ON THE WAVELET METHODS TO SOLVE PERTURBATION BOUNDARY VALUE PROBLEMS

Kais Ismail & Thamir Abdul Hafedh
Department of Computer Sciences
Mosul University

Received 20/12/2006 **Accepted** 04/04/2007

الملخص

يتضمن هذا البحث عرض النتائج العددية التي تؤكد التحسينات التي أجريت على طريقة (PASVA) لحل بعض المسائل للقيم الحدودية، وذلك باستخدام طرق الموجة القصيرة في التكرارات الداخلية في (PASVA) والتي تعتبر احد طرق حل المعادلات الاخطية.

Abstract

In this paper we present our numerical results to insure the improvement on the performance of PASVA algorithm to solve perturbation two-point boundary value problems, by using wavelet method in the inner loop of the nonlinear solver.

Introduction

Two point boundary value problems (TPBVP) arise naturally in the process of solving partial differential equations. Methods based on shooting, finite difference and collocation are well known in the literature and well represented also in standard software. Methods based on wavelet representation of the solution are relatively new and may very much be dependent on the type of wavelet to be used. We select several standard TPBVP problems, run our wavelet algorithm and compare its performance and results to those obtainable from PASVA. As a standard algorithm PASVA is well known for its outstanding performance on non-perturbation and mildly perturbation ordinary differential equations, while wavelets (considering their rich variety and neat representation) allow new ideas to be incorporated in order to enhance the performance of existing software. This paper documents preliminary results of our numerical experiments.

Two-point Boundary Value Problems (TPBVP)

We consider the general TPBVP

$$u'(t) = f(t,u) \text{ in } t \in [a,b]$$
$$r(u(a),u(b)) = 0. \qquad (1)$$

The currently accepted standard practice is to approximate it by a set of nonlinear algebraic equations in the following sense. In the interval $[a,b]$, introduce m mesh points $\{t_j\}$, $j = 1,2,3,\ldots,n$, with $n \geq 2$, $a = t_1 < t_2 < t_3 <\ldots< t_{n-1} < t_n = b$ and approximate $u(t_j)$ by x_j, $j = 1,2,\ldots,n$. The approximate equations which as the form

$$F_j(x_1,x_2,\ldots,x_n) = 0, \qquad j = 1,2,\ldots,n \tag{2}$$

is solved for x_j^*, $j = 1,2,\ldots,n$, and this equation, each F_j describes the relation which is to be satisfied at the mesh point t_j. On the basis of the assumption $u(t_j^*) \cong x_j^*$, $j = 1,2,3,\ldots,n$, we obtain the solution of (1). In this case $x_j \in R^n$ for $j = 1,2,3,\ldots,n$, and obviously $x \equiv [x_1,x_2,\ldots,x_n]^T$ and $F \equiv [F_1,F_2,\ldots,F_n]^T$ are in R^N with $N = n{\cdot}n$, where

We introduce a level function $\varphi(x) = ||F(x)||_2$, and solve the set of nonlinear equations by the damped Newton method:

$$x^{k+1} = x^k - \lambda_k[J(x^k)]^{-1} F(x^k), \tag{3}$$

Using the following algorithm (detail omitted):

1. start with an initial estimate x^0;
2. for $k = 0,1,2,\ldots$.:
   a. compute $F(x^k)$, $J(x^k)$ Jacobian of $F(x)$ at x^k;
   b. solve the linear equation for s^k :
      $J(x^k)s^k = -F(x^k)$;
   c. compute λ_k, $0 < \lambda_k \leq 1$, such that $\varphi(x^k+\lambda_k s^k) < \varphi(x^k)$ and set the next estimate $x^{k+1} = x^k+\lambda_k s^k$;
   d. test the convergence of that x^k; if yes, exit;
      otherwise set $k = k+1$, and go to (a).

Equation (2) is often called the discrete boundary value problem for the Eq.(1). If the approximation is consistent, and the discrete equation is numerically stable, then the solution to the discrete version converges to the desired solution of the original equation. For definitions of consistency, stability and convergence, see Watt (1968).

