Advances in Pure Mathematics, 2015, 5, 672-682
Published Online September 2015 in SciRes. http://www.scirp.org/journal/apm
http://dx.doi.org/10.4236/apm.2015.511061

Multistage Numerical Picard Iteration Methods for Nonlinear Volterra Integral Equations of the Second Kind

Lian Chen, Junsheng Duan*

School of Sciences, Shanghai Institute of Technology, Shanghai, China
Email: *duanjs@sit.edu.cn

Received 5 July 2015; accepted 4 September 2015; published 7 September 2015

Abstract

Using the Picard iteration method and treating the involved integration by numerical quadrature formulas, we propose a numerical scheme for the second kind nonlinear Volterra integral equations. For enlarging the convergence region of the Picard iteration method, multistage algorithm is devised. We also introduce an algorithm for problems with some singularities at the limits of integration including fractional integral equations. Numerical tests verify the validity of the proposed schemes.

Keywords

Volterra Integral Equation, Picard Iteration Method, Numerical Integration, Multistage Scheme

1. Introduction

The Volterra integral equations arise in many scientific and engineering fields such as the population dynamics, spread of epidemics, semi-conductor devices, vehicular traffic, the theory of optimal control, the kinetic theory of gases and economics [1]-[7]. The initial or boundary value problems for ordinary differential equations and some fractional differential equations can be equivalently expressed by the second-kind Volterra integral equation [6]-[9].

In this work, we consider the general nonlinear Volterra integral equation of the second kind

$$u(t) = C(t) + \int_{t_0}^{t} f\left(t, s, u(s)\right) \mathrm{d}s, \quad t \geq t_0, \tag{1}$$

*Corresponding author.

How to cite this paper: Chen, L. and Duan, J.S. (2015) Multistage Numerical Picard Iteration Methods for Nonlinear Volterra Integral Equations of the Second Kind. *Advances in Pure Mathematics*, **5**, 672-682.
http://dx.doi.org/10.4236/apm.2015.511061

where it permits weak singularity at the limits of integration.

The specific conditions under which a solution exists for the nonlinear Volterra integral equation are considered in [1]-[4] [7]. Many analytical and numerical methods have been proposed for solving this type of equations, such as the linearization and collocation method [10]-[14], the trapezoidal numerical integration and implicit scheme method [15], the implicit multistep collocation methods [16], the reproducing kernel method [17], the wavelet method [18] [19], the Adomian decomposition method [6] [7] [20] and the methods by using function approximation [21]-[23].

The Picard iteration method, or the successive approximations method, is a direct and convenient technique for the resolution of differential equations. This method solves any problem by finding successive approximations to the solution by starting with the zeroth approximation. The symbolic computation applied to the Picard iteration is considered in [24] [25], and the Picard iteration can be used to generate the Taylor series solution for an ordinary differential equation [25].

In this work, we concern on the numerical Picard iteration methods for nonlinear Volterra integral Equation (1). By using the proposed methods, we treat the involved integrals numerically and enlarge the effective region of convergence of the Picard iteration. The rest of the paper is organized as follows. In Section 2, the scheme in a single interval is considered, and the validity of the method is verified by some numerical tests. Basing on the scheme proposed in Section 2, we devise a multistage algorithm in Section 3 for enlarging the convergence region. In Section 4, an algorithm is introduced for problems with some singularity. To show the effectiveness of the proposed algorithms, we perform some numerical results.

2. Numerical Picard Iteration Method for Integral Equations

The Picard iteration scheme for the considered Equation (1) reads [7] [26]

$$\varphi_0(t) = C(t), \tag{2}$$

$$\varphi_{n+1}(t) = C(t) + \int_{t_0}^{t} f\left(t, s, \varphi_n(s)\right)\mathrm{d}s, \quad n \geq 0. \tag{3}$$

The Picard iteration scheme has been applied in almost each textbook on differential equations to mainly prove the existence and uniqueness of solutions. It is direct and easily learned for numerical calculation.

Assume the recursion scheme is convergent for $t \in I = [t_0, T]$. Denote

$$h = (T - t_0)/N, \qquad t_i = t_0 + hi, \quad i = 0, 1, 2, \cdots, N.$$

At $t = t_i$, (3) becomes

$$\varphi_{n+1}(t_i) = C(t_i) + \int_{t_0}^{t_i} f\left(t_i, s, \varphi_n(s)\right)\mathrm{d}s, \; n \geq 0. \tag{4}$$

Treating the integral involved in (4) by numerical quadrature formulas, we have the numerical Picard iteration scheme for (1) over $[t_0, T]$

$$\varphi_0(t_i) = C(t_i), \quad 0 \leq i \leq N, \tag{5}$$

$$\varphi_{n+1}(t_0) = C(t_0), \qquad \varphi_{n+1}(t_i) = C(t_i) + \sum_{j=0}^{i} \omega_{i,j} f\left(t_i, t_j, \varphi_n(t_j)\right), \quad 1 \leq i \leq N, \tag{6}$$

where $n \geq 0$ and $\omega_{i,j}$ are the corresponding weights. Considering the compound trapezoidal formula in (6), the weights are

$$\omega_{i,0} = \omega_{i,i} = \frac{t_i - t_0}{2i} = \frac{1}{2}h, \qquad \omega_{i,j} = \frac{t_i - t_0}{i} = h, \quad 1 \leq j \leq i - 1.$$

Numerical results are given to validate the proposed scheme. Let us start with an example in which the integrand $f\left(t, s, u(s)\right)$ is independent with *t*.

