PERGAMON Computers and Mathematics with Applications 42 (2001) 1411–1420

An International Journal
computers & mathematics with applications

www.elsevier.nl/locate/camwa

A Nonlinear Shooting Method for Two-Point Boundary Value Problems

SUNG N. HA
Department of Mathematics, KyungHee University
Suwon, Kyunggi-do, 449-701, Korea

(Received October 2000; accepted November 2000)

Abstract—We study a new nonlinear shooting method for solving two-point boundary value problems and show numerical experiments with various initial velocity conditions. We discuss and analyze the numerical solutions which are obtained by the shooting method. © 2001 Elsevier Science Ltd. All rights reserved.

Keywords—Self-adjoint, Shooting method, Initial velocity, Critical value, Generalized Newton's method.

1. INTRODUCTION

Two-point boundary value problems occur in applied mathematics, theoretical physics, engineering, control and optimization theory. If the two-point boundary value problem cannot be solved analytically, which is the usual case, then recourse must be made to numerical methods.

Shooting methods take their name from the situation in the two-point boundary value problem for a single second-order differential equation with initial and final values of the solution prescribed. Varying the initial slope gives rise to a set of profiles which suggest the trajectory of a projectile "shot" from the initial point. That initial slope is sought which results in the trajectory "hitting" the target, that is, the final value [1].

Shooting methods require a minimum of problem analysis and preparation. For nonlinear differential equations, shooting methods have certain advantages for the problem solver. The shooting methods are quite general and are applicable to a wide variety of differential equations. It is not necessary for applicability of shooting methods that the equations be of special types such as even-order self-adjoint.

2. TWO-POINT BOUNDARY VALUE PROBLEM STATEMENT

Two-point boundary value problems are problems in which, for a set of possibly nonlinear ordinary differential equations, some boundary conditions are specified at the initial value of independent variable, while the remainder of boundary conditions are specified at the terminal value of the independent variable. The boundary conditions are therefore split between the two points, the initial and terminal values of the independent variable [2].

0898-1221/01/$ - see front matter Typeset by $\mathcal{A}\mathcal{M}\mathcal{S}$-TEX
PII: S0898-1221(01)00250-4

In general, by two-point boundary value problems, we mean problems with the following characteristics:

1. n first-order ordinary differential equations to be solved over the interval $[a, b]$, where a is the initial point and b is the final point;
2. r boundary conditions are specified at the initial value of the independent variable;
3. $(n-r)$ boundary conditions are specified at the terminal value of the independent variable.

Problems originally expressed as higher-order nonlinear ordinary differential equations can be reduced to a system of first-order nonlinear ordinary differential equations. But we will concentrate on a single second-order differential equation with boundary conditions as in the following type of problem:

$$y'' = f(x, y, y'), \qquad a \le x \le b, \quad y(a) = \alpha, \quad y(b) = \beta.$$

The following theorem gives general conditions that ensure that the solution to a second-order boundary value problem will exist and be unique [3].

THEOREM 1. *Suppose the function f in the boundary value problem*

$$y'' = f(x, y, y'), \qquad a \le x \le b, \quad y(a) = \alpha, \quad y(b) = \beta,$$

is continuous on the set

$$D = \{(x, y, y') \mid a \le x \le b, \ -\infty < y' < \infty\},$$

and that $\frac{\partial f}{\partial y}$ and $\frac{\partial f}{\partial y'}$ are also continuous on D. If

(1) $\frac{\partial f}{\partial y}(x, y, y') > 0$ *for all* $(x, y, y') \in D$, *and*

(2) $|\frac{\partial f}{\partial y'}(x, y, y')| < M$ *for all* $(x, y, y') \in D$,

then the boundary value problem has a unique solution.

PROOF. See [4].

3. A NEW SHOOTING METHOD FOR NONLINEAR BOUNDARY VALUE PROBLEMS

Since we are primarily interested in shooting techniques, we want to characterize a new approach to two-point boundary value problems.

The new shooting method for the nonlinear second-order boundary value problem

$$y'' = f(x, y, y'), \qquad a \le x \le b, \quad y(a) = \alpha, \quad y(b) = \beta, \tag{1}$$

is similar to the linear case as using initial-value problem solver. We will need to use the solutions to a *sequence* of initial-value problems of the form

$$w'' = f(x, w, w'), \qquad a \le x \le b, \quad w(a) = \alpha, \quad w'(a) = v, \tag{2}$$

involving a parameter v, to approximate the solution to our boundary value problem.

We do this by choosing the parameters $v = v_k$ in a manner that will ensure that

$$\lim_{k \to \infty} w(b, v_k) = y(b) = \beta, \tag{3}$$

where $w(x, v_k)$ denotes the solution to the initial-value problem (2) with $v = v_k$, and $y(x)$ denotes the solution to the boundary value problem (1).

