information Technology and Communication , Baghdad*
[*] *Corresponding author E-mail:jamal.othman569@gmail.com*

**Abstract**

In this paper we propose an approach to find approximate solution to the nonlinear Volterra integral equation of the second type through a nonlinear programming technique by firstly converting the integral equation into a least square cost function as an objective function for an unconstrained nonlinear programming problem which solved by a nonlinear programming technique (The preconditioned limited- memory quasi-Newton conjugates, gradient method) and as far as we read this is a new approach in the ways of solving the nonlinear Volterra integral equation. We use Maple 11 software as a tool for performing the suggested steps in solving the examples.

***Keywords:*** *Volterra integral equations; Optimization ; Non linear programming.*

## 1. Introduction

Integral equation is a functional equation in which the unknown function found under the sign of integration, in many cases the integral equation originates from the conversion of a boundary-value or an initial-value problem associated with a differential equation, but many problems lead to an integral equations and cannot be formulated as differential equations.[1]

Integral equation has been one of the principal tools in various areas of applied mathematics, physics and engineering encountered in a variety of applications in many fields including continuum mechanics, potential theory, geophysics, electricity and magnetism, antenna synthesis problem, communication theory, mathematical economics, population genetics and radiation, the particle transport problems of astrophysics and reactor theory, fluid mechanics etc ,In recent years, there has been a growing interest in these mathematical field.[2]

Many of integral equations which result from modeling different type of problem are nonlinear , various types of polynomials , have been used by many researchers to develop solutions. Very recently, Maleknejad [3] , Mandal and Bhattacharya [4]and A. Shirin and M. S. Islam [5] used Bernstein polynomials in approximation techniques , Shahsavaran solved by Block Pulse functions [6] Taylor polynomials were also used by Bellour and Rawashdeh [7] and Wang [8] .

Previous work on using Nonlinear Programming technique was for solving nonlinear Fredholm integral equation [9], her in this paper the work is on solving nonlinear Volterra integral equation and the main issue was on how to construrt the objective function of the nonlinear programming problem from the Volterra integral equation which we call it (JV-formula) presented in section 5. We organize this paper as follows. In the next section we briefly speak about the Volterra integral equation. In Section 3, we speaks about the Trapezoidal Rule which we use it through establishing the objective function used to solve our problem, In section 4 we speaks about optimization and nonlinear programming in general and in a subsection of it we introduce the specific nonlinear programming method named "The preconditioned limited-memory quasi-Newton conjugates gradient method" which used to solve our problem after we reform it from Volterra integral equation problem into a nonlinear programming problem , In section 5 we present the steps for the proposed method to reform our problem from Volterra integral equation problem into a nonlinear programming problem to solve it .In section 6 ,we present three example which solved due our proposed method and finally in section 7 we will show conclusion .

## 2. Volterra Integral Equation

An integral equation is an equation in which the unknown function $u(x)$ to be determined appears under the integral sign. A typical from of an integral equation in $u(x)$is as follows:

$$u(x) - \lambda \int_{\alpha(x)}^{\beta(x)} k(x,t)u(t)dt \qquad (1)$$

Where $k(x,t)$ is called the kernel of the integral equation, $\alpha(x)$ and $\beta(x)$ are the limits of integration. [10]

Integral equation arises from various physical and biological models in a wide applications .

In recent years, there has been a growing interest in the Volterra integral equations arising in various fields of physics and engineering [11] e.g., potential theory and Dirichlet problems, electrostatics, the particle transport problems of astrophysics and reactor theory, contact problems, diffusion problems, and heat transfer problems. [12]

Several numerical methods for approximating the solution of nonlinear integral equations are known. The numerical solutions of the nonlinear Volterra-Fredholm integral equations by using homotopy perturbation method was introduced by Tavassoli in [13] ,another method by C. Minggen and D. Hong and [14] used the representation of the exact solution which is given by the form of series for the

nonlinear Volterra-Fredholm integral equations in the reproducing kernel space. rationalized Haar functions to approximate of the nonlinear Volterra-Fredholm-Hammerstein integral equations is another method proposed by M. Razzaghi [15]. Some valid numerical methods, for solving Volterra equations using various polynomials have been developed by many researchers Taylor polynomial solutions used by S. Yalcinbas[16] , Bellour and Rawashdeh and Wang for the nonlinear Volterra-Fredholm integral equations.[17]Bernstein polynomials in approximation techniques used by K. Maleknejad, E. Hashemizadeh [3],and Subhra Bhattacharya, B. N. Mandal[4] .Some other methods were introduced such as single-term Walsh series method [18] , Langrange interpolation [19] , mixed interpolation collocation methods [20], Adomains decomposition method [21] .