Example 1 *Consider the initial value problem* (*IVP*) *for the nonlinear differential equation*

$$\begin{cases} \dfrac{\mathrm{d}u}{\mathrm{d}t} = -\left(1 + \dfrac{1}{4}u^2\right)^{1/2}, \quad 0 < t \leq T, \\ u(0) = 1. \end{cases}$$

This IVP has the exact solution

$$u^*(t)=2\sin h\left(\sin h^{-1}\frac{1}{2}-\frac{t}{2}\right)$$

The equivalent integral equation of the IVP is

$$u(t)=1-\int_0^t\left(1+\frac{1}{4}u^2(s)\right)^{1/2}\mathrm{d}s.$$

Denote φ_n^N the result after n iterations when discretization parameter N is taken. Take T = 10, N = 20. **Figure 1(a)** and **Figure 1(b)** show the results of the first 5 iterations and the errors at T for each iteration respectively. It’s shown in the figure that, the iterative solution converges exponentially respect to iteration number n.

The relative errors $\left|\frac{\varphi_n^N(T)-u^*(T)}{u^*(T)}\right|$ are larger than 10^{-2} when N = 20. For higher accuracy, more nodes in numerical integration are needed. For each fixed N, iterations stop when $\left|\varphi_{n+1}^N(T)-\varphi_n^N(T)\right|<10^{-14}$. Errors for $N=10,20,40,80,160,320$ are plotted in **Figure 2(a)**. Especially at T, we report the dependence of the error on n and N in **Figure 2(b)** and **Figure 2(c)**, respectively. The figures show that the errors increase respect to t and decrease respect to n exponentially, and decrease respect to N at an order about $\log\left(10^{-3}\right)/\log\left(320/10\right)\approx-2$.

Next we give an example with t-dependent integrand.

Example 2 *Consider the pendulum equation*

$$\begin{cases}\dfrac{\mathrm{d}^2u}{\mathrm{d}t^2}+\sin u=0, & 0<t\le T,\\ u(0)=0,u'(0)=1.\end{cases}\tag{7}$$

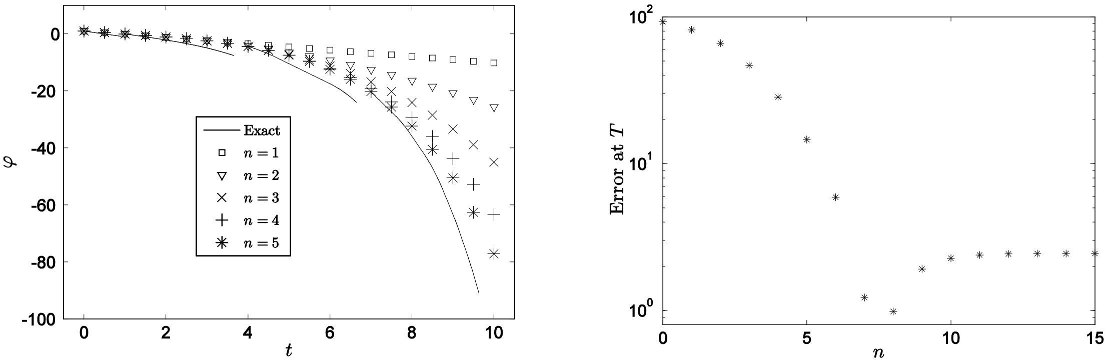
The exact solution can be expressed in terms of the Jacobi elliptic function

$$u^*(t)=2\arcsin\left(\frac{1}{2}sn\left(t,\frac{1}{4}\right)\right).$$

Integrating the differential equation in (7) yields

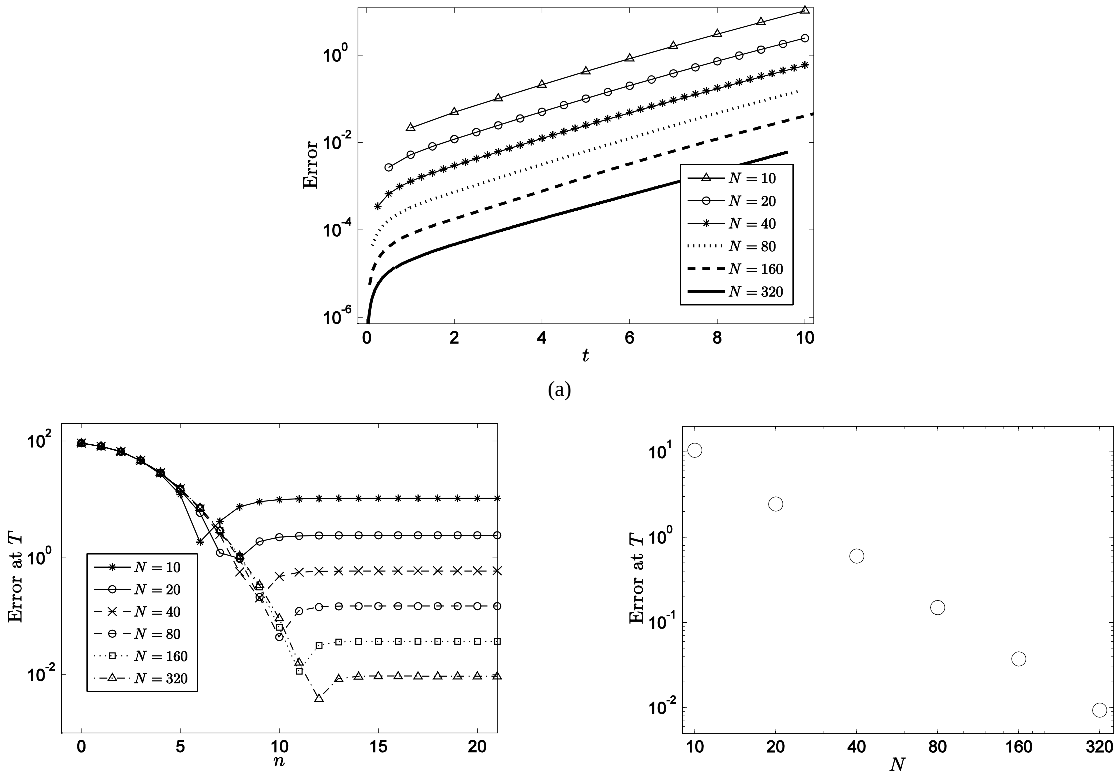
$$u=t-\int_0^t\int_0^s\sin u(\tau)\mathrm{d}\tau\mathrm{d}s=t-\int_0^t(t-s)\sin u(s)\mathrm{d}s.$$