We start with a parameter v_0 that determines the initial elevation at which the object is fired from the point (a, α) and along the curve described by the solution to the initial-value problem

$$w'' = f(x, w, w'), \qquad a \le x \le b, \quad w(a) = \alpha, \quad w'(a) = v_0. \tag{4}$$

If $w(b, v_0)$ is not sufficiently close to β, we attempt to correct our approximation by choosing another elevation v_1 and so on, until $w(b, v_k)$ is sufficiently close to "hitting" β.

To determine how the parameters v_k can be chosen, suppose a boundary value problem of the form (1) has a unique solution. If $w(x, v)$ is used to denote the solution to the initial problem (2), the problem is to determine v so that

$$w(b, v) - \beta = 0. \tag{5}$$

Since this is a nonlinear equation, we employ Newton's method to solve the problem. We need to choose initial approximation v_0 and then generate the sequence by

$$v_k = v_{k-1} - \frac{(w(b, v_{k-1}) - \beta)}{\left(\frac{dw}{dv}\right)(b, v_{k-1})}, \qquad \text{where } \left(\frac{dw}{dv}\right)(b, v_{k-1}) = \frac{dw}{dv}(b, v_{k-1}), \tag{6}$$

and requires the knowledge of $\left(\frac{dw}{dt}\right)(b, v_{k-1})$. This presents a difficulty, since an explicit representation for $w(b, v)$ is not known; we know only the values $w(b, v_0), w(b, v_1), w(b, v_2), \ldots, w(b, v_{k-1})$.

Suppose we rewrite the initial-value problem (2), emphasizing that the solution depends on both x and v

$$w''(x, v) = f(x, w(x, v), w'(x, v)), \qquad a \le x \le b, \quad w(a, v) = \alpha, \quad w'(a, v) = v, \tag{7}$$

retaining the prime notation to indicate differentiation with respect to x. Since we are interested in determining $\left(\frac{dw}{dv}\right)(b, v)$ when $v = v_{k-1}$, we take the partial derivative of (7) with respect to v. This implies that

$$\begin{aligned}
\frac{\partial w''}{\partial v}(x, v) &= \frac{\partial f}{\partial v}(x, w(x, v), w'(x, v)) \\
&= \frac{\partial f}{\partial x}(x, w(x, v), w'(x, v))\frac{\partial x}{\partial v} + \frac{\partial f}{\partial w}(x, w(x, v), w'(x, v))\frac{\partial w}{\partial v}(x, v) \\
&\quad + \frac{\partial f}{\partial w'}(x, w(x, v), w'(x, v))\frac{\partial w'}{\partial v}(x, v),
\end{aligned}$$

or, since x and v are independent,

$$\frac{\partial w''}{\partial v}(x, v) = \frac{\partial f}{\partial w}(x, w(x, v), w'(x, v))\frac{\partial w}{\partial v}(x, v) + \frac{\partial f}{\partial w'}(x, w(x, v), y'(x, v))\frac{\partial w'}{\partial v}(x, v), \tag{8}$$

for $a \le x \le b$. The initial conditions give

$$\frac{\partial w}{\partial v}(a, v) = 0 \qquad \text{and} \qquad \frac{\partial w'}{\partial v}(a, v) = 1.$$

If we simplify the notation by using $z(x, v)$ to denote $\left(\frac{\partial w}{\partial v}\right)(x, v)$ and assume that the order of differentiation of x and v can be reversed, (8) becomes the initial-value problem

$$\ddot{z} = \frac{\partial f}{\partial w}(x, w, w')z + \frac{\partial f}{\partial w'}(x, w, w')\dot{z}, \qquad a \le x \le b, \quad z(a) = 0, \quad \dot{z}(a) = 1. \tag{9}$$

Therefore, one requires that two initial-value problems be solved for each iteration, (2) and (9). Then from (6),

$$v_k = v_{k-1} - \frac{w(b, v_{k-1}) - \beta}{z(b, v_{k-1})}. \tag{10}$$

In practice, none of these initial-value problems is likely to be solved exactly; instead, the solutions are approximated by one of initial-value problem solvers.

4. SOME NUMERICAL EXAMPLES

EXAMPLE 1. Consider the boundary value problem

$$y''(x) = y^2(x) + 2\pi^2 \cos(2\pi x) - \sin^4(\pi x), \qquad \text{for } 0 < x < 1,$$

with the boundary conditions

$$y(0) = 0, \qquad y(1) = 0.$$

The problem has the exact solution $y(x) = \sin^2(\pi x)$.

For the numerical solution, we chose $h = 0.05$ and tried initial velocities $v_0 = 0.25, 0.50, 1.0, 5.0, 10.0$. We used the fourth-order Runge-Kutta method and Newton's method with error bound 10^{-7}. Table 1 shows approximated errors with various initial velocities, and we have converging final initial velocity $v_k = 0.0000008$ for each initial velocity. Table 2 shows iteration procedures for each initial velocity.