Methods to derive the nonlinear equation exist in literature. For example, the finite difference method used in PASVA (see further below) uses the trapezoidal rule:

$$\frac{x_{j+1} - x_j}{h_j} = \tfrac{1}{2}\left\{f(t_{j+1},x_j) + f(t_{j+1},x_{j+1})\right\} \tag{4}$$

where $h_j = t_{j+1} - t_j$ and $j = 1,2,\ldots,(n-1)$. We obtain

$$F_j(x_j,x_{j+1}) = x_j - x_{j+1} + \tfrac{1}{2}h_j.\lfloor f(t_j,x_j) + f(t_{j+1},x_{j+1})\rfloor = 0 \tag{5a}$$

which, together with the boundary conditions now written as

$$F_n(x_1,x_n) = r(x_1,x_n) = 0 \tag{5b}$$

Form well-defined equations of the standard type.

$G_j = I + \dfrac{h_j}{2}.\dfrac{\partial f}{\partial u}(t_j,x_j)$, and

$$\overline{G}_{j+1}=I-\frac{h_j}{2}.\frac{\partial f}{\partial u}(t_{j+1},x_{j+1}),$$

We obtain that

$$G_{j+1}^{-1}.G_j \cong I+\frac{h_j}{2}\left[\frac{\partial f}{\partial u}(t_{j+1},x_{j+1})+\frac{\partial f}{\partial u}(t_j,x_j)\right].$$

The Wronskian can be approximated by

$$W\left(t_{j+1},t_j\right)=I+\int_{t_j}^{t_{j+1}}\left(\frac{\partial f}{\partial f}\right)dt+\ldots$$

$$\cong W_h\,(t_{j+1},t_j).$$

and thus by the trapezoidal approximation, we arrive at

$$W_h\,(t_{j+1},t_j) = G_{j+1}^{-1}.G_j\,.$$

Next, by virtue of the group property of the Wronskian,

$$E=A+B\overline{G}_n^{-1}G_{n-1}\overline{G}_{n-1}^{-1}G_{n-2}\ldots\overline{G}_2^{-1}G_1$$

can be approximated by

$E\cong E_h\cong A+B.W_h(t_n,t_{n-1}).W_h(t_{n-1},t_{n-2})\ldots W_h(t_2,t_1)$ The non-singularity of $\overline{G}_{j+1}$ through a restriction on the stepwise h_j is then a basic feature of this method.

$E^*=A^*+B^*.W_h(a,b)$ is nonsingular with

$$A^*=\frac{\partial r}{\partial y(a)} \qquad \mathrm{B}^*=\frac{\partial \mathrm{r}}{\partial \mathrm{y(b)}}$$

$$E^*=\frac{\partial r}{\partial y^*(a)}+\frac{\partial \mathrm{r}}{\partial \mathrm{y(b)}}\bullet\frac{\partial \mathrm{y}^*(\mathrm{b})}{\partial^* y(a)}$$

Since equation (4) is only O(h^2) accurate, a combination of correction and mesh selection are developed to arrive at the user's prescribed tolerance. We recall that the n-vectors x_j are meant to approximate $u_j^*=u^*(t_j)$. If we write (5a,5b) with x_j replaced by u_j and expand in Taylor series around t_j + ½h_j, recalling that $f(t_j,u_j^*)=u^{*'}(t_j)$, we obtain the Local Truncation Error (LTE) of the method

$$\mathrm{LTE}_j=\frac{u_{j+1}^*-u_j^*}{h_j}-\tfrac{1}{2}\left[f(t_j,u_j^*)+f(t_{j+1},u_{j+1}^*)\right]$$

$$=\sum_{v=1}^{K}\frac{v\bar{f}^{(2v)}}{2^{2v-1}(2v+1)}\cdot\frac{h_j^{2v}}{(2v)!}+\mathrm{O}(h^{2K+2})$$

$$=\tfrac{1}{12}h_j^2u^{*(3)}(t_j+\tfrac{1}{2}h_j)+\mathrm{O}(h^4) \qquad (6)$$

with $\bar{f}^{(2v)}$ the 2vth derivative of $f(t,u^*)$ evaluated at t_j + ½h_j. If a mesh π of discretization points (t_j) has been selected such that LTEj is constant for j = 1,2,…,m, then the mesh is called equidistributing. Roughly speaking, an equidistributing mesh will have small step sizes when the third derivative of the solution is large.