Take $T=6$, $N=20$. Similar behavior of errors as in **Figure 1** can be observed from **Figure 3** which shows



(a) (b)

Figure 1. Example 1 is simulated by numerical scheme (5), (6) with discretization parameter N = 20. (a) The numerical solution φ of the first 5 iterations for integration time $t\in[0,10]$; (b) Dependence of the error $\left|\varphi_n^N(10)-u^*(10)\right|$ on iteration number n.

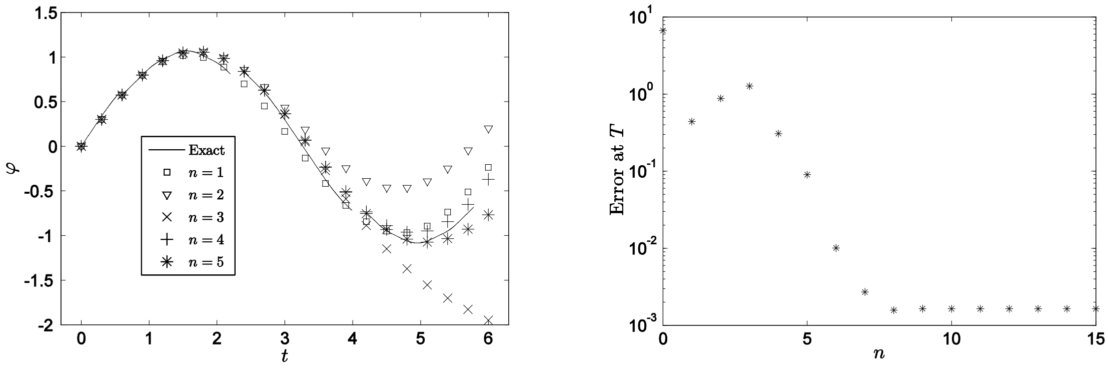


(a)

(b)

(c)

Figure 2. Example 1 is simulated by (5), (6) with various discretization parameter *N*. (a) Dependence of the error $\left|\varphi_n^N(t)-u^*(t)\right|$ on integration time *t*; (b) Dependence of the error $\left|\varphi_n^N(10)-u^*(10)\right|$ on iteration number *n*; (c) Dependence of the error $\left|\varphi_n^N(10)-u^*(10)\right|$ on discretization parameter *N*.



(a)

(b)

Figure 3. Example 2 is simulated by numerical scheme (5), (6) with *N* = 20. (a) The numerical solution φ of the first 5 iterations for integration time $t\in[0,6]$; (b) Dependence of the error $\left|\varphi_n^N(6)-u^*(6)\right|$ on iteration number *n*.

the results of the first 5 iterations and the errors at *T* for each iteration. It confirms the validity of the scheme (5), (6) for equations with general integrand *f*.

What's different from Example 1 is that, at *T*, the results of the second and the third iterations are even worse than the first one. However, it can be noticed that, in the interval closer to *t* = 0, for example $[0,3]$, the errors decrease as *n* increases all the same. So the underlying numerical iteration method can be viewed as a point-by-point correction process.

3. Multistage Scheme

It's well-known that the convergence of the Picard iteration is constrained in some interval. Then how can we get the numerical solution to the integral Equation (1) when t is outside the interval of convergence? We will take advantage of the multistage method and design a scheme by which the considered problem can be solved interval by interval. For example, the Equation (1) is considered on $[t_0, T]$, however, assume that the single-stage-scheme designed in the previous section is convergent only on $[t_0, t_1]$, where $t_1 < T$. For achieving the numerical result at T, we can regard the problem on $[t_1, T]$ as a new one, in which we take the numerical result at t_1 as the initial value. Now we begin to design the multistage scheme in detail.

Denote the time interval considered for (1) by $I = [t_0, T]$. For a given positive integer K, we break I into K disjoint subintervals such that $I = \bigcup_{k=1}^{K} I_k$,

$$I_k = \left(t^{k-1}, t^k\right], \quad h_k = t^k - t^{k-1}, \quad k = 1, \cdots, K,$$

where $t_0 = t^0 < t^1 < \cdots < t^K = T$. For $k = 1, \cdots, K$, take N_k uniformly distributed nodes $\left\{t_i^k\right\}$ on I_k satisfying

$$h_k = \frac{h_k}{N_k}, \quad t_i^k = t^{k-1} + ih_k, \quad 0 \le i \le N_k. \tag{8}$$

Suppose the equation has been solved on $\left[t^0, t^k\right]$, namely, the first k subintervals. For I_κ ($1 \le \kappa \le k$), denote the times of iteration by n_κ and the iterative solutions by $\left\{\tilde{\varphi}_i^\kappa, 0 \le i \le N_\kappa\right\}$, where $\tilde{\varphi}_i^\kappa = \varphi_{n_\kappa}\left(t_i^\kappa\right)$.

