ISSN: 0067-2904

Numerical Treatment of First Order Volterra Integro-Differential Equation Using Non-Polynomial Spline Functions

Rawaa I. Esa*[1], Atefa J. Saleh[2]
[1]Department of Computer Science, College of Basic Education al Mustansiriya University, Baghdad, Iraq
[2]Department of Mathematics, College of Basic Education al Mustansiriya University, Baghdad, Iraq

Received: 26/11/ 2019 Accepted: 15/ 3/2020

Abstract
The approach given in this paper leads to numerical methods to find the approximate solution of volterra integro –diff. equ.1st kind. First, we reduce it from integro VIDEs to integral VIEs of the 2nd kind by using the reducing theory, then we use two types of Non-polynomial spline function (linear, and quadratic). Finally, programs for each method are written in MATLAB language and a comparison between these two types of Non-polynomial spline function is made depending on the least square errors and running time. Some test examples and the exact solution are also given.

Keywords: General 1st order Volterra integro-differential equation, Linear non-polynomial spline , and quadratic non-polynomial spline function.

المعالجة العددية لمعادلة فولتيرا التكاملية التفاضلية من الدرجة الأولى باستخدام دوال الثلمة

رواء ابراهيم عيسى [1]*،عاطفة جليل صالح[2]
[1]قسم الحاسبات، كلية التربية الاساسيه، الجامعه المستنصرية، بغداد، العراق
[2]قسم الرياضيات، كليه التربيه الاساسيه، الجامعه المستنصريه، بغداد، العراق

الخلاصة
في هذا البحث قدم طرق عددية لإيجاد الحل التقريبي لـ volterra integro –diff. equ.1st kind.
أولاً ، قمنا بتحويل المعادلة من VIDEs integro إلى VIEs متكاملة من النوع الثاني عن طريق استخدام نظرية تخفيض الرتبة ثم معالجتها باستخدم نوعين من معادلات المتعددة غير الحدودية (الخطية والتربيعية). أخيرًا ، تم كتابة برامج لكل طريقة في لغة MATLAB وإجراء مقارنة بين هذين النوعين من وظائف شريحة غير متعدّدة الحدود اعتمادًا على مربع مجموع الأخطاء وزمن تنفيذ البرنامج. تم استعراض النتائج بحل بعض االأمثلة الاختبارية ومقارنتها بالحل الدقيق.

1. Introduction

Over the last decades, many authors have interested in finding the numerical solutions of differential equations and integro –differential equations using varies numerical schemes such as finite difference and Runge-Kutta methods, see for instance [1,2].

In recent years, there has been a growing interest in the volterra integro –differential equations (VIDEs) in various fields of physics and engineering along with their applied sciences such as viscoelasticity, thermodynamics, heat conditions in materials, and wave propagation [3]. Many researchers used polynomial and non-polynomial spline functions to solve VIDEs of the second kind

*Email: rawaaa_e.edbs@uomustansiriyah.edu.iq

[4,5,6]. In this paper, we used two types of non-polynomial spline function (linear and quadratic) to find the approximate solution of VIDEs of the first kind, which takes the following form:

$$\frac{d}{dt}y(t)+U_0(t)y(t)=g(t)+\int_a^t k(t,s)y(s)ds;t\in \mathrm{I}=[a,b]........(1)$$

where a, b(t) are the limits of integration , K(t,s) is a function of two variables, t and s, called Kernel, while the function y(t) that will be determined appears inside and outside the integral sign. The functions k(t,s) ,g(t) and U$_0$ (t) are known [7].

1- Integro-Differential Equation [8]

An integro-differential equation is an equation that involves one (or more) unknown functions, together with both differential and integral operations on t. A linear integro-differential equation of order n has the form:

$$y^{(n)}(t)+\sum_{i=0}^{n-1}U_i(t)y^{(i)}(t)=g(t)+\lambda\int_a^{b(t)}K(t,s)y(s)ds.......\,.......(2)$$

$$where.......\,......y^{(i)}(t)=\frac{d^i y}{dt^i}$$

, $K(t,s),g(t),U_i(t)(i=0,1,2,...n-1)$ are known functions, $y(t)$ is the unknown function, and λ is a scalar parameter.