PASVA insists on computing the leading term of the LTE*j* of the equedistributing mesh π^0. This is done in a two-pass algorithm. First, given an initial mesh π, the solution $x_j^{[0]}$ is computed. Then the LTE*j* are estimated, end the mesh is refined to achieve equedistribution…and so on until some stopping criteria are satisfied. See Lentini and Pereyra (1975,1977) and Pereyra (1965,1966,1967,1968) for the justification.

Let $x^{[0]}$ be the computed $O(h^2)$ solution of the nonlinear equations based on the equedistributing mesh π^0. Let $S_1(x^{[0]})$ be the corresponding $O(h^2)$ approximation t_i the LTE*j*, then by solving the linear equation

$$J(x^{[0]})\,\delta = -\,S_1(x^{[0]}),$$

an approximation to the global error $x_j^* - u_j^*$ is obtained; i.e.

$$\delta_j = x_j^* - u_j^* + O(h^2)$$

According to the Pereyra (1979). It turns out that by solving the nonlinear problem

$$F(x) = S_1(x^{[0]})$$

One obtains an $O(h^4)$ approximate solution $\bar{x}_j$ .this represents the first step of the deferred correction, which could be improved by improving the estimate on LTE*j*. Let $S_k(x^{[k-1]})$ be the first *k*th terms of the LTE*j* expansion, and let $x^{[k-1]}$ be the $O(h^{2k})$ approximate solution after (*k*-1) correction steps. The solution of

$$F(x) = S_k(x^{[k-1]}) \qquad (7)$$

is accurate to order h^{2k+2}. This is the iterative deferred correction algorithm, and given the user's tolerance requirement TOL, PASVA will attempt to obtain *x* satisfying

$$\max_{i,j} \left| x_{ij} - u_i^*(t_j) \right| \le \mathrm{TOL} \qquad (8)$$

On a mesh π^f containing the original mesh π provided by the user. Mesh refinement may occur. The basic strategy is t_i achieve (8) by increasing the order of the method and performing mesh refinement as well. Mesh refinement implies an increase in *m*, unequal value of h_j. Mesh recompilation of the Jacobian matrix. The computing cost for solving the nonlinear equations is proportional to the number of mesh pointed. The nonlinear equation is proportional to the number of mesh points. Once the nonlinear equation F(*x*) = 0 on a mesh π^k are solved, the Jacobian available in forced form, thus the deferred correction process based on π^k is no longer expensive. What is needed is a scheme to carefully manage the operation based on currently available information. PASVA contains a heuristic algorithm to do this (Soesianto, 1991).

The Wavelet Method

Wavelet analysis is a rapidly developing area in the mathematical sciences which is emerging an in independent field of investigation. Moreover, it has already created a common link between mathematicians, electrical engineers, and has even drawn a great deal of attention from scientists and engineers in other disciplines. In physics and engineering, problems requiring numerical solution of differental equations rank among the most computing-intensive operation. New mathematical bases of their computation are always developed. Wavelet represents one which permit accurate representation of a variety of functions and operators without redundancy. Through the ability to represent local, high frequency information with localized basis elements, wavelets allow adaptation in a straightforward, consistent fashion. Soesianto and Riyad Mubarak, 2000 gives a brief presentation on wavelet and wavelet transforms.