Now we consider the solution on I_{k+1}. Taking $t = t^k$ in (1),

$$u\left(t^k\right) = C\left(t^k\right) + \int_{t^0}^{t^k} f\left(t^k, s, u(s)\right) \mathrm{d}s,$$

we have for $t \in I_{k+1}$,

$$\begin{aligned} u(t) &= C(t) + \int_{t^0}^{t} f\left(t, s, u(s)\right) \mathrm{d}s = C(t) + \int_{t^0}^{t^k} f\left(t, s, u(s)\right) \mathrm{d}s + \int_{t^k}^{t} f\left(t, s, u(s)\right) \mathrm{d}s \\ &= C\left(t^k\right) + \int_{t^0}^{t^k} f\left(t^k, s, u(s)\right) \mathrm{d}s + \int_{t^0}^{t^k} \left[f\left(t, s, u(s)\right) - f\left(t^k, s, u(s)\right) \right] \mathrm{d}s + C(t) - C\left(t^k\right) + \int_{t^k}^{t} f\left(t, s, u(s)\right) \mathrm{d}s \\ &= u\left(t^k\right) + \int_{t^0}^{t^k} \left[f\left(t, s, u(s)\right) - f\left(t^k, s, u(s)\right) \right] \mathrm{d}s + C(t) - C\left(t^k\right) + \int_{t^k}^{t} f\left(t, s, u(s)\right) \mathrm{d}s, \end{aligned} \tag{9}$$

the right hand side of which will be analyzed below.

- An approximation $\tilde{\varphi}_{N_k}^k$ of the first term $u\left(t^k\right)$ has been gotten in previous resolution.
- The second part, with the approximations of u on nodes in $\left[t^0, t^k\right]$ having been gained, can also be approximated

$$\begin{aligned} \int_{t^0}^{t^k} \left[f\left(t, s, u(s)\right) - f\left(t^k, s, u(s)\right) \right] \mathrm{d}s &= \sum_{\kappa=1}^{k} \int_{t^{\kappa-1}}^{t^\kappa} \left[f\left(t, s, u(s)\right) - f\left(t^k, s, u(s)\right) \right] \mathrm{d}s \\ &\approx \sum_{\kappa=1}^{k} \sum_{i=0}^{N_\kappa} \omega_i^\kappa \left[f\left(t, t_i^\kappa, u\left(t_i^\kappa\right)\right) - f\left(t^k, t_i^\kappa, u\left(t_i^\kappa\right)\right) \right] \approx \sum_{\kappa=1}^{k} \sum_{i=0}^{N_\kappa} \omega_i^\kappa \left[f\left(t, t_i^\kappa, \tilde{\varphi}_i^\kappa\right) - f\left(t^k, t_i^\kappa, \tilde{\varphi}_i^\kappa\right) \right], \end{aligned}$$

where the corresponding weights for numerical integration on I_κ are

$$\omega_0^\kappa = \omega_{N_\kappa}^\kappa = \frac{1}{2} h_\kappa, \quad \omega_i^\kappa = h_\kappa, \quad 1 \le i \le N_\kappa - 1. \tag{10}$$

- $C(t) - C\left(t^k\right)$ can be calculated directly.

Denoting

$$C_{k+1}(t) = \tilde{\varphi}_{N_k}^k + \sum_{\kappa=1}^{k} \sum_{i=0}^{N_\kappa} \omega_i^\kappa \left[f\left(t, t_i^\kappa, \tilde{\varphi}_i^\kappa\right) - f\left(t^k, t_i^\kappa, \tilde{\varphi}_i^\kappa\right) \right] + C(t) - C\left(t^k\right), \quad t \in I_{k+1}, 0 \le k < K, \tag{11}$$

(9) leads to a new equation, which is similar to the considered problem (1),

$$u(t) \approx \tilde{u}_{k+1}(t) := C_{k+1}(t) + \int_{t^k}^{t} f\left(t, s, u(s)\right) \mathrm{d}s, \quad t \in I_{k+1}, 0 \le k < K,$$

namely,

$$u(t) \approx \tilde{u}_k(t) := C_k(t) + \int_{t^{k-1}}^{t} f(t, s, u(s))\,\mathrm{d}s, \quad t \in I_k, 1 \le k \le K. \tag{12}$$

Using (5), (6) over $\left[t^{k-1}, t^k\right]$, numerical solution to (12) can be obtained.

We conclude the previous analysis as an algorithm.

Algorithm 1 *Choose the algorithm's parameters*: *number of subintervals* K, *set of nodes* $\left\{t^k, 0 \le k \le K\right\}$ *and discretization parameters* $\left\{N_k, 1 \le k \le K\right\}$.

Step 1. For $1 \le k \le K$, generate

- the uniformly distributed nodes and corresponding weights $\left\{\left(t_i^k, \omega_i^k\right), 0 \le i \le N_k\right\}$ on I_k according to (8) and (10)

- the weights $\left\{\omega_{i,j}^k, 0 \le j \le i\right\}$ for numerical integration on $\left[t_0^k, t_i^k\right]$ $\left(1 \le i \le N_k\right)$,

$$\omega_{i,0}^k = \omega_{i,i}^k = \frac{t_i^k - t_0^k}{2i} = \frac{1}{2}\hbar_\kappa, \qquad \omega_{i,j}^k = \frac{t_i^k - t_0^k}{i} = \hbar_\kappa, \quad 1 \le j \le i-1.$$

Step 2. For *k* = 1, solve (12). Note that the first term of $C_1(t): \tilde{\varphi}_{N_0}^0 = u\left(t^0\right) = C\left(t^0\right)$. So solving (12) for *k* = 1 is equivalent to solving the original Equation (1) for $T = t^1$. Use (5), (6) with $\left\{t_i^1, \omega_{i,j}^1\right\}$ instead of $\left\{t_i, \omega_{i,j}\right\}$.

Step 3. Recursively solve (12) for $1 < k \le K$ using a similar scheme to (2) as follows:

- Calculate $C_k\left(t_i^k\right)$ $\left(1 \le i \le N_k\right)$ by (11).