## 3. Trapezoidal integral approximation

Trapezoidal Rule is based on Newton-Cotes Formula that states that if one can approximate the integrand as an $n_{th}$ order polynomial $f_n(x)$[22]

$$I = \int_a^b f(x)dx \qquad where \qquad f(x) \approx f_n(x) \tag{2}$$

Then the integral of that function is approximated by the integral of that $n_{th}$ order polynomial.

$$f_n(x) = a_0 + a_1 x + a_2 x^2 + \ldots + a_{n-1}x^{n-1} + a_n x^n \tag{3}$$

$$\int_b^a f(x)dx = (b-a)\frac{f(a)+f(b)}{2} \tag{4}$$

Trapezoidal Rule assumes $n = 1$ , that is, the area under the linear polynomial,

$$\int_a^b f(x)d(x) = \frac{b-a}{2n}\left[f(a) + 2\left\{\sum_{i=1}^{n-1} f(a+ih)\right\} + f(b)\right] \tag{5}$$

Extending this procedure by dividing the interval into equal segments to apply the trapezoidal rule, the sum of the results obtained for each segment is the approximate value of the integral. In this paper we use such formula in the proposed method to approximate the integration in the integral equation; in reformulating the non liner Volterra integral equation as an objective function to be solved.

## 4. Optimization and Nonlinear Programming

Optimization is minimizing or maximizing of a function with constrains or without constrains on its variables. The following is a general formulation of an optimization problem:

$$min_{x\in R^n} f(x) subject \quad to \begin{cases} c_i(x) = 0 \\ c_j(x) \geq 0 \end{cases} \tag{6}$$

$x$ is the vector of variables, also called unknowns or parameters .
$f$ is the objective function, a function of that we want to maximize or minimize .
$c$ is the vector of constraints that the unknowns must satisfy .
Here $f$ and each $c_i, c_j$ are scalar-valued functions.
The process of identifying objective function, variables, and constraints for a given problem is known as modeling. Problems with the general form (4.1) can be classified according to the objective function and constraints, when both of them are linear functions , the problem is a linear programming problem, if any of the constraints or the objective is nonlinear functions the problem is a nonlinear programming problem. [23] Optimization algorithms begin with an initial guess of the optimal values and generate a sequence of improved estimates until they reach a solution. The strategy used to move from one iterate to the next distinguishes one algorithm from another. There are two fundamental strategies for moving from the current point $x_i$ to a new iterate $x_{i+1}$ namely Line search and Trust Region search , In the line search strategy which followed in this paper , the algorithm chooses a direction $P_k$ and searches along this direction from the current iterate $x_k$ for anew iterate with a lower function value, at the new point a new search direction and step length are computed and the process is repeated. There are many algorithms use Line Search strategy to converge to the optimum solution, In this paper we propose an approach to find approximate solution through a nonlinear programming technique called the preconditioned limited-memory Quasi-Newton Conjugates Gradient Method by firstly converting the integral equation into a least square cost function as an objective function for the nonlinear programming problem We use Maple 11 software as a tool for performing the suggested steps in solving the examples.

### 4.1. The Preconditioned Limited-memory Quasi-Newton Conjugated Gradient method

Quasi-Newton conjugates gradient method can be seen as extensions of the conjugate gradient Method, It uses an approximation to the inverse Hessian matrix to steer its search through variable space , at the $k_{th}$ stage of the algorithm we will compute a certain matrix $H_k$ :the matrix $H_k$ is supposed to be an approximation to $H^{-1}(x_k)$ ,the inverse of the Hessian of $f$ at the current iterate [24]
The basic form of the algorithm is as follows:

1. Make an initial guess $x_0$ at the minimum; set $k = 0$ initialize $H_0 = I$ (the $n \times n$ identity matrix).
2. Compute $\nabla f(x_k)$ and take the search direction as $h_k = -H_k \nabla f(x_k)$.
3. Do a line search from $x_k$ in the direction $h_k$ and take $x_{k+1} = x_k + t^* \times h_k$ where $t^*$ minimize $f(x_k + t \times h_k)$ .
4. Compute $H_{k+1}$ by modifying $H_K$ appropriately .This is usually done by setting $H_{k+1} = H_k + U_k, U_k$ is some easy to compute "updating " matrix.
5. Set $k = k+1$ and go to step 2.