The Haar wavelet is a common and simplest wavelet transformation which can be used to transform values within a matrix. Haar transformation is a two-dimensional generalization of one-dimensional wavelet transform. The Haar transform uses the standard decomposition algorithm. To obtain the standard decomposition of a matrix, we first apply the one-dimensional wavelet transform to each row of value. This operation gives us an average value with detail coefficient for each row. Next we treat these transformed rows as if they were themselves an matrix and apply the one-dimensional transform to each column. The resulting values are all detail coefficients except for a single overall coefficient Stollintz, et al, (1996). An algorithm to compute the standard decomposition is given below

```
procedure StandardDecomposition (c: array [1 .. $2^j$,1 .. $2^k$] of reals)
    for row ←1 to $2^j$ do
        Decomposition(c[row, 1 .. $2^k$])
    end for
        for col ←1 to $2^k$ do
            Decomposition(c[1 .. $2^j$,col])
        end for
end procedure.
```

The corresponding reconstruction algorithm simply reserves the steps performed during decomposition:

```
procedure StandardRedecomposition (c: array [1 .. $2^j$,1 .. $2^k$] of reals)
    for col ←1 to $2^k$ do
        Redecomposition(c[1 .. $2^j$,col])
    end for
```

```
        for row ← 1 to 2^j do
            Redecomposition(c[row, 1 .. 2^k])
        end for
end procedure..
```

In this paper the wavelet method is used to solve the linear equation problem in the inner-loop of the nonlinear equation with two objectives: (1) cheaper operation, (2) more accurate computation. We proceed to the numerical experiments to obtain a general idea of the feasibility of introducing wavelet method in PASVA. In a later paper we will pursue the proposal of integrating wavelet concept to enhance the deferred correction and mesh refinement process.

Numerical Results

Problem 1. This problem is given in Cash (1994)
We consider a problem with a boundary layer at the right end of the interval:

$$\varepsilon\,\mathrm{u}'' - u = 0, \qquad \mathrm{u}(0) = 1, \;\; \mathrm{u}(1) = 0$$

The exact solution for this problems $u(x) = \dfrac{e^{\frac{-x}{\sqrt{\varepsilon}}} - e^{\frac{x-2}{\sqrt{\varepsilon}}}}{1 - e^{\frac{-2}{\sqrt{\varepsilon}}}}$, we run PASVA and our Wavelet method then we get the result below when we put the $\in = 0.01$.

Table 1. The result of TPBVPs produced by PASVA, MW, and exact solution for problem 1

x	(u)Pasva	(u)Wavelet	(u) Exact
.000000000E+00	.100000000E+01	0.100000000E+01	0.1000e+001
.781250000E-02	.924848800E+00	0.92308760258849	9.2312e-001
.156250000E-01	.855345300E+00	0.85208296583363	8.5214e-001
.234375000E-01	.791065100E+00	0.79442734424371	7.9453e-001
.312500000E-01	.731615600E+00	0.73330176934923	7.3345e-001
.390625000E-01	.676633800E+00	0.67686932577867	6.7706e-001
.468750000E-01	.625784000E+00	0.62476884589306	6.2500e-001
.546875000E-01	.578755600E+00	0.57666688662115	5.7695e-001
.625000000E-01	.535261400E+00	0.53225559542361	5.3259e-001
.781250000E-01	.457833300E+00	0.45816583344933	4.5841e-001
.937500000E-01	.391605600E+00	0.39049926667963	3.9063e-001
.101562500E+00	.362176000E+00	0.36041477625489	3.6059e-001
.109375000E+00	.334958000E+00	0.33598302470858	3.3622e-001
.125000000E+00	.286504800E+00	0.28632110882879	2.8650e-001
.140625000E+00	.245060500E+00	0.24398901333502	2.4414e-001
.156250000E+00	.209611400E+00	0.20997541836969	2.1014e-001
.171875000E+00	.179290100E+00	0.17890164078226	1.7907e-001
.187500000E+00	.153354900E+00	0.15240774519887	1.5259e-001
.203125000E+00	.131171400E+00	0.13111319816529	1.3134e-001
.218750000E+00	.112196900E+00	0.11165062197728	1.1192e-001
.234375000E+00	.959670600E-01	0.09600033376204	9.6328e-002
.250000000E+00	.820849700E-01	0.08168750127639	8.2085e-002
.281250000E+00	.600546300E-01	0.05990732280339	6.0205e-002
.312500000E+00	.439368900E-01	0.04346321606655	4.3718e-002
.343750000E+00	.321448800E-01	0.03177717071410	3.2065e-002
.375000000E+00	.235176600E-01	0.02314491146724	2.3518e-002
.437500000E+00	.125879800E-01	0.01245514128323	1.2525e-002
.500000000E+00	.673764100E-02	0.00676141723832	6.7376e-003
.562500000E+00	.360599200E-02	0.00363816697231	3.5880e-003
.625000000E+00	.192938600E-02	0.00197428180933	1.9294e-003
.687500000E+00	.103130300E-02	0.00106093325171	1.0261e-003
.750000000E+00	.549357700E-03	0.00057319392538	5.4936e-004
.812500000E+00	.289082400E-03	0.00030328294381	2.8757e-004
.875000000E+00	.145453900E-03	0.00015517840426	1.4545e-004
.937500000E+00	.605170200E-04	0.00006578657846	5.9973e-005
.968750000E+00	.288388800E-04	0.00003138522898	2.8601e-005
.100000000E+01	.131415700E-15	000000000E+00	000000000E+00