- The initial value of iteration:

$$\varphi_0\left(t_0^k\right) = C_k\left(t_0^k\right) = C_k\left(t^{k-1}\right) = \tilde{\varphi}_{N_{k-1}}^{k-1}, \qquad \varphi_0\left(t_i^k\right) = C_k\left(t_i^k\right), \quad 1 \le i \le N_k.$$

- For $n \ge 0$, $\varphi_{n+1}\left(t_0^k\right) = \tilde{\varphi}_{N_{k-1}}^{k-1}$ and

$$\varphi_{n+1}\left(t_i^k\right) = C_k\left(t_i^k\right) + \sum_{j=0}^{i} \omega_{i,j}^k f\left(t_i^k, t_j^k, \varphi_n\left(t_j^k\right)\right), \quad 1 \le i \le N_k.$$

Here, we perform a numerical test to examine the effectiveness of Algorithm 1 and compare it with the scheme in single interval (2).

Example 3 *Consider the Lane-Emden equation*

$$\begin{cases} u'' + \dfrac{2}{t}u' + u^5 = 0, & 0 < t \le T, \\ u(0) = 1, u'(0) = 0. \end{cases}$$

The exact solution is

$$u(t) = \left(1 + \frac{t^2}{3}\right)^{-\frac{1}{2}}.$$

The equivalent integral form of the Lane–Emden equation is [20]

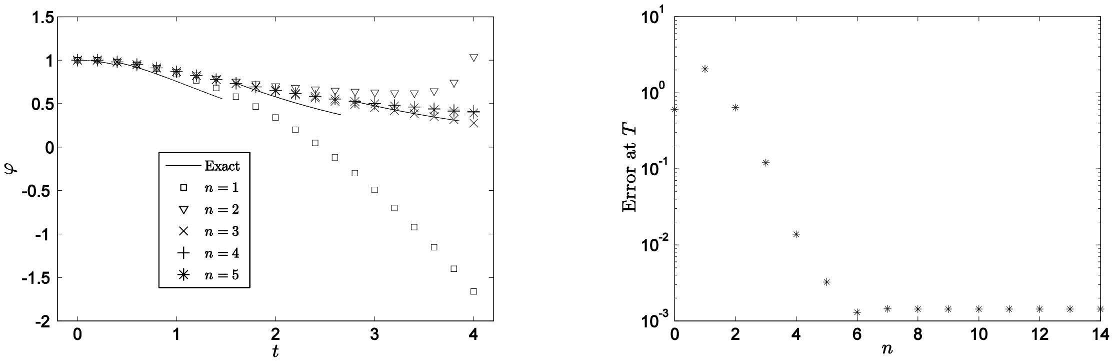
$$u(t) = 1 - \int_0^t s\left(1 - \frac{s}{t}\right)u^5(s)\,\mathrm{d}s.$$

First, taking *T* = 4, $N = 20$, we solve the current problem by (5), (6). The numerical solutions of the first 5 iterations and the errors at *T* are shown in **Figure 4** from which the convergence can be observed. Unfortunately, the scheme is not convergent for *T* = 6.

Consider the underlying problem for larger *T* by Algorithm 1. The time interval $[0, T]$ is uniformly divided into *K* subintervals, in which the same discretization parameter, denoted by *N*, is taken. Take $K = 3, 4, 6, 12$ and $N = 10, 20, 30, 40$. For each *N*, iterations on I_k $(1 \le k \le K$) stop when

$$\left|\varphi_{n+1}^{N,K}\left(t_N^k\right) - \varphi_n^{N,K}\left(t_N^k\right)\right| < 10^{-14}, \tag{13}$$

where $\varphi_n^{N,K}$ denotes the result after *n* iterations when discretization parameters *N* and *K* are taken. Errors and convergence rates respect to *N* at *t* = 12 are reported in **Table 1**, from which one can see that the underlying scheme is of order N^{-2}.



(a) (b)

Figure 4. Example 3 is simulated by numerical scheme (5), (6) with $N = 20$. (a) The numerical solution φ of the first 5 iterations for integration time $t \in [0,4]$; (b) Dependence of the error $\left|\varphi_n^N(4) - u^*(4)\right|$ on iteration number n.

Table 1. The error $\left|\varphi_n^{N,K}(t) - u^*(t)\right|$ and convergence rate at $t = 12$ (Example 3 is simulated by Algorithm 1 with various discretization parameter N and number of subintervals K).

N	$K = 3$		$K = 4$		$K = 6$		$K = 12$	
	Error	Order	Error	Order	Error	Order	Error	Order
10	1.814e−3		9.131e−4		3.771e−4		9.040e−5	
20	3.771e−4	−2.27	2.070e−4	−2.14	9.040e−5	−2.06	2.237e−5	−2.01
30	1.625e−4	−2.08	9.040e−5	−2.04	3.987e−5	−2.02	9.922e−6	−2.01
40	9.040e−5	−2.04	5.054e−5	−2.02	2.237e−5	−2.01	5.578e−6	−2.00

In fact, from the errors reported in the table, the convergence order K^{-2} can also be obtained. So the scheme is of order $(NK)^{-2}$. Errors for $K = 3$ and $N = 10, 20, 30, 40$ are plotted in Figure 5(a). The validity of Algorithm 1 is numerically confirmed.

It's an interesting phenomenon observed from Table 1 that almost the same results are obtained for same NK. For example, when $NK = 120$, the errors are all $9.040\text{e}-5$. This may be because "enough" iteration numbers are taken for all subintervals in the sense of (13). Setting the maximal iteration number allowed for each subinterval to 3 and taking $NK = 120$, we recalculate the current example up to $T = 12$ for $K = 3, 4, 5, 6, 8, 10, 12, 15$. The errors at T are presented in Figure 5(b) which shows the decrement of the errors respect to K.