The key step is the update from $H_k$ to $H_{K+1}$ .It's required that $H_{k+1}$ satisfy the following Quasi-Newton condition:

$$x_{i+1} - x_i = H_{k+1}\left(\nabla f(x_{i+1}) - \nabla f(x_i)\right), \qquad for \qquad 0 \leq i \leq k \tag{7}$$

A variety of quasi-Newton methods available differ from each other in the way to compute $U_k$ (the updating matrix in step 4),one of the most famous quasi Newton formula is the BFGS (Broyden,Fletcher,Goldfarb,and Shanno)update, which is compute $U_k$ as follows :

$$U_k = \left(1 + \frac{\Delta g_k^T H_k g_k}{\Delta x_k^T \Delta g_k}\right)\frac{\Delta x_k \Delta x_k^T}{\Delta x_k^T \Delta g_k} - \frac{H_k \Delta g_k \Delta x_k^T + \left(H_k \Delta g_k \Delta x_k^T\right)^T}{\Delta x_k^T \Delta g_k} \tag{8}$$

Where

$$\Delta x_i = x_{i+1} - x_i; \Delta g_i = \nabla f(x_{i+1}) - \nabla f(x_i) \tag{9}$$

A less computationally intensive method when n is large is the Limited-Memory BFGS method (LBFGS).Instead of updating and storing the entire approximated inverse Hessian $H_k$ , the LBFGS method never explicitly forms or stores this matrix. Instead it stores information from the past m iterations and uses only this information to implicitly do operations requiring the inverse Hessian (in particular computing the next search direction).[25]

## 5. The Proposed Method

Considering the following form for the nonlinear Volterra integral equation of the second kind as it's in many important cases:

$$u = g(x) + \int_0^x k(x,t,u(t))d(t) \tag{10}$$

The following represent the steps followed to reform the nonlinear Volterra integral equation (5.1) to an objective function (5.6) in which the unknown $u_i(i = 0..9)$ is a solution for both problems (the nonlinear Fredholm integral equation (5.1) and the nonlinear programming problem (5.6):

- Step1: Define the right hand side of equation (5.1) to be equal to $h(x)$

$$h(x) = g(x) + \int_0^x k(x,t,u(t)) \ dx \tag{11}$$

- Step2:Define a least square cost function $f(x)$ as follows:

$$f(x) = \left\| [u(x) - h(x)]^2 \right\| \tag{12}$$

- Step3:Take the norm as integration over the given interval [a,b]

$$f(x) = \int_a^b [u(x) - h(x)]^2 \tag{13}$$

- Step4:Substitution $h(x)$ in (5.4) from (5.2) gives

$$f(x) = \int_a^b \left[ u(x) - g(x) - \int_a^b k(x,t,u(t))dt \right]^2 \tag{14}$$

- Step5:Use Trapezoidal rule to approximate $f(x)$ as follows:

$$\begin{aligned} f(x) \approx F(x) = \frac{1}{20}(b-a) \Bigg( \sum_{j=0}^{9} \Bigg( \Bigg[ u\left(a + \frac{1}{10}j(b-a)\right) - g\left(a + \frac{1}{10}j(b-a)\right) \\ - \frac{1}{20}\left(a + \frac{1}{10}j(b-a)\right) \left( \sum_{i=0}^{9} \left( k\left(a + \frac{1}{10}j(b-a), \frac{1}{10}i\left(a + \frac{1}{10}j(b-a)\right)\right.\right.\right. \\ \left.\left.\left., u\left(\frac{1}{10}i\left(a + \frac{1}{10}j(b-a)\right)\right)\right)\right) + k\left(a + \frac{1}{10}j(b-a), \frac{1}{10}(i+1)\left(a + \frac{1}{10}j(b-a)\right)\right. \\ \left.\left.\left., u\left(\frac{1}{10}(i+1)\left(a + \frac{1}{10}j(b-a)\right)\right)\right)\right)\right)\Bigg]^2 + \Bigg[ u\left(a + \frac{1}{10}(j+1)(b-a)\right) \\ - g\left(a + \frac{1}{10}(j+1)(b-a)\right) - \frac{1}{20}\left(a + \frac{1}{10}(j+1)(b-a)\right) \left( \sum_{i=0}^{9} (k(a \right. \\ \left. + \frac{1}{10}(j+1)(b-a), \frac{1}{10}i\left(a + \frac{1}{10}(j+1)(b-a)\right), u\left(\frac{1}{10}i\left(a + \frac{1}{10}(j+1)(b-a)\right)\right)\right) \\ + k\left(a + \frac{1}{10}(j+1)(b-a), \frac{1}{10}(i+1)\left(a + \frac{1}{10}(j+1)(b-a)\right)\right. \\ \left.\left., u\left(\frac{1}{10}(i+1)\left(a + \frac{1}{10}(j+1)(b-a)\right)\right)\right)\right)\Bigg]^2\Bigg)\Bigg) \end{aligned} \tag{15}$$

