New Parallel Algorithms for BVPs in ODEs

Bashir M. Khalaf
College Of Education
University of Mosul, Iraq

Received on: 16/02/2002 Accepted on: 01/09/2002

ABSTRACT

The main objective of this paper is the development of a new parallel integration algorithm for Solving Boundary Value Problem (BVPs) in Ordinary Differential Equation, (ODEs). This algorithm is suitable for running on MIMD computing systems. We will analysis the stability and error control of the developed algorithm .We shall also consider the treatment of stiff boundary value problems by developed technique.

Keywords: Ordinary Differential Equation (ODEs), parallel integration algorithm, stability, error control, stiff boundary value problems.

خوارزميات متوازية جديدة لمسائل القيم التخومية في المعادلات التفاضلية الاعتيادية

بشير محمد خلف

كلية التربية

جامعة الموصل

تاريخ الاستلام: 2002/02/16 تاريخ القبول: 2002/09/01

الملخص

هدف هذا البحث الرئيسي هو تطوير طريقة جديدة متوازية لحل مسائل القيم التخومية في المعادلات التفاضلية الاعتيادية ملائمة للتنفيذ في حاسبات متوازية من نوع MIMD (حاسبات ذات عمليات متعددة في آن واحد). تمت دراسة خاصية الأستقرارية والسيطرة على الخطأ للطريقة وتم تحسين الطريقة عن طريق زيادة عدد

التخمينات والتكامل الرجعي. كما تمت معالجة مسائل القيم التخومية الجافة بالطريقة المبتكرة.

الكلمات المفتاحية: المعادلات التفاضلية الاعتيادية، طريقة متوازية، الأستقرارية، السيطرة على الخطأ، مسائل القيم التخومية الجافة.

Introduction

Numerical solution of BVPs in ODEs is a very active research area. Mostly the numerical solution of these problems takes a lot of computer time, specifically, if the integration interval is very large or if the system describing the problem consists of a large number of differential equations or if the problem is a stiff problem (consult the following references for more detail Cash (1995), Khalaf (1988, 1990), Khalaf and Al-Wajih (2000), Khalaf and Al-Murshid (2000). Hence the objective of this research is the development of a new parallel algorithm which combines parallel integration processes with parallel interpolation to estimate the unknown BCs.

The developed algorithm is suitable for running on a MIMD (Multiple instruction streams with Multiple Data streams) computer (Flynn (1972)), Khalaf, (1990), Meiko (1989), and Khalaf (1995)).

MIMD Computer Consists of several processors, each processor has its own memory and processing unit. These processors communicate through a suitable communication network. (for more detail, see Meiko (1989), Khalaf (1990), Al-Wajih (1999) and Al-Murshid (2000)). In MIMD computer each processor can carry out its own set of instructions, often on its own set of data, independently of all the other processors. Such computers usually number their (more complex) processors in tens rather than thousands that may be found in SIMD (Single Instruction Stream with Multiple Data Stream) computers. MIMD computers are well suited to algorithmic parallelism in

which problems can be separated into concurrent independent processors (Brocklehurs, 1992).

1-The Numerical Methods for BVPs:

The well - known numerical methods for solving BVPs in ODEs are:

i) Finite difference methods (Fd), ii) shooting methods and iii) collocation methods. Finite difference methods based on dividing the given interval of the independent variable by nodes and then approximating the differential equation by a given finite difference formulas at each node and this will produce a set of algebraic equations mostly non - linear which can be solved by Newton iteration or one of its alternatives (for more detail see Khalaf (1988)). To get accurate results by using these methods we have to increase the number of nodes which will produce a great number of algebraic equations which increase the complexity of the solution and takes a lot of computer time. Mostly, the iteration process will diverge because the approximation processes at the nodes will create noisy data and this noise can be accumulated by iteration processes and render the solution meaningless.