Problem 2. This problem is given in Cash (1995**)**

We consider a problem with a boundary layer at the right end of the interval: $\varepsilon\, u'' + xu = 0, \qquad u(-1) = 0,\ \ u(1) = 2$

The exact solution for this problems $u(x) = \dfrac{erf\left(\dfrac{x}{\sqrt{2\varepsilon}}\right)}{erf\left(\dfrac{1}{\sqrt{2\varepsilon}}\right)}$, we run PASVA and our Wavelet method then we get the result below when we put the $\in$ = 0.1.

Table 2. The result of TPBVPs produced by PASVA, MW, and exact solution for problem 2

x	(u) Pasva	(u)Wavelet	(u) Exact
-1.000000000	.000000000E+00	.000000000E+00	000000000E+00
-.973684200	.511949500E-03	0.00045595938299	5.0499e-004
-.947368400	.117331800E-02	0.00107017268180	1.1838e-003
-.894736800	.310290100E-02	0.00284013335190	3.0908e-003
-.868421100	.447080700E-02	0.00412869769205	4.4954e-003
-.842105300	.618970300E-02	0.00569125818801	6.1974e-003
-.789473700	.109931400E-01	0.01024011963971	1.1046e-002
-.736842100	.182640900E-01	0.01702897198811	1.8238e-002
-.684210500	.289701200E-01	0.02727546783998	2.9021e-002
-.631578900	.443043800E-01	0.04171961820779	4.4160e-002
-.578947400	.656689500E-01	0.06230894786573	6.5644e-002
-.526315800	.946238900E-01	0.09036167089042	9.4824e-002
-.473684200	.132796300E+00	0.12668982616070	1.3254e-001
-.421052600	.181748500E+00	0.17425436052368	1.8180e-001
-.368421000	.242813600E+00	0.23376470368238	2.4335e-001
-.315789500	.316912400E+00	0.30462618642957	3.1659e-001
-.263157900	.404375900E+00	0.38992745914598	4.0466e-001
-.210526300	.504800400E+00	0.48602320161942	5.0384e-001
-.157894700	.616963600E+00	0.59545924958278	6.1673e-001
-.105263200	.738822600E+00	0.71445994367457	7.3945e-001
-.052631570	.867606900E+00	0.83783212108040	8.6669e-001
.000000007	.100000000E+01	0.96712552851499	1.0000e+000
.052631590	.113239300E+01	1.09641893596982	1.1333e+000
.105263200	.126117700E+01	1.21979111336288	1.2605e+000
.157894700	.138303600E+01	1.33879180752674	1.3833e+000
.210526300	.149520000E+01	1.44822785532819	1.4962e+000
.263157900	.159562400E+01	1.54432359794601	1.5953e+000
.315789500	.168308800E+01	1.62962487044292	1.6834e+000
.368421100	.175718600E+01	1.70048635328681	1.7566e+000
.421052600	.181825100E+01	1.75999669638335	1.8182e+000
.473684200	.186720400E+01	1.80756123085404	1.8675e+000
.526315800	.190537600E+01	1.84388938613877	1.9052e+000
.578947400	.193433100E+01	1.87194210913752	1.9344e+000
.631578900	.195569600E+01	1.89253143886851	1.9558e+000
.684210500	.197103000E+01	1.90697558918891	1.9710e+000
.736842200	.198173600E+01	1.91722208507964	1.9818e+000
.789473700	.198900700E+01	1.92401093744926	1.9890e+000
.842105300	.199381000E+01	1.92855979878770	1.9938e+000
.868421100	.199552900E+01	1.93012235942794	1.9955e+000
.894736900	.199689700E+01	1.93141092373170	1.9969e+000
.947368400	.199882700E+01	1.93318088437344	1.9988e+000
.973684200	.199948800E+01	1.93379509768011	1.9995e+000
1.000000000	.200000000E+01	.200000000E+01	.200000000E+01