2- Reduction to Integral Equation

The reduction of IDEs to integral equation can be used for the analysis of variety of VIDEs.

Reduction theorem [9]

Let g, k be iterated L$_2$ integrable functions on interval [a,b] and U_i ϵ C^n[a,b], then equation (2) can be written as follows:

$$[D^n+\sum_{i=1}^{n-1}U_i(t)D^i]y(t)=g(t)+\int_a^t K(t,s)y(s)ds, \qquad t\in[a,b]$$

With the initial conditions $y(a)=y_0, y'(a)=y_1,......, y^{(n)}(a)=y_n$

which can be reduced to linear VIE in the form:

$$y(t)=G_n(t)+\int_a^t K_n(t,s)y(s)ds.........\,..........\,....(3)$$

where

$$G_n(t)=\sum_{i=0}^{n-1}\frac{y_i t^i}{i!}+\frac{1}{(n-1)!}\int_a^t(t-s)^{n-1}g(s)ds+\sum_{k=0}^{n-2}\sum_{j=k+1}^{n-1}\sum_{i=n+k-j}^{n-1}(-1)^k U_j^{(k)}(a)A_{k,n-i-1}\frac{y_{i+j-n-k}t^i}{i!}$$

and

$$K_n(t,s)=\frac{1}{(n-1)!}\int_t^x(t-z_1)^{n-1}k(z_1,s)dz_1-\sum_{k=0}^{n-1}\sum_{j=0}^{k}(-1)^j U_{j+n-k-1}^{(k)}(s)B_{n-k-1,j}\frac{(t-s)^k}{k!}$$

[A]and [B] are two special constant matrices of dimensions n-2×n-2 and n-1×n-1, respectively.

3- Non-Polynomial Spline Function

3-1 General Definition of Spline Function

In mathematics, a spline is a special function defined piecewise by polynomials .In computer sciences, the term spline refers to a piecewise polynomial curve.

The solution was to place a metal weight (called Knots) at the control points, and bend a thin metal or wooden beam (called a Spline) through the weights [10].

A piecewise polynomial $f(t)$ is obtained by dividing t into contiguous intervals, and representing $f(t)$ by a separate polynomial in each interval.

The polynomials are joined together at the interval endpoints (Knots) in such a way that a certain degree of smoothness of the resulting function is guaranteed.

A Spline S is called a Spline of order n if:

1- The domain of S is in the interval [a, b].

2- $S \in C^{k-1}[a,b]$

3- $S, S', S'', S'''....., S^{n-1}$, are all continuous functions on [a, b].

4- There are t_i (the knots of S)s.t $a = t_0 < t_1 < t_2 < ... < t_n = b$ and such that S is polynomial of a degree at most n on each subinterval $[t_i, t_{i+1}]$.[11]

3-2 Non-Polynomial Function

Consider the partition $\Delta = \{t_0, t_1, t_2, ..., t_n\}$ of $[a,b] \subset$ R, where $\Delta : a = t_0 < t_1 < t_2 < ... < t_n = b$. Let $S_\Delta(t)$ denotes the set of piecewise polynomials on subinterval $I_i = [t_i, t_{i+1}]$ of partition Δ. Let P_i (t) be the segment of non-polynomial spline function that has the form:

$$P_i(t) = a_i \cos m(t-t_i) + b_i \sin m(t-t_i) + ... + y_i(t-t_i) + z_i, i = 0,..., n, ...(4)$$

where $a_i, b_i, ..., y_{i\,i}, z_i$ are constants to be determined and m is the frequency of the trigonometric functions which will be used to raise the accuracy of the method [12].