Shooting methods based on dividing the integration interval to n subintervals. At the beginning of each subinterval, values estimated for the given dependent variables then the ODEs of the problem integrated in the subinterval, by using the estimated values and then at the end of each subinterval the corresponding estimated and integrated values of corresponding depended variables are matched, i.e. at the end of each subinterval matching functions are defined. The estimated values can readjusted by Newton iteration or one of its alternatives. The problem with these methods is that the estimated values need to be very close to real solution, otherwise the iteration processes will diverge. (For more detail see for example Keller (1988), Khalaf (1988, 1990)). For Collocation methods see Asher et al. (1.988), and Khalaf (1997)) based on approximating the solution

of the ODE by a linear combination of a set of independent simple functions. The coefficients of the combination can be estimated (using the boundary conditions BCs and substituting the differentiation of the approximate solution at given nodes of the independent interval) by Newton methods or by some other iterative methods. Hence the iteration will diverge at most if there are noises or error in combination coefficients so that we try here to develop a new method for solving BVDs in ODEs which uses integration's and interpolation instead of iteration processes. The new methods are more suitable for running on MIMD computers, (see KLhalaf and Matti (2001) and Khalaf and Sawoor (2002)

2-The new method:

Let us consider a two point BVP (this will not affect the generality of the method):

$$\mathbf{y'' = f(x, y, y'), y(a) = \alpha, y(b) = \beta, x \in (a, b)} \qquad \mathbf{(2.1)}$$

The above problem can be reduced to the following system.

$$y_1' = y_2$$
$$y_2' = f(x, y_1, y'), y_1(a) = \alpha, y(b) = \beta \qquad (2.2)$$

To integrate system (2.2) in the interval (a,b) we need a value of $y_2(a)$ which is unknown. To get this value, we proceed as follows:

Process 1: estimate a value S_1 for $y_2(a)$ and integrate the system (2.2) in the interval (a,b), we get $y_1(b) = -m_1$.

Process 2: estimate another different value 82 for y2(a) and integrate the system (2.2) in the interval (a,b), we get $y_1(b) = m_2$, and so on

Process n: estimate another different value Sn for $y_2(a)$ and integrate the system (2) in the interval (a,b) , we get $y_1(b) = m_n$. Hence we get the following table of data:

$y_2(a)$	S_o	S_1	S_2	- -	S_n
$y_1(b)$	m_o	m_1	m_2	- -	m_n

we can write the above table as:

$y_1(b)$	m_o	m_1	m_2	- -	m_n
$y_2(a)$	S_o	S_1	S_2	- -	Sn

Then by using Newton divided difference formula for interpolation,we can find the value of $y_2(a)$ corresponding to $y_1(b)=\beta$ by:

$$p(\beta)=S_o+(\beta-m_o)\Delta S_o + (p-mo)\,(P-ml)\,\Delta^2 S_o+\ldots$$

$$+(\beta-m_o)(\beta-m_l)\ldots(\beta-m_{n-l})\,\Delta^n So$$

And the approximate value of $y_1(a)$ corresponding to $y_1(b)=\beta$ is $y_2(a)=p(\beta)$

3-ParalleIising of the Algorithm

It is clear from the last section that the integration processes 1,2,3,...,n are independent processes. Hence each integration process can be carried out in a processor, i.e. each integration process can be assigned to one of the processors of a MIMD computer, which consists of p processors. If p>n then integration processes can be speeded up by $S_p=n$, where S_p is the speed-up factor and defined as follows:

$$S_p = \frac{\textbf{Total execution time of the task in a single processor}}{\textbf{Total time of the execution of the task in p processors}}$$

if p<n and n/p=r (r is a positive integer) then the integration processes can be speeded-up approximately by the factor S_p=r. It is also clear that finding $\Delta.S_o$, $\Delta.S_1$, ...,$\Delta.S_{n-1}$ are independent processors, so that they can be calculated simultaneously. Calculation of Δ^2S_o, Δ^2S_1,… Δ^2S_{n-2} are independent so that they can be calculated in parallel and the higher order calculation Δ^kS_i, i=0,l,...,n-k, k=3,...,n, $S_i=S(m_i)$ can be done similarly.