4. Problem with Singular Integrand

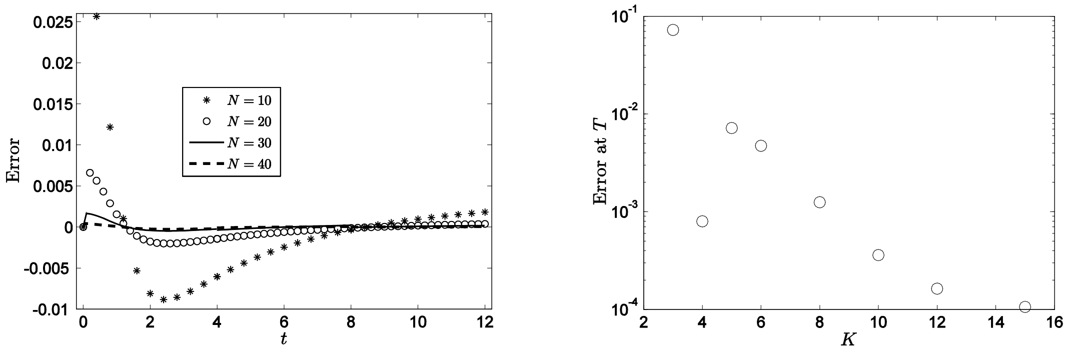
In recent years, the fractional differential or integral equations are much involved. In fact, fractional integral is a class of integration with weak singular kernel. So many fractional differential and integral equations can be equivalently expressed by the singular Volterra integral equation of the second kind. Let us consider such an integral equation with some singularity.

Example 4 *Consider the singular Volterra integral equation* [14]

$$u(t) = \sqrt{t} + \frac{3}{8}\pi t^2 - \int_0^t \frac{u^3(s)}{\sqrt{t-s}}\,\mathrm{d}s.$$

The exact solution is $u^*(t) = \sqrt{t}$. Note that in the integrand there has $\dfrac{1}{\sqrt{t-s}}$, which is infinity at $s = t$. In such case, the numerical scheme (5), (6) and corresponding multistage scheme (Algorithm 1) are not valid any more.

A simple idea is to avoid the value of the integrand at $s = t$ in the numerical integration, so an alternative is to



(a) (b)

Figure 5. Example 3 is simulated by Algorithm 1. (a) Dependence of the error $\varphi_n^{N,K}(t) - u^*(t)$ on integration time t with the number of subintervals K = 3; (b) Dependence of the error $\left|\varphi_n^{N,K}(12) - u^*(12)\right|$ on the number of subintervals K with NK = 120 and $\max(n) = 3$.

integrate with compound rectangular formula. The only things we need to do are changing the nodes of numerical integration and generating approximations for the values of φ on these points since only the values on the nodes $\{t_i^k\}$ have been gained.

For $1 \le k \le K$, denote the midpoint of $\left[t_{i-1}^k, t_i^k\right]$ ($1 \le i \le N_k$) and the corresponding weight by

$$\hat{t}_i^k = \frac{1}{2}\left(t_{i-1}^k + t_i^k\right) = t^{k-1} + \left(i - \frac{1}{2}\right)h_k, \quad \hat{\omega}_i^k = h_k. \tag{14}$$

Denote

$$\hat{C}_k(t) = \tilde{\varphi}_{N_{k-1}}^{k-1} + \sum_{\kappa=1}^{k-1}\sum_{i=1}^{N_\kappa}\hat{\omega}_i^\kappa\left[f\left(t, \hat{t}_i^\kappa, \hat{\varphi}_i^\kappa\right) - f\left(t^{k-1}, \hat{t}_i^\kappa, \hat{\varphi}_i^\kappa\right)\right] + C(t) - C\left(t^{k-1}\right), \quad t \in I_k, 1 \le k \le K, \tag{15}$$

in which

$$\hat{\varphi}_i^\kappa = \frac{1}{2}\left(\tilde{\varphi}_{i-1}^\kappa + \tilde{\varphi}_i^\kappa\right).$$

Thus, (12) becomes

$$u(t) \approx \hat{u}_k(t) := \hat{C}_k(t) + \int_{t^{k-1}}^t f\left(t, s, u(s)\right)\mathrm{d}s, \quad t \in I_k, 1 \le k \le K. \tag{16}$$

We present the following algorithm.

Algorithm 2 Choose the algorithm's parameters: number of subintervals K, set of nodes $\left\{t^k, 0 \le k \le K\right\}$ and discretization parameters $\left\{N_k, 1 \le k \le K\right\}$.

Step 1. For $1 \le k \le K$, generate

- the nodes $\{t_i^k\}$ on I_k according to (8).
- the integral nodes and weights $\left\{\left(\hat{t}_i^k, \hat{\omega}_i^k\right), 1 \le i \le N_k\right\}$ on I_k according to (14).
- the weights $\left\{\hat{\omega}_{i,j}^k, 1 \le j \le i\right\}$ for numerical integration on $\left[t_0^k, t_i^k\right]$ $\left(1 \le i \le N_k\right)$,

$$\hat{\omega}_{i,j}^k = h_\kappa, \quad 1 \le j \le i.$$

Step 2. Solve (16) for k = 1. As in Algorithm 1, since $\hat{C}_1(t) = C(t)$, it is equivalent to solving (1) for $T = t^1$. Detail algorithm reads:

- For $0 \le i \le N_1$, calculate $C\left(t_i^1\right)$ and get the initial value of iteration: $\varphi_0\left(t_i^1\right) = C\left(t_i^1\right)$.
- For $n \ge 0$, $\varphi_{n+1}\left(t_0^1\right) = C\left(t^0\right)$ and

$$\varphi_{n+1}\left(t_i^1\right) = C\left(t_i^1\right) + \sum_{j=1}^{i}\hat{\omega}_{i,j}^1 f\left(t_i^1, \hat{t}_j^1, \hat{\psi}_j^1\right), \quad 1 \le i \le N_1,$$

where $\hat{\psi}_j^1 = \frac{1}{2}\left(\varphi_n\left(t_{j-1}^1\right) + \varphi_n\left(t_j^1\right)\right)$.