(15) is a general formula we call it (JV-formula) to find an approximate solution to a Volterra non linear integral equation of the second kind on an interval $[a,b]$ by considering it as an objective function to be minimized by some nonlinear programming methods ; we use Preconditioned Quasi-Newton Conjugate Gradient Method ( one of the nonlinear programming method) used for non-constrained nonlinear objective functions. We used Maple11 mathematical package which facilitate these method to find an approximate solution on a specific interval $[a,b]$ .

**Table 1:** Comparison between exact and approximate solution

| X | Exact solution | Approximate solution | error |
|---|---|---|---|
| 0 | 0 | -3.667130711026*10$^{-7}$ | -3.6671307110*10$^{-7}$ |
| .1 | .1 | 0.0976276841546660 | 0.00237231585 |
| .2 | .2 | 0.193184567025837 | 0.0068154330 |
| .3 | .3 | 0.281404121700274 | 0.0185958783 |
| .4 | .4 | 0.376109629352903 | 0.0238903706 |
| .5 | .5 | 0.492046863461681 | 0.0079531365 |
| .6 | .6 | 0.550148644075373 | 0.0498513559 |
| .7 | .7 | 0.607424281265213 | 0.0925757187 |
| .8 | .8 | 0.704067225527287 | 0.0959327745 |
| .9 | .9 | 0.747715861709644 | 0.1522841383 |
| 1.0 | 1.0 | 1.04312108608697 | 0.043121086 |

## 6. Numerical Examples

### 6.1. Example 1

Consider the following problem :

$$u(x) = x + (1/5)x^2 - \int_0^x tu(t)^3 \ dt \tag{16}$$

With exact Solution $u(x) = x$.
Table (1)shows comparison between the exact and numerical solution on the interval $[0,1]$.

**Table 2:** Comparison between exact and approximate solution

| X | Exact solution | Approximate solution | error |
|---|---|---|---|
| 0 | 0 | -3.52755567494226*$10^{-7}$ | -3.52755567494226*$10^{-7}$ |
| .1 | .2 | 0.201073312751560 | 0.0010733128 |
| .2 | .4 | 0.403199202244726 | 0.0031992022 |
| .3 | .6 | 0.605957793739756 | 0.0059577937 |
| .4 | .8 | 0.803337562866959 | 0.0033375629 |
| .5 | 1.0 | .986891364811940 | 0.0131086352 |
| .6 | 1.2 | 1.17228380923445 | 0.027716191 |
| .7 | 1.4 | 1.33445688254654 | 0.065543117 |
| .8 | 1.6 | 1.46311419818004 | 0.136885802 |
| .9 | 1.8 | 1.56032024088127 | 0.1522841383 |
| 1.0 | 2.0 | 1.57632993100464 | 0.423670069 |

**Table 3:** Comparison between exact and approximate solution

| X | Exact solution | Approximate solution | error |
|---|---|---|---|
| 0 | 0 | -2.02398780992364*$10^{-7}$ | -2.02398780992364*$10^{-7}$ |
| .1 | 0.0998 | 0.00457812129780025 | 0.00237231585 |
| .2 | 0.198669 | 0.0165202683754665 | 0.0068154330 |
| .3 | 0.29552 | 0.0420113452919495 | 0.0185958783 |
| .4 | 0.389418 | 0.0668496265389050 | 0.0238903706 |
| .5 | 0.4794255 | 0.0649244450079436 | 0.0079531365 |
| .6 | .6 | 0.550148644075373 | 0.0498513559 |
| .7 | .7 | 0.607424281265213 | 0.0925757187 |
| .8 | .8 | 0.704067225527287 | 0.0959327745 |
| .9 | .9 | 0.747715861709644 | 0.1522841383 |
| 1.0 | 1.0 | 1.04312108608697 | 0.043121086 |

### 6.2. Example 2

Consider the following problem :

$$u(x) = 2x - (1/2)x^4 + 0.25 \int_0^x (x-t)u(t)^2 \quad dt \tag{17}$$

With exact solution $u(x) = 2x$.
Table (2)shows comparison between the exact and numerical solution
.