4-Testing the new method

In this section we try to show the effectivenesses of the new method for solving linear BVPs and non - linear BVPs:

Example (I): (Linear Problems)

$y'' = -y' + 2y$, y(o)=l, y(l)=e, $y= e^x$

We can reduce the problem to the following form:

$= y_2\ y_1'$

$=-y2+2y1\ y_2'$

$y_1(0)=1$

$y_1(1)=e$

To integrate the above system by the new method, we give the following values for $y_2(0)$: -1,0,2,3, and we integrate the system by Euler method using step size h=0.5 we get the following table:

$S_i=y_2(0)$	-1	0	2	3

$M_i=y_1(l)$	0.75	1.5	3	3.75

We rewrite the above table as:

$y_1(1)$	$y_2(0)$	ΔS_i
0.75	-1	ΔS_o=1.333
1.5	0	ΔS_1=1.333
3	2	ΔS_2=1.333
3.75	3	

$y_2(0)$=-l+(e-0.75)(1.33)+0=1.624

Where the exact value of y2(0) is 1.

The accuracy of the final value of $y_2(0)$ can be increased by reducing the step size and increasing the number of estimations or the accuracy can be increased by using higher order integration methods.

Example (2): (a non - linear problem) Consider the following nonlinear BVPs:

$y''=6\,y^{\frac{1}{3}}$, y(0)=0, y(l)=l, $y(x) = x^3$

To integrate the system by the new method we need estimation for the unknown BC which is y'(0). The result obtained by the method for y'(0) is 0.671 and the exact value is 0. The accuracy of the result can be increased by: i) reducing the step size h. ii) by increasing the number of estimation and iii) by using a higher order integration method.

Example (3):

Let us consider the BVP:

y''=-xy'+3y+4.2x , y(0) - 0, y(l)=1.9, y{x)=x3 +0.9x

We can reduce the problem to the following form:

$y'_1=y2$ $y_1(0)=0$
$y'_2=-xy,+3y,+4.2x$ $y_1(l)=1.9$
This example can be solved by section (2) to get the value: $y_2(a)=y_2(0)=0.597992$, where the exact value is $y_2(0)=0.9$, i.e. the error of the solution is 0.302008.

5-Improving the method by using higher order integration methods

To solve example (1) by the method we give the following value for y2(0): -1,0,2,3 using forward integrating process but we integrate the system by a higher-order method, such as, fourth order R-K method using step size h=0.5 ,we get the following results:

$S_i=y_2(0)$	-1	0	2	3
$M_i=y_1(l)$	0.471	1.571	3.771	4.871

We rewrite the above tables as:

$Y_1(0)$	$Y_2(0)$	ΔS_I	$\Delta^2 S_i$	$\Delta^3 S_i$

0.471	-1	0.90909		
1.571	0	0.9090	0	
3.77	2	0.9090	0	0
4.871	3			

$p(\beta)=S_o+(\beta-m_o)\Delta S_o + (\beta -m_o)(\beta -m_l)\ \Delta^2 S_o+(\beta -m_o)(\beta-m_l)(\beta – m_2)\ \Delta^3 So$

$= -l+(e-0.471)(0.9090909)+0$

$\therefore p(\beta)=1.043$

And since the exact value of $y_2(0)$ is 1, i.e. the error of this method is $4.29817\times l0^{-2}$.

So that the effectiveness of a higher-order integration method is clear.

6-Improving the method by backward integration techniques:

To improve the performance of the parallel integration and interpolation method for BVPs , we have used Backward integrating processes to estimate the value of unknown boundary condition (BC) y2(a) . This improvement is done by integrating the system from b to a by using negative step size - h , h > 0 . The 2nd - order boundary - value problem (BVP):

$y''=f(x,y,y')\ .\ a \le x \le b$

$y(a)=\alpha$, $y(b)=\beta$ (6.1)

Can be reduced to a system of equations of the form :

$y_1 = y_2$ $\quad y_1(a) = \alpha$

$= f(x, y_1, y_2)$ $\quad y_1(b) = \beta\ y'_2$

To integrate system (6.2) in the interval (a,b) we need a value of $y_2(a)$ Which is unknown. To get this value we proceed as follows:

Process (I): estimate a value S_1 for $y_2(b)$ and integrate the system (6.2) from b to a by using negative step size-h , $h > 0$ i.e. using one of the single-step method such as Euler method, we get:

$$y_2(a) = r_1$$

Process (2): estimate another different value S_2 for $y_2(b)$ and integrate the system (6.2) from b to a by using negative stepsize -h,h>0 i.e. using one of the single-step method such as Euler method, we get $y_2(a) = r_2$,and so on

Process (n): estimated another different value Sn for $y_2(b)$ and integrate the system (6.2) from b to a by using negative step size -h,h>0 i.e. using one of the single-step method such as Euler method, we get $y_2(a) = r_n$.

Hence we get the following table of the data:

$S_i = y_2(b)$	S_o	S_1	S_2	…	S_n
$r_i = y_i(a)$	r_o	r_1	r_2	…	r_n

Then by using Newton divided difference formula for interpolation we can find the value of $y_2(a)$ corresponding to $y_1(b) = \beta$. Here we do not need the conversion of the table.

$y_2 = p(\beta) = r_o + (\beta - S_o)\,\Delta r_o + (\beta - S_o)(\beta - S_i)\,\Delta r_o + \ldots +$
$(\beta - S_o)(\beta - S_1)(\beta - S_2)\ldots(\beta - S_{n-1})\Delta_n$ "ro and the approximate value of y2(a) corresponding to $y_1(b) = \beta$ is $y_2(a) = p(\beta)$

6-1 Parallelising of the Backward integration algorithm

It is clear from the last section that the integration processes 1,2,3,..., n are independent processes. Hence each integration process can be carried out in a processor, i.e. each integration process can be assigned to one of the processor of a MIMD computer that usually number their processors in tens rather than thousands that may be found in single instruction stream with multiple data stream computers, which consists of p processors, if p>n then integration processes can be speeded-up by Sp=n, where Sp is the speed-up factor and defined as follows:

$$S_p = \frac{\textbf{Total execution time of the task in single processor}}{\textbf{Total time of the execution of the task in p processors}}$$

If p<n and n/p=m (m is a positive integer) then the integration processes can be sped up by the factor S_p=m. It is also clear that finding Δr_o, Δr_1 ,..., Δr_{n-1} are independent processors, so that they can be calculated simultaneously. Calculations of $\Delta^2 r_o$, $\Delta^2 r_1$,..., $\Delta^2 r_{n-1}$ are independent so that they can be calculated in parallel and the higher order calculations $\Delta^k r_i$, k=3,...,n, $r_i = r(S_i)$ can be done similarly.

6.2 Testing of the newly developed method Example(4):

Let us consider the BVP of example (1).

$y''=- y'+2y$ $y(0)=l, y(l)=e, y(x)=e^x$

which can be reduced to the following systems

$y_1' = y_2$ $\mathbf{y_1(0)=1}$

$y_2' = \mathbf{-y2+2y, y_1(1)=e}$

To integrate the above system by using backward integrating process, we give the following values for $y_2(b)=y_2(l)= -1,0,2,3$ and integrate the system by Euler method using negative step size –h,h=0.5>0 and x_o=b= 1.0 ;

$x_i=x_o-h= 0.5$;

$x_2= a=x_1-h= 0.0$, we get the following results:

$S_i=y2(b)=y_2(l)$	-1	0	2	3
$R_i-y_2(a)=y_2(0)$	-9.5457	-6.7957	-1.2957	1.4543

$Y_2(b)$	$y_2(a)$	Δ r	Δ^2r	Δ^3 r
-1	**-9.5457**	**2.75**		
0	**-6.7957**	**2.75**	**0**	**0**
2	**-1.2957**	**2.75**	**0**	
3	**1.4543**			

$\therefore$ y2(a)=y2(0)=p(p)=-9.5457+(e+l)(2.75)+0 =0.67957

The error of this newly method is 0.32043 and the effectiveness of this technique is clear from the comparison of the errors, where the error of the forward integrating process is 0.624.