Problem 3: This problem is given in Cash (1996)
We consider a problem with a boundary layer at the right end of the interval:

$$\varepsilon\, \mathrm{u}'' - u' = 0, \qquad \mathrm{u}(0) = 1,\ \ \mathrm{u}(1) = 0$$

The exact solution for this problems $u(x) = \dfrac{1 - e^{\frac{x-1}{\varepsilon}}}{1 - e^{\frac{-1}{\varepsilon}}}$, we run PASVA and our Wavelet method then we get the result below when we put the $\in$ = 0.01.

Table 3. The result of TPBVPs produced by PASVA, MW, and exact solution for problem 3

x	(u) Pasva	(u)Wavelet	(u) Exact
.000000000	.100000000E+01	1.00000000000000	1.00000000000000
.052631580	.100000000E+01	0.99999917470581	1.00000000000000
.105263200	.100000000E+01	1.00000078670631	1.00000000000000
.157894700	.100000000E+01	0.99999994708488	1.00000000000000
.210526300	.100000000E+01	1.00000132891930	1.00000000000000
.263157900	.100000000E+01	0.99999993929315	1.00000000000000
.315789500	.100000000E+01	0.99999996311693	1.00000000000000
.368421000	.100000000E+01	0.99999997715468	1.00000000000000
.421052600	.100000000E+01	1.00000200320727	1.00000000000000
.473684200	.100000000E+01	0.99999998626601	1.00000000000000
.526315800	.100000000E+01	0.99999993658944	1.00000000000000
.578947400	.100000000E+01	0.99999999693698	1.00000000000000
.631578900	.100000000E+01	0.99999996283448	1.00000000000000
.684210500	.999999900E+00	0.99999990798167	0.99999999999998
.736842100	.100000000E+01	0.99999983654878	0.99999999999622
.789473700	.999999900E+00	0.99999997457454	0.99999999931390
.815789500	.100000000E+01	1.00000314654388	0.99999998979104
.842105300	.999999900E+00	0.99999997148043	0.99999986254923
.859649100	.999999300E+00	0.99999996561794	0.99999916847128
.877193000	.999995500E+00	0.99999919437383	0.99999544825554
.894736800	.999973200E+00	0.99998540359013	0.99997246355065
.905263100	.999923200E+00	0.99995638339149	0.99992514817011
.915789500	.999779900E+00	0.99984972449140	0.99977513267582
.926315800	.999369200E+00	0.99954920332247	0.99938874723887
.936842100	.998192500E+00	0.99844737344034	0.99816369522297
.947368400	.994821100E+00	0.99534209083208	0.99500840609309
.952153100	.991643200E+00	0.99223671409578	0.99177025295098
.956937800	.986515600E+00	0.98706124021676	0.98643144098780
.961722500	.978241500E+00	0.97843528254048	0.97762922814383
.966507200	.964890400E+00	0.96405979543972	0.96311683259876
.971291900	.943347000E+00	0.94608858279115	0.94497677994359
.976076500	.908585000E+00	0.91014737441623	0.90928204671059
.978468900	.883877600E+00	0.89018780650929	0.88919684163767
.980861200	.852492400E+00	0.85142104716872	0.85043138077737
.983253600	.812624400E+00	0.81840350541827	0.81731647594727
.985645900	.761981100E+00	0.75431063026188	0.75340303605839
.988038300	.697650000E+00	0.69971299587509	0.69880578808780
.990430700	.615931600E+00	0.63298255383190	0.63212055882856
.992025600	.549521300E+00	0.55142312467670	0.55067103588278
.993620400	.471631000E+00	0.45173937792821	0.45118836390597
.995215300	.380269300E+00	0.39402773507389	0.39346934028737
.996411400	.301524900E+00	0.33024118244542	0.32967995396436
.997607600	.212770300E+00	0.18140589297884	0.18126924692202
.998803900	.112737700E+00	0.09523809381386	0.09516258196404
1.000000000	.413500300E-15	.413500300E-15	0