Step 3. Recursively solve (16) for $1 < k \le K$ as follows:

- For $1 \le i \le N_k$, calculate $\hat{C}_k\left(t_i^k\right)$ by (15) and get the initial value of iteration:

$$\varphi_0\left(t_0^k\right) = \tilde{\varphi}_{N_{k-1}}^{k-1}, \qquad \varphi_0\left(t_i^k\right) = \hat{C}_k\left(t_i^k\right), \quad 1 \le i \le N_k.$$

- For $n \ge 0$, $\varphi_{n+1}\left(t_0^k\right) = \tilde{\varphi}_{N_{k-1}}^{k-1}$ and

$$\varphi_{n+1}\left(t_i^k\right) = \hat{C}_k\left(t_i^k\right) + \sum_{j=1}^{i} \hat{\omega}_{i,j}^k f\left(t_i^k, \hat{t}_j^k, \hat{\psi}_j^k\right), \quad 1 \le i \le N_k,$$

where $\hat{\psi}_j^k = \frac{1}{2}\left(\varphi_n\left(t_{j-1}^k\right) + \varphi_n\left(t_j^k\right)\right)$.

Now, we come back to Example 4. Taking $t = 0.5, 0.8$ to subdivide the time interval $[0,1]$ and N = 5, 10, 20, 40. **Figure 6** presents the dependence of the error on t for each N and that on N at t = 1. The results verify the validity of Algorithm 2 in solving problems with some singularity at the limits of integration. However, the method is of order about only $N^{-0.5}$ for this example.

Remark 1. Algorithm 2 is devised not especially for singular problems. It's also valid for regular problems. For instance, we recalculate Example 1 with K = 2 and N = 5, 10, 20, 40, 80, 160. Errors and convergence rates respect to N are reported in **Table 2**, from which we can find the order is N^{-2}.

5. Conclusions

In this work, Picard iteration methods with numerical integration are devised for the second kind nonlinear Volterra integral equations. The Picard iteration method solves the considered nonlinear equation explicitly, while the multistage scheme solves it interval by interval and enlarges the convergence region of the Picard iteration method. Numerical results validate the proposed schemes and algorithms and reveal that the schemes are of order $(NK)^{-2}$ for regular problems.

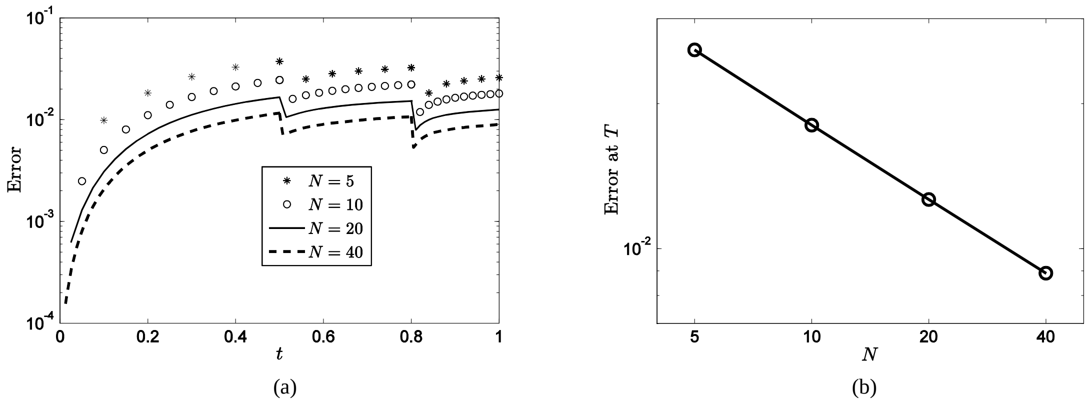


Figure 6. Example 4 is simulated by Algorithm 2. (a) Dependence of the error $\varphi_n^{N,K}(t) - u^*(t)$ on integration time t; (b) Dependence of the error $\left|\varphi_n^{N,K}(1) - u^*(1)\right|$ on discretization parameter N.

Table 2. The error $\left|\varphi_n^{N,K}(t) - u^*(t)\right|$ and convergence rate at t = 10 (Example 1 is simulated by Algorithm 2 with number of subintervals K = 2 and various discretization parameter N).

N	5	10	20	40	80	160
Error	5.807e−0	1.376e−0	3.394e−1	8.459e−2	2.113e−2	5.281e−3
Order		−2.08	−2.02	−2.00	−2.00	−2.00

What should be noticed is that the errors reported in the numerical results decrease exponentially respect to times of iteration n (for example, through simple calculation, we can observe from **Figure 3(b)** and **Figure 4(b)** that the convergence rates are about 4^{-n} for Examples 2 and 3) and are of order -2 respect to discretization parameter NK. Future work may concern on enhancing the rate of convergence respect to NK.

Acknowledgements

This work was supported by the Natural Science Foundation of Shanghai (No. 14ZR1440800) and the Innovation Program of the Shanghai Municipal Education Commission (No. 14ZZ161).

References

[1] Davis, H.T. (1962) Introduction to Nonlinear Differential and Integral Equations. Dover, Publications, New York.

[2] Jerri, A. (1999) Introduction to Integral Equations with Applications. Wiley, New York.

[3] Linz, P. (1985) Analytical and Numerical Methods for Volterra Equations. SIAM, Philadelphia. http://dx.doi.org/10.1137/1.9781611970852

[4] Miller, R.K. (1967) Nonlinear Volterra Integral Equations. W. A. Benjamin, Menlo Park.