Example (5): (a non-linear problem) If we have he following BVP:

$$y'' = y^3 - yy' \qquad , 1 \le x \le 2$$

$$y(1)= \frac{1}{2} \ ,y(2)= \frac{1}{3} \ , y(x) = \frac{1}{(x+1)}$$

The value of y2(a) by using forward integrating process is 0.7548172 and the exact value is 0. To integrate the system by the newly developed method that uses backward integrating process: first, reduce the BVP to the following system:

$$y_1' = y_2 \qquad y_1(1)= \frac{1}{2}$$

$$y_2' = y_1^3 - y_1y_2 \qquad y_1(2)= \frac{1}{3}$$

So $x_o = b = 2.0$

$x_1 = x_o - h = 1.5$

$x_2 = a = x_1 - h = 1.0$

By estimating the following values for $y_2(b) = y_2(2) = -1, 0, 2, 3$; and integrating the system by Euler method with negative step size -h, h=0.5>0, we get the following results:

$S_i = y_2(b) = y_2(2)$	**-1**	**0**	**2**	**3**
$R_i = y_2(a) = y_2(l)$	**-0.98071004**	**-0.033950617**	**3.2345678**	**6.3063256**

S_i	r_I	Δr_I	$\Delta^2 r_i$	$\Delta^3 r_i$
-1	-0.98071004	0.94675943	0.2291666	0.302083
0	0.03950617	1.6342592	1.4374986	
2	3.2345678	3.0717578		
3	6.3063256			

$P(\beta) = y_2(a) - y_2(l) = 0.1597225$, this value which is obtained by backward integrating process is more accurate than that obtained by forward integrating process , $y_2(a) = 0.7548172$, compared with the exact value 0.

6.3-Improving the performance of the backward integration process:

The accuracy of the results that are obtained by backward integration process can be increased by increasing the number of estimations provided that $\Delta^2 r_i$, $\Delta^3 r_i$, …, $\Delta^k r_i$, k=2,... ,n-l are not equal to zero.

Example (6):

Let us consider the non- linear BVP of example (5).

$$y'' = y^3 - yy' \quad , \qquad 1 \le x \le 2$$

$$y(1)=\frac{1}{2}\ ,y(2)=\frac{1}{3}\ ,\ y(x)=\frac{1}{(x+1)}$$

where the estimation value for yz(b) that have been used is:

$y_2(b)$=-1,0,2,3

In this example we will increase the number of estimations for $y_2(b)$ by: $y_2(b)$-$y_2(2)$= -9,-4,-2,-1,0,2,3

and, also, integrate the system by Euler method with negative step size -h, h=0.5>0, we get

$S_i=y_2(b)$	-9	-4	-2	-1	0	2	3
$r_i=y_2(a)$	-41.5547	-7.7407	-1.9691	-9.8071E-01	-0.0339	3.2345	6.3063

Table (I): This table gives the values of $\Delta^k r_i$, k=l,...,6 which are used to find $y_2(a)$

S_I	R_I	Δr_I	$\Delta^2 r_i$	$\Delta^3 r_I$	$\Delta^4 r_I$	$\Delta^5 r_I$	$\Delta^6 r_I$
-9	-41.5547						
		6.7658					
-4	-7.7407		-0.6442				
		2.2533		0.0396			
-2	-1.9691		-0.3266		5.8822×10^{-3}		
		0.9884		0.0926		2.1853×10^{-3}	
-1	-0.9807		-0.0208		2.9921×10^{2}		3.2113×10^{-3}
		0.9467		0.0624		2.5707×10^{-3}	
0	-0.0339		0.2291		4.7916×10^{-3}		
		1.6342		0.3020			
2	3.2345		1.4374				
		3.0717					
3	6.3063						

We get $y_2(a) = p(\beta) = 0.08576546$

Hence, increasing the number of estimations increases the accuracy of the method.