### 6.3. Example 3

Consider the following problem :

$$u(x) = \int_0^x (x-t)u(t) \quad dt \tag{18}$$

With exact Solution $u(x) = \sin x$.
Table (3)shows comparison between the exact and numerical solution on the interval $[0,1]$.

## 7. Conclusion

This paper presents a method to find the solution of a nonlinear Volterra integral equation by an optimization technique that is based on some principles of measure theory, functional analysis and nonlinear programming ,such method seems to be accurate and solves the problems directly, without need of any initial guess.

## References


[1] A.-M. Wazwaz, Linear and Nonlinear Integral Equations Methods and Applications, springer, 2011.

[2] B. B. E. Hashemizadeha, K. Maleknejada, Hybrid functions approach for the nonlinear volterra-fredholm integral equations, Procedia Computer Science 3 (2011) 1189–1194.

[3] R. E. K. Maleknejad, E. Hashemizadeh, A new approach to the numerical solution of volterra integral equations by using bernstein's approximation, Communications in Nonlinear Science and Numerical Simulation 16 (2011) 647–655.

[4] B. N. M. Subhra Bhattacharya, Use of bernstein polynomials in numerical solutions of volterra integral equations, Applied Mathematical Sciences 2 (2008) 1773 – 1787.

[5] A. Shirin, M. S. Islam, Numerical solutions of fredholm integral equations using bernstein.

[6] A. Shahsavaran, Computational method to solve nonlinear integral equations using block pulse functions by collocation method, Applied Mathematical Sciences 5 (2011) 3211 – 3220.

[7] E. A. R. Azzedine Bellour, Numerical solution of first kind integral equations by using taylor polynomials.

[8] W. Wang, A mechanical algorithm for solving the volterra integral equation, Applied Mathematics and Computation 172 (2006) 1323–1341.

[9] J. A. Othman, R. S. Kareem, Solving nonlinear fredholm integral equation of the second type via nonlinear programming techniques, International Journal of Applied Mathematics 29 (2012) 1285–1262.

[10] A. T. Lonseth, Approximate solution of fredholm-type integral equations, Bulletin of the American Mathematical Society 60 (1954) 415–430.

[11] A. J. Jerri, Introduction to Integral Equations with Applications, John Wiley Sons Inc, 1999.

[12] M. K. H. M. K. A. L. N. A. M. M. Rahman, M. A. Hakim, Numerical solutions of volterra integral equations of second kind with the help of chebyshev polynomials, Annals of Pure and Applied Mathematics 1 (2012) 158–167.

[13] M. T. K. M. Ghasemi, E. Babolian, Numerical solutions of the nonlinear volterra-fredholm integral equations by using homotopy perturbation method, Appl. Math. Comput. 188 (2007) 446–449.

[14] C. Minggen, D. Hong, Representation of exact solution for the nonlinear volterra-fredholm integral equations, Appl. Math. Comput. 182 (2006) 1795– 1802.

[15] M.Razzaghi, Y.ordokhani, Solution of nonlinear volterro-hammerstien integral equation via rationalized haar function, Mathematical Problem in Engineering 7 (2001) 205–219.

[16] S. Yalcinbas, Taylor polynomial solutions of nonlinear volterra-fredholm integral equations, Appl. Math. Comput. 127 (2002) 195–206.

[17] M. Zarebnia, A numerical solution of nonlinear volterra-fredholm integral equation, Journal of Applied Analysis and Computation 3 (2013) 95–104.

[18] M. R. Sepehrian, Single-term walsh series method for the volterra integrodifferential equations, Engi. Anal. Boun. Elem. 28 (2004) 1315–1319.

[19] M. Rashed, Numerical solution of functional differential, integral and integro-differential equations, Appl. Numer. Math. 156 (2004) 485–492.

[20] R. M. H. Brunner, A. Makroglou, Mixed interpolation collocation methods for first and second volterra integro-differential equations with periodic solution, Appl. Numer. Math. 23 (1997) 381–402.

[21] S. X. E. Deeba, S.A. Khuri, An algorithm for solving a nonlinear integro-differential equation, Appl. Numer. Math. 115 (2000) 123–131.

[22] W. P. Gerald. C. F., Applied Numerical Analysis, Addison Wesley, 1984.

[23] J. N. . S. J. Wright, Numerical Optimization, Springer-Verlag, 1999.

[24] W. W.Hager, H. Zhang, The limited memory conjugate gradient method, SIAM J. OPTIM. 23 (2013) 2150 – 2168.

[25] A. S. Igor Griva, Stephen G. Nash, Linear and Nonlinear Optimization, 2nd Edition, SIAM BOOK, 2008.