The Comparison Between PASVA and MW to Solve Perturbation TPBVPs

In this section we tried to solve the same problems above and we found that PASVA failed to solve the problems above when the value of $\in < 0.01$, this is called a perturbation problems. But MW able to solve these perturbation problems and the table below include the results of problem 1.

Table 4. the result of perturbation TPBVPs produced by MW and failed of PASVA

x	$\in = 0.001$	$\in = 0.0001$	$\in = 0.00001$	PASVA $\in < 0.01$
.000000000E+00	0.100000E+01	0.100000E+01	0.100000E+01	failed
.781250000E-02	7.7697e-001	4.5884e-001	1.2085e-001	failed
.156250000E-01	6.0367e-001	2.1134e-001	1.5144e-002	failed
.234375000E-01	4.8373e-001	1.0572e-001	2.1575e-003	failed
.312500000E-01	3.7567e-001	4.8455e-002	2.6059e-004	failed
.390625000E-01	2.9166e-001	2.2198e-002	3.1475e-005	failed
.468750000E-01	2.2631e-001	1.0149e-002	3.8015e-006	failed
.546875000E-01	1.7545e-001	4.5939e-003	4.5765e-007	failed
.625000000E-01	1.3581e-001	1.9793e-003	4.2723e-008	failed
.781250000E-01	8.4929e-002	4.9127e-004	1.6991e-009	failed
.937500000E-01	5.1713e-002	1.2237e-004	7.7404e-011	failed
.101562500E+00	4.0069e-002	5.5391e-005	9.6487e-012	failed
.109375000E+00	3.1985e-002	2.5867e-005	1.0187e-012	failed
.125000000E+00	1.9391e-002	5.9787e-006	3.6959e-014	failed
.140625000E+00	1.1761e-002	1.3959e-006	1.3778e-015	failed
.156250000E+00	7.3428e-003	3.4503e-007	5.4778e-017	failed
.171875000E+00	4.4510e-003	7.9747e-008	2.0586e-018	failed
.187500000E+00	2.6985e-003	1.8620e-008	-2.9273e-018	failed
.203125000E+00	1.6830e-003	4.6046e-009	-9.7714e-018	failed
.218750000E+00	1.0172e-003	1.0745e-009	-9.7578e-019	failed
.234375000E+00	6.2948e-004	2.6314e-010	6.7763e-021	failed
.250000000E+00	3.7203e-004	5.0296e-011	1.1858e-020	failed
.281250000E+00	1.4425e-004	4.3161e-012	-9.1056e-020	failed
.312500000E+00	5.4538e-005	3.5900e-013	4.0658e-020	failed
.343750000E+00	2.0881e-005	3.1070e-014	-2.5411e-020	failed
.375000000E+00	7.2913e-006	1.9853e-015	1.0694e-020	failed
.437500000E+00	1.2626e-006	3.6429e-017	1.3553e-020	failed
.500000000E+00	2.2194e-007	-2.0817e-017	2.7105e-020	failed
.562500000E+00	3.8431e-008	-2.3039e-019	1.6498e-021	failed
.625000000E+00	6.7557e-009	-6.7085e-019	-3.5734e-022	failed
.687500000E+00	1.1698e-009	-2.2362e-019	6.6174e-024	failed
.750000000E+00	2.0564e-010	-2.0329e-020	9.9262e-023	failed
.812500000E+00	3.5611e-011	1.7138e-020	-4.7043e-025	failed
.875000000E+00	6.2764e-012	-9.4609e-022	8.0639e-027	failed
.937500000E+00	1.1827e-012	-8.3028e-023	-2.3367e-028	failed
.968750000E+00	3.9941e-013	1.1374e-024	-8.1351e-031	failed
.100000000E+01	000000000E+00	. 000000000E+00	000000000E+00	failed