7.Stability and Convergence Analysis of the Method:

It is clear that the developed method consists of two stages: stage of integration and stage of interpolation. Stability of the integration stage is well discussed by many authors and researchers such as Lambert (1974), Henrici (1962), Gear (1971) Conte and de Boor (1981) and others.

To control the induced instability, we have to use methods of integration whose finite difference equations are of the same order as the order of differential equation (for more detail see for example Khalaf (2000). To control the inherent instability we have to use multiple shooting or by using backward integration (for more detail see khalaf (2000)).

Another way for controlling the phenomena of instability of the numerical integration methods is improving the stability interval of the methods by using the principle of the fixed point iterations. This technique is used successfully by Khalaf and Abdulah (2000, 2001)) and then used by Khalaf and Mahmood (2001) for developing new methods for solving chaotic systems.

It is well known that (see Lambert (1974))

Stability + consistent $\longrightarrow$ convergence of the integration method, hence we can control the convergence character of the integration stage very easily.

We have no stability problem in the stage of interpolation. Hence there is no convergence problem in this stage. We only need the interpolation to be as accurate as possible by increasing the number of estimations and using higher order polynomials to interpolate the unknown boundary condition, i.e. we need in interpolation stage to reduce the error of interpolation. This can be done easily because the error of the interpolation at tlie point $\bar{x}$ is given by Conte and de Boor (1981):

$$e_n(\bar{x}) = f(\bar{x}) - p_o(\bar{x}) \; \frac{f^{(n+1)}(\zeta)}{(n+1)!} \prod_{j=0}^{n} (\bar{x} - x_j)$$

Where f(x) is a real-valued function defined on [a,b], differentiable (n+1) times on (a,b), $P_n(x)$ a polynomial interpolates f(x) at (n+1) distinct points x_O, x_1,...,x_n in [a,b], $\zeta = \zeta(\bar{x}) \in$ (a,b) and $\bar{x} \in$ (a,b). It is clear from the form of $e_n(\bar{x})$ that the error of interpolation decreases as the number of estimations increases. Hence, we can increase the accuracy of the method by increasing the number of estimations and using accurate convergent integration method. Now we can state the following:

Proposition: The new method is convergent if the integration stage is convergent and the interpolation stage is accurate.

8- Treatment of Stiff Boundary Value Problem by the New Methods:

Let us consider the following Boundary value Problem:

$y'' + l00ly' + l000y = 0$, $y(0) = 1$, $y(l) = e^{-1}$

Whose general solution is

$y(x)=Ae^{-x} + Be^{-1000x}$

Hence the exact solution is: $y(x)=e^{-x}$ The problem can be rewritten as first-order system

$y_1' = y_2$ $\quad y_1(0)$

$=-1001y_2-1000y_1$ $\quad y_1(l)=e^{-1}\, y_2'$

Solving the above system by Runge-Kutta of order4, we have to use step-size h<0.0028, i.e. h is for stability reasons restricted by the most rapidly changing component of the solution, namely e^{-1000x} The standard multistep methods would similarly restrict the step h.

Use implicit methods such as trapezoidal method, whose region of stability is the entire negative half-plane, so that h is unrestricted by the stability requirement (see Gear (1971)). But solving the problem by trapezoidal method will lead to a system of algebraic equations which can be solved by fixed-point iterations or by Newtonian iteration, which increases the complexity of the method as well as these two techniques converge under hard conditions. So to be out of the new problem we can modify the explicit Runge-Kutta method of order 4, which that we can use a larger h. The standard Runge-Kutta of order 4 is:

$y_{n+1} = \phi(x_n, y_n, h)$

which is the fixed-point formula:

$y = \phi(x, y, h)$

We can write the above formula as:

$> 0\ y = \dfrac{\phi(x, y, h) + \alpha h}{(1+\alpha)}, \alpha$

The new iteration formula of Runge-Kutta becomes:

Using this new formula for solving the above stiff problem we require h to be 1000h<2.8(l+a). In case a=0.9, h<0.00532, which is about two times larger than that used by standard Ruuge-Kutta. Any how using oc»l, will change the convergence requirement of the method (for more detail the reader can consult the following references Khalaf and Abudullah (2000), (2001)).