3-3 Linear Non-polynomial Function [13]

The form of Linear non-polynomial spline function is:

$$P_i(t) = a_i \cos m(t-t_i) + b_i \sin m(t-t_i) + c_i(t-t_i) + d_i, i = 0,....n, ...(5)$$

In order to obtain the values of a_i, b_i, c_i, d_i , we differentiate equation (5) three times with respect to t, then we get:

$$\left.\begin{aligned} P_i'(t) &= -ma_i \sin m(t-t_i) + mb_i \cos(t-t_i) + c_i \\ P_i''(t) &= -m^2 a_i \cos m(t-t_i) - m^2 b_i \sin(t-t_i) \\ P_i'''(t) &= m^3 a_i \sin m(t-t_i) - m^3 b_i \cos m(t-t_i) \\ &\text{Now, replacing t by } t_i \text{ in equ(1) and equ(2) yields} \end{aligned}\right\} \quad ....(6)$$

$$a_i = -\frac{1}{m^2} P_i''(t_i) ....(7)$$

$$b_i = -\frac{1}{m^3} P_i'''(t_i) ....(8)$$

$$c_i = P_i'(t_i) - mb_i ....(9)$$

$$d_i = P_i(t_i) - a_{i.} ....(10), \; for i = 0,1,..., n$$

3-4 Quadratic Non-Polynomial Function [14]

The form of the Quadratic non-polynomial spline function is:

$$Q_i(t) = a_i \cos m(t-t_i) + b_i \sin m(t-t_i) + c_i(t-t_i) + d_i(t-t_i)^2 + e_i, ...(11)$$

where $a_i, b_i, c_i, d_i,$ and e_i are constants that we can determine by differentiate equation (11) four times with respect to t, then we get the following equations:

$$\left.\begin{aligned} Q_i'(t) &= -ma_i \sin m(t-t_i) + mb_i \cos m(t-t_i) + c_i + 2d_i(t-t_i) \\ Q_i''(t) &= -m^2 a_i \cos m(t-t_i) - m^2 \sin m(t-t_i) + 2d_i \backslash \\ Q_i'''(t) &= m^3 a_i \sin m(t-t_i) - m^3 \cos m(t-t_i) \\ Q_i^{(4)}(t) &= m^4 a_i \cos m(t-t_i) + m^4 b_i \sin m(t-t_i) \end{aligned}\right\} \quad \ldots\ldots\ldots\ldots..(12)$$

Replacing t by t_i in equations (11) and (12) yields:

$Q_I(t_i) = a_i + e_i$

$Q_i'(t_i) = mb_i + c_i$

$Q_i''(t_i) = -m^2 a_i + 2d_i$

$Q_i'''(t_i) = -m^3 b_i$

$Q_i^{(4)} = m^4 a_i$

From the above relations we obtain that:

$$a_i = \frac{1}{m^4} Q_i^{(4)}(t_i) \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots(13)$$

$$b_i = -\frac{1}{m^3} Q_i'''(t_i) \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots(14)$$

$$c_i = Q_i'(t_i) - mb_i \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots(15)$$

$$d_i = 1/2[Q_i''(t_i) + m^2 a_i] \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots(16)$$

$$e_i = Q_i(t_i) - a_i \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots(17)$$

For i=0,1,2,…,n

4.1 Solution of Linear VIEs of the 2nd kind

Now, we shall use linear and quadratic non-polynomial spline functions to find the numerical approximation of VIEs in eq.3, by differentiating it four times with respect to t, using Libenze formula:

$$y(t) = g_n(t) + \int_a^t K_n(t,s) y(s) ds \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots(18)$$

$$y'(t)=g'(t)+\int_a^t \frac{\partial k(t,s)}{\partial t}y(s)ds+k(t,t)y(t)\ldots\ldots\ldots\ldots(19)$$

$$y''(t)=g''(t)+\int_a^t \frac{\partial^2 k(t,s)}{\partial t^2}y(s)ds+\left(\frac{\partial k(t,s)}{\partial t}\right)_{s=t}y(t)+\frac{dk(t,t)}{dt}y(t)+k(t,t)y'(t)\ldots\ldots(20)$$

$$y'''(t)=g'''(t)+\int_a^t \frac{\partial^3 k(t,s)}{\partial t^3}u(s)ds+\left(\frac{\partial^2 k(t,s)}{\partial t^2}\right)_{s=t}y(t)+\frac{d}{dt}\left(\frac{\partial k(t,s)}{\partial t}\right)_{s=t}y(t)+\left(\frac{\partial k(t,s)}{\partial t}\right)_{s=t}y'(t)+$$
$$\frac{d^2k(t,t)}{dt^2}y(t)+2\frac{dk(t,t)}{dt}y'(t)+k(t,t)y''\ldots\ldots\ldots\ldots(21)$$