Conclusion

The Wavelet Method for solving two point boundary value problems is introduced. The method examined for several popular problems and compared the results with the results of the same problems solved by PASVA and the exact solution for each problem. From the tables in our discussions we can conclude that:

1. The accuracy of both methods are same
2. Although PASVA software is well-known to solve TPBVP but it failed to solve perturbation problems, WM is able to do it.
3. The result of $\in = 0.00001$ method was failed, and PASVA method when $\in = 0.001$ already failed.

REFERENCES

Cash, J.R., 1994, “Mesh Selection for Stiff Two Point Boundary Value Problems”, Numerical Algorithms., 7, 205-224.

Cash, J.R., 1996, “Runge-Kutta Methods for the Solution of Stiff Two-Point Boundary Value Problems”, Appl. Numer. Math., 22, 165-177.

Cash, J.R., Moore, G., and Wright, R.W., 1995, “An Automatic Continuation Strategy for the Solution of Singularly Perturbed Two-Point Boundary Value Problems”, J. Comput. Phys., 122, 266-279.

Lentini M., and Pereyra, V., 1975, “PASVA2-Two Point Boundary Value Problem Solver for Nonlinear First Order Systems”, Lawrence Berkeley Lab. Program Documentation Rep.

Lentini, M and Pereyra, V, 1977, “An Adaptive Finite Difference Solver for Non-linear Two-Point Boundary Problems with Mild Boundary Layers”, SIAM J. Numer. Anal., 14, 91-112.

Pereyra, V., 1965, “The Difference Correction Method for Nonlinear Two-Point Boundary Value Problems of Class M”, SIAM J. Numer. Anal., 22, 184-198

Pereyra, V., 1966, “On Improving An Approximate Solution of A Functional Equation by Deferred Corrections”, Numer. Math., 8, 376-391.

Pereyra, V., 1967a, “Accelerating the Convergence of Discretization Algorithms”, SIAM J. Numer Anal., 4, 508-833.

Pereyra, V., 1967b, “Iterated Deferred Corrections for Non-Linear Operator Equations”, Numer. Math. 10, 316-323

Pereyra, V., 1968, “Iterated Deferred Corrections for Non-Linear Boundary Value Problems”, Numer Math 11, 111-125.

Pereyra, V., 1979, “Pasva3: An adaptive finite difference FORTRAN program for first –order non-linear ordinary boundary problems”, Lecture Notes in Computer Science, pp. 67-88., Spinger-Verlag, Berlin-Heidelberg.

Soesianto, F., 1991; *On the Solution of The Highly Structured Non-Linear Equation*” Ph.D. Dissertation from Department of Computer Science, University of Essex, UK.

Soesianto, F., and Abdullah, R. M., 2000, "Two Competing Methods to Solve Sparse Linear Equations", *this proceeding.*

Stollintz, E. J., T. D. Derose, and D.H. Salesin, Wavelet for Computer Graphics Theory and application, Morgan Kaufmann Publishers, Inc., San Francisco, California, 1996.

Watt, J.M., 1968, “Convergence and Stability of Discretization Methods for Function Equation ’’, Comput. J., 11, 77-82