9- Conclusions:

We have developed a new parallel integration algorithm for solving boundary value problems suitable for MIMD computing systems. We showed how the results of the method can be improved by increasing the number of estimations and using backward parallel integrations.

The stability and error control of the method are well analysed. The treatment of the stiff boundary value problems is considered and we have developed explicit integration algorithms for solving these stiff boundary value problems.

REFERENCES

[1] AL-Murshid H. (2000) An investigation of parallel numerical algorithms for solving stiff ODEs suitable for parallel computers, Ph.D. Thesis, Mosul University .

[2] Al-Wajih K. (1999) , parallel algorithms for solving Unconstraint Optimization problems, M.Sc. Thesis , Mosul University.

[3] Ascher U., R. Mattheij and R.Russel (1988) Numerical Solution of BVPs for ODEs, Prentice-Hall, New Jersey.

[4] Brocklehurst E. (1992) Parallel processing, NPL News, No. 372,pp.24-27.

[5] Cash J. (1996) Runge - Kutta methods for the solution of stiff two-point BVPs , Appalied Numerical Mathematics , Vol22 , pp.165-177.

[6] Conte, S.D and C. de Boor (1981); Elementary Numerical Analysis;An Algorithmic Approach, Me Graw - Hill, London.

[7] Flynn M. (1972) , Some computer organizations and their effectiveness, IEEE Trans . On Computers, Vol.C-21, No.9 pp.948-960.

[8] Gear, C.W. (1971); Numerical: Initial Value Problems in Ordinary Differential Equation, Prentice - Hall.

[9] Henrici, P.K. (1962); Discrete Variable Methods for Ordinary Differential Equations. John Wiley, New York.

[10] Keller H. (1968) , Numerical methods for two-point BVPs, Blaisdell, Mass.

[11] Khalaf B. (1988) .Boundary value Techniques in continues system simulation , M. Phil Thesis, Bradford University .

[12] Khalaf B. (1990) , Parallel numerical algorithm for solving ODEs,Ph.D. Thesis, Leeds University.

[13] Khalaf B. (1995), A classification for computer architectures J.Educ. Sci, Vol. (24), pp. 212-222.

[14] Khalaf B. (1997) Parallel collocation for BVPs in ODEs, J, Educ. Sci., Vol. 27, pp. 135-146.

[15] Khalaf B. (2000) Techniques for controlling the stability of the numerical solution of initial value, Raf.Jour. sci. vol 11, No.l,pp 95-106.

[16] Khalaf B. and H. Al-Murshid (2000), Effectiveness of parallel shooting for solving stiff ODEs; J. Educ. Sci., To appear.

[17] Khalaf B.M.S. and A.Al-Sawoor (2002), Improving the parallel integration and interpolation Method; Raf. Jour. Sci, To appear.

[18] Khalaf B. and K. Al-Wajih (2000), Parallel shooting for unconstraint optimization, J. Al-Rafidin for Science, To appear.

[19] Khalaf and G.M. Abdullah; (2000) Improving the stability Region of Explicit Methods for IVPs; Raf. Jour. Sci. Vol. 11, vol2, pp82-100.

[20] Khalaf, B.M.S and G.M. Abdulah, (2001) Improving the stability Region of Implicit Numerical Methods for IVPs, Raf. Jour. Sci. Vol. 12,No3,pp.76-84.

[21] Khalaf B.M.S. and E.M. Mahmood (2002) Explicit Methods for solving chaotic system, Raf. Jour. Sci, To appear.

[22] Khalaf B.M.S. and M. Matti (2001) Parallel integration and interpolation method for BVPs in ODEs, J. Edcu. Sci, To Appear.

[23] Lambert J.D (1974); Computational Methods in ordinary differential equations, John Wiley and Sons Ltd.

[24] Meiko (1989), A programmer introduction to Sun-CSTools, Meiko LTD, England.