$$y^{(4)}(t)=g^{(4)}(t)+\int_a^t \frac{\partial^4 k(t,s)}{\partial t^4}y(t)+\left(\frac{\partial^3 k(t,s)}{\partial t^3}\right)_{s=t}y(t)+\frac{d}{dt}\left(\frac{\partial^2 k(t,s)}{\partial t}\right)_{s=t}y(t)+\left(\frac{\partial^2 k(t,s)}{\partial t^2}\right)y'(t)+$$
$$\frac{d^2}{dt^2}\left(\frac{\partial k(t,s)}{\partial t}\right)_{s=t}y(t)+\frac{d}{dt}\left(\frac{\partial k(t,s)}{\partial t}\right)_{s=t}y'(t)+\frac{d}{dt}\left(\frac{\partial k(t,s)}{\partial t}\right)_{s=t}y''(t)+\frac{d^3k(t,s)}{dt^3}y(t)+3\frac{d^2k(t,s)}{dt^2}y'(t)$$
$$+3\frac{dk(t,s)}{dt}y''(t)+k(t,t)y'''(t)\ldots\ldots\ldots\ldots(22)$$

Now, by substituting $t=a$ in equations 18-22, we get:

$$y_0=y(a)=g(a)\ldots\ldots\ldots\ldots(23)$$

$$y_0'=y'(a)=g'(a)+k(a,a)y(a)\ldots\ldots\ldots\ldots(24)$$

$$y_0''=y''(a)=g''(a)+\left(\left(\frac{\partial k(t,s)}{\partial t}\right)_{s=t}\right)_{t=a}y(a)+\left(\frac{\partial k(t,s)}{\partial t}\right)_{t=a}\frac{dk(t,t)}{dt}y(a)+k(a,a)y'(a)\ldots\ldots(25)$$

$$y_0'''=y'''(a)=g'''(a)+\left(\left(\frac{\partial^2 k(t,s)}{\partial t^2}\right)_{s=t}\right)_{t=a}y(a)+\left[\frac{d}{dt}\left[\frac{\partial k(t,s)}{\partial t}\right]_{s=t}\right]_{t=a}y(a)+\left[\left[\frac{\partial k(t,s)}{\partial t}\right]_{s=tt}\right]_{t=a}y'(a)$$
$$+\left(\frac{d^2k(t,t)}{dt^2}\right)_{t=a}y(a)+2\left(\frac{dk(t,t)}{dt}\right)_{t=a}y'(a)+k(a,a)y''(a)\ldots\ldots\ldots\ldots(26)$$

$$y_0^{(4)}=y^{(4)}(a)=g^{(4)}(a)+\left[\left[\frac{\partial^3 k(t,s)}{\partial t^3}\right]_{s=t}\right]_{t=a}y(a)+\left[\frac{d}{dt}\left(\frac{\partial^2 k(t,s)}{\partial t^2}\right)_{s=t}\right]_{t=a}y(a)+\left(\frac{\partial^2 k(t,s)}{\partial t^2}\right)_{s=t}y'(a)$$
$$\left[\frac{d^2}{dt^2}\left[\frac{\partial k(t,s)}{\partial t}\right]_{s=t}\right]_{t=a}y(a)+2\left[\frac{d}{dt}\left[\frac{\partial k(t,s)}{\partial t}\right]_{s=t}\right]_{t=a}y'(a)+\left(\left[\frac{\partial k(t,s)}{\partial t}\right]_{s=t}\right)_{t=a}y''(a)+\left(\frac{d^3k(t,t)}{dt^3}\right)_{t=a}y(a)$$
$$+3\left(\frac{d^2k(t,t)}{dt^2}\right)_{t=a}y'(a)+3\left(\frac{dk(t,t)}{dt}\right)_{t=a}y''(a)+k(a,a)y'''(a)\ldots\ldots\ldots\ldots(27)$$

5- The Algorithm

5-1 Algorithms of Linear Non-Polynomial

Step 1 : set $h=\frac{b-a}{n}, t_i=t_0+ih, i=0,1,2...,n$ where ($t_0=a, t_n=b$) and $y_0=g(a)$

Step2: Evaluate a_0, b_0, c_0 & d_0 by substituting equations (23)-(26) in equations (7)-(10).