[5] Wazwaz, A.M. (1997) A First Course in Integral Equations. World Scientific, Singapore City. http://dx.doi.org/10.1142/3444

[6] Wazwaz, A.M. (2009) Partial Differential Equations and Solitary Waves Theory. Higher Education, Beijing, and Springer, Berlin. http://dx.doi.org/10.1007/978-3-642-00251-9

[7] Wazwaz, A.M. (2011) Linear and Nonlinear Integral Equations: Methods and Applications. Higher Education, Beijing, and Springer, Berlin. http://dx.doi.org/10.1007/978-3-642-21449-3

[8] Duan, J.S. and Rach, R. (2011) A New Modification of the Adomian Decomposition Method for Solving Boundary Value Problems for Higher Order Nonlinear Differential Equations. *Applied Mathematics and Computation*, **218**, 4090-4118. http://dx.doi.org/10.1016/j.amc.2011.09.037

[9] Daftardar-Gejji, V. and Jafari, H. (2006) An Iterative Method for Solving Nonlinear Functional Equations. *Journal of Mathematical Analysis and Application*, **316**, 753-763. http://dx.doi.org/10.1016/j.jmaa.2005.05.009

[10] Maleknejad, K. and Najafi, E. (2011) Numerical Solution of Nonlinear Volterra Integral Equations Using the Idea of Quasilinearization. *Communications in Nonlinear Science and Numerical Simulation*, **16**, 93-100. http://dx.doi.org/10.1016/j.cnsns.2010.04.002

[11] Brunner, H. (2004) Collocation Methods for Volterra Integral and Related Functional Equations. Cambridge University Press, Cambridge. http://dx.doi.org/10.1017/CBO9780511543234

[12] Brunner, H., Pedas, A. and Vainikko, G. (2001) A Spline Collocation Method for Linear Volterra Integro-Differential Equations with Weakly Singular Kernels. *BIT Numerical Mathematics*, **41**, 891-900. http://dx.doi.org/10.1023/A:1021920724315

[13] Brunner, H., Pedas, A. and Vainikko, G. (2001) Piecewise Polynomial Collocation Method for Linear Volterra Integro-Differential Equations with Weakly Singular Kernels. *SIAM Journal on Numerical Analysis*, **39**, 957-982. http://dx.doi.org/10.1137/S0036142900376560

[14] Te Riele, H.J.J. (1982) Collocation Methods for Weakly Singular Second-Kind Volterra Integral Equations with Non-Smooth Solution. *IMA Journal of Numerical Analysis*, **2**, 437-449. http://dx.doi.org/10.1093/imanum/2.4.437

[15] Atkinson, K.E., Han, W. and Stewart, D. (2009) Numerical Solution of Ordinary Differential Equations. John Wiley & Sons, Hoboken. http://dx.doi.org/10.1002/9781118164495

[16] Fazeli, S., Hojjati, G. and Shahmorad, S. (2012) Super Implicit Multistep Collocation Methods for Nonlinear Volterra Integral Equations. *Mathematical and Computer Modelling*, **55**, 590-607. http://dx.doi.org/10.1016/j.mcm.2011.08.034

[17] Ketabchi, R., Mokhtari, R. and Babolian, E. (2015) Some Error Estimates for Solving Volterra Integral Equations by Using the Reproducing Kernel Method. *Journal of Computational and Applied Mathematics*, **273**, 245-250. http://dx.doi.org/10.1016/j.cam.2014.06.016

[18] Saberi-Nadjafi, J., Mehrabinezhad, M. and Akbari, H. (2012) Solving Volterra Integral Equations of the Second Kind by Wavelet-Galerkin Scheme. *Computers & Mathematics with Applications*, **63**, 1536-1547. http://dx.doi.org/10.1016/j.camwa.2012.03.043

[19] Yousefi, S.A. (2006) Numerical Solution of Abel's Integral Equation by Using Legendre Wavelets. *Applied Mathematics and Computation*, **175**, 574-580. http://dx.doi.org/10.1016/j.amc.2005.07.032

[20] Wazwaz, A.M., Rach, R. and Duan J.S. (2013) Adomian Decomposition Method for Solving the Volterra Integral

Form of the Lane-Emden Equations with Initial Values and Boundary Conditions. *Applied Mathematics and Computation*, **219**, 5004-5019. http://dx.doi.org/10.1016/j.amc.2012.11.012

[21] Costarelli, D. and Spigler, R. (2013) Solving Volterra Integral Equations of the 2nd Kind by Sigmoidal Functions Approximations. *Journal of Integral Equations and Applications*, **25**, 193-222. http://dx.doi.org/10.1216/JIE-2013-25-2-193

[22] Costarelli, D. and Spigler, R. (2014) A Collocation Method for Solving Nonlinear Volterra Integro-Differential Equations of the Neutral Type by Sigmoidal Functions. *Journal of Integral Equations and Applications*, **26**, 15-52. http://dx.doi.org/10.1216/JIE-2014-26-1-15

[23] Maleknejad, K., Hashemizadeh, E. and Ezzati, R. (2011) A New Approach to the Numerical Solution of Volterra Integral Equations by Using Bernstein's Approximation. *Communications in Nonlinear Science and Numerical Simulation*, **16**, 647-655. http://dx.doi.org/10.1016/j.cnsns.2010.05.006

[24] Mathews, J. (1989) Symbolic Computational Algebra Applied to Picard Iteration. *Mathematics and Computer Education*, **23**, 117-122.

[25] Parker, G.E. and Sochacki, J.S. (1996) Implementing the Picard Iteration. *Neural*, *Parallel and Scientific Computations*, **4**, 97-112.

[26] Bailey, P.B., Shampine, L.F. and Waltman, P.E. (1968) Nonlinear Two Point Boundary Value Problems. Academic, New York/London.