Step3 : Calculate $P_0(t)$ using step 2 and equation (5)

Step4: Approximate $y_1=P_0(t_1)$

Step5: For i-1 to n-1, do the following steps:

Step6: Evaluate $a_i, b_i, c_i and, d_i$ by using equations (7)-(10) and replacing $y(t_i), y'(t_i), y''(t_i), y'''(t_i),$ by $P_i(t_i), P_i'(t_i), P_i''(t_i), P_i'''(t_i)$

Step7: Calculate $P_i(t)$ using step 6 and equation (5)

Step 8: Approximate $y_{i+1} = P_i(t_{i+1})$

5-2 Algorithms of Quadratic Non-Polynomial

Step1: Set $h = \frac{b-a}{n}, t_i = t_0 + ih, i = 0,1,2,..., n$ where ($t_0 = a, t_n = b$) and $y_0 = g(a)$

Step2: Evaluate a_0, b_0, c_0, d_0, and e_0 by substituting equations (23)-(27) in equations (13)-(17).

Step3: Calculate $P_0(t)$ using step 2 and equation (11).

Step4: Approximate $y_1 = P_0(t0$

Step5: For i=1 to n-1 do the following steps:

Step6: Evaluate a_i, b_i, c_i, d_i and e_i by using equations (13)-(17) and replacing $y(t_i), y'(t_i), y''(t_i), y'''(t_i)$,and $y^{(4)}(t_i)$ by $P_i(t_i), P_i'(t_i), P_i''(t_i), P_i'''(t_i)$ and $P^{(4)}(t_i)$

Step7: Calculate $P_i(t)$ using step 6 and equation (11)

Step8: Approxi0mate $y_{i+1} = P_i(t_{i+1})$

6- Numerical Examples

Example 1

Consider the following non-linear VIDEs of 1st kind problem

$$y'(t) = 1 - \int_0^t y(s)ds$$

where $\begin{matrix} g(t) = 1, \\ k(t,s) = 1 \end{matrix}$

After reducing it to VIE of 2nd kind we get**:**

$$y(t) = t - \int_0^t (t-s)y(s)ds : 0 \le t \le 1$$

Where $\begin{matrix} g(t) = t, \\ k(t,s) = t - s \end{matrix}$

And the exact solution is $y(t) = \sin t$**.**

In this problem, we obtain the comparative results between the approximate solution $P_i(t)$ (Linear & Quadratic Non-polynomial spline functions) and $y(t)$ .The exact solution is shown in Table-1 at t=t_i =ih ,h=0.1,i=0,1,2,..,10.

Table-1

t	Exact	$P_i(t)$	
		Linear	Quadratic
0.0	0.0000000	0.0000000	0.00000000
0.1	0.099833416	0.0998333	0.09983341
0.2	0.198669330	0.19866916	0.198669330
0.3	0.2955520206	0.2955520203	0.295520205
0.4	0.389418342	0.389418026	0.389418342
0.5	0.479425538	0.479425158	0.479425527
0.6	0.564642473	0.5646424059	0.564642470

0.7	0.644217687	0.644217679	0.644217685
0.8	0.717356090	0.717355588	0.717356085
0.9	0.783326909	0.783326858	0.783326895
1.0	0.841470984	0.841470882	0.841470952
L.S.E		1.7124E-06	3.18886E-05
R.T		0:1:45	0:1:75

Example 2

Consider the following non-linear VIDEs of 1st kind problem

$$y'(t) = te^{2t} + \int_0^t e^{t+s} y(s)ds,$$

Where $g(t) = te^{2t}$, $k(t,s) = e^{t+s}$

After reducing the VIE of 2nd kind we get

$$y(t) = 2t + 1 - \int_0^t e^{-t+s} y(s)ds : 0 \le t \le 1$$

where $g(t) = 2t + 1$, $k(t,s) = e^{-t+s}$

With the exact solution $y(t) = 1 + t$

In this problem, the value h=0.1 will be compared at the meish point 0(h)1 as shown in Table-2 for y(t) at t=t_i =ih , i=0,1,2,..,10**.**

Table 2

t	Exact	$P_i(t)$	
		Linear	Quadratic
0.000000	1	1	1
0.100000	1.1	1.1095125	1.108321
0.200000	1.2	1.207476	1.207404
0.300000	1.3	1.306705	1.306600
0.400000	1.4	1.409803	1.409950
0.500000	1.5	1.495576	1.505490
0.600000	1.6	1.598773	1.605052
0.700000	1.7	1.704533	1.703985
0.800000	1.8	1.803300	1.802478
0.900000	1.9	1.895899	1.904118
1.000000	2	1.998584	2.003888
L.S.E		0.052497	0.057286
R.T		1:25	1:45

7-Conclustion

In this paper, we first reduced VIDEs of the 1st kind to VIEs of the 2nd kind, and then we constructed linear and quadratic non-polynomial spline functions to find the approximate solution of y (t).

A comparison was made between these methods depending on least square error (L.S.E) and (timing), which were calculated from the numerical solution and the exact solution.

The results in tables 1 and 2 show that:

- The methods were effective in solving the VIDEs of the 1st kind with accurate results.
- The results obtained by using the Quadratic Non-polynomial give the best approximation to our work.
- This method is useful when g(t) and k(t,s) are analytic **.**

Acknowledgments

The authors would like to thank Mustansiriyah University (WWW.uomustansiriyah.edu.iq) Baghdad –Iraq for its support in the present work.

References

1. Saleh A. J. , Esa R.E. and Jameel A.F. **2019**. Numerical treatment of non-linear Volterra integro-differential equation by using Runge-Kutta methods, AIP Conference Proceedings, 2138, 030034, doi.org/10.1063/1.5121071.

2. Rasheed M. A., Balasim A. T. and Jameel A. F. **2019**. Some Results for the Vorticity Transport Equation by using A.D.I Scheme, AIP Conference Proceedings, 2138, 030031, 2019, doi.org/10.1063/1.5121068.

3. Lakshmikantham, V. and Rama, M.M.R. **1995**.‘‘Theory of integro –Differential Equations ’’,Gordon and Breach Publishers ,1995.

4. Abd Alhammeed, F.T. **2002**. ‘‘Numerical solution of Integro-differential equation using Spline Functions”, M.Sc. Thesis, University of Technology.

5. Mohammad, R.K. **2006**. ‘‘Numerical Solutions of Non-linear Volterra Integro Differential Equations”,M.Sc. Thesis, Al-Mustansiriayah University.

6. Saba, N. M. **2014**. ”Solution of second kind Volterra Integro Equations Using Linear Non-polynomial Spline Function”, Mathematical Theory and Modeling, **4**(9).

7. Wizwaz, A.M. **2011**. ‘‘ *Linear and Nonlinear Integral Equation*”, Methods &Application ,Springer Science &Business Media.PP38.

8. Filiz, **2014** ‘‘ Numerical Method for a linear Volterra Integro –Differential Equations with Cash – Karp Method” , *Asian Journal of Fuzzy &Application Mathematics*, **02**(01): 1-11,2014.

9. Shawki, **S**. **2002**. "Solution of Second Kind Volterra Integro –Differential Equations”,M.Sc,Thesis University of Technology,2002.

10. Friedman, J., Hastie, T. and Tibshirani, R. **2007**. ‘‘ *The Elements of Statistical Learning*”, Applied Machine Learning ,chapter 5 ,‘‘ Splines and Applications”.

11. Larry L. **2007**. Schuhmker,‘‘ *Spline Function Basic Theory* ”,Third Edition ,Cambridge University Press ,New York.

12. Burhan, F. J., Abbas, H.T. **1992**. ‘‘ Non-polynomial Spline Method for the Solution of the System of Two Nonlinear Volterra Integral Equations”, *Kirkuk University Journal/ Scientific Studies(KUJSS)* , **11**(3): 15-25,2016.ISSN 1992-0849.

13. Muna, M. and Sarah, H. **2014**.‘‘ Solution of Second Kind Volterra Integral Equations Using Non-Polynomial Spline Function”, *Baghdad Science Journal*, **11**(2).

14. Adraa M. **2017**. ‘‘Numerical Solution of Volterra Integral Equation with Delay by Using Non-Polynomial Spline Function”, *Misan Journal for Academic Studies*.