Table 1. Approximated errors with each initial velocity in Example 1.

x	$v_0 = 0.25$	$v_0 = 0.50$	$v_0 = 1.0$	$v_0 = 5.0$	$v_0 = 10.0$
0.00	0.0000000	0.0000000	0.0000000	0.0000000	0.0000000
0.05	0.0000002	0.0000002	0.0000002	0.0000002	0.0000002
0.10	0.0000008	0.0000008	0.0000008	0.0000007	0.0000008
0.15	0.0000017	0.0000017	0.0000017	0.0000017	0.0000017
0.20	0.0000028	0.0000028	0.0000028	0.0000028	0.0000028
0.25	0.0000041	0.0000041	0.0000041	0.0000041	0.0000041
0.30	0.0000054	0.0000054	0.0000054	0.0000054	0.0000054
0.35	0.0000066	0.0000066	0.0000066	0.0000066	0.0000066
0.40	0.0000075	0.0000075	0.0000075	0.0000075	0.0000075
0.45	0.0000081	0.0000081	0.0000081	0.0000081	0.0000081
0.50	0.0000083	0.0000083	0.0000083	0.0000083	0.0000083
0.55	0.0000082	0.0000082	0.0000082	0.0000082	0.0000082
0.60	0.0000075	0.0000075	0.0000075	0.0000075	0.0000075
0.65	0.0000066	0.0000066	0.0000066	0.0000066	0.0000066
0.70	0.0000054	0.0000054	0.0000054	0.0000054	0.0000054
0.75	0.0000040	0.0000040	0.0000040	0.0000040	0.0000040
0.80	0.0000027	0.0000027	0.0000027	0.0000027	0.0000027
0.85	0.0000015	0.0000015	0.0000015	0.0000015	0.0000015
0.90	0.0000006	0.0000006	0.0000006	0.0000006	0.0000006
0.95	0.0000002	0.0000002	0.0000002	0.0000002	0.0000002
1.00	0.0000001	0.0000001	0.0000001	0.0000001	0.0000001

Table 2. The iterations for calculating v_k with various v_0 in Example 1.

k	$v_0 = 0.25$	$v_0 = 0.50$	$v_0 = 1.0$	$v_0 = 5.0$	$v_0 = 10.0$
1	0.2500000	0.5000000	1.0000000	5.0000000	10.0000000
2	0.0043503	0.0174880	0.0685602	1.4053063	4.6188340
3	−0.0000008	0.0000108	0.0002951	0.1326154	1.2200974
4	0.0000010	0.0000010	0.0000008	0.0011884	0.1009321
5	0.0000008	0.0000008	0.0000000	0.0000000	0.0006724
6	0.0000000	0.0000000	0.0000000	0.0000011	0.0000003
7	0.0000000	0.0000000	0.0000000	0.0000009	0.0000008

Table 3. Approximated errors with each initial velocity in Example 2.

x	$v_0 = 0.25$	$v_0 = 0.50$	$v_0 = 1.0$	$v_0 = 5.0$	$v_0 = 10.0$
1.00	0.0000000	0.0000000	0.0000000	0.0000000	0.0000000
1.10	0.0000010	0.0000010	0.0000010	0.0000010	0.0000010
1.20	0.0000020	0.0000020	0.0000010	0.0000020	0.0000020
1.30	0.0000020	0.0000020	0.0000010	0.0000020	0.0000020
1.40	0.0000030	0.0000030	0.0000010	0.0000030	0.0000030
1.50	0.0000020	0.0000020	0.0000000	0.0000020	0.0000020
1.60	0.0000010	0.0000010	−0.0000010	0.0000010	0.0000010
1.70	0.0000010	0.0000010	−0.0000010	0.0000010	0.0000010
1.80	0.0000010	0.0000010	−0.0000010	0.0000010	0.0000010
1.90	0.0000010	0.0000010	−0.0000010	0.0000010	0.0000010
2.00	0.0000000	0.0000000	−0.0000020	0.0000000	0.0000000
2.10	0.0000000	0.0000000	−0.0000020	0.0000000	0.0000000
2.20	−0.0000020	−0.0000020	−0.0000050	−0.0000020	−0.0000020
2.30	−0.0000020	−0.0000020	−0.0000050	−0.0000020	−0.0000020
2.40	−0.0000040	−0.0000040	−0.0000070	−0.0000040	−0.0000040
2.50	−0.0000050	−0.0000050	−0.0000080	−0.0000050	−0.0000050
2.60	−0.0000060	−0.0000060	−0.0000090	−0.0000060	−0.0000060
2.70	−0.0000090	−0.0000090	−0.0000120	−0.0000090	−0.0000090
2.80	−0.0000090	−0.0000090	−0.0000120	−0.0000090	−0.0000090
2.90	−0.0000100	−0.0000100	−0.0000130	−0.0000100	−0.0000100
3.00	−0.0000110	−0.0000110	−0.0000140	−0.0000110	−0.0000110

Table 4. The iterations for calculating v_k with various v_0 in Example 2.

k	$v_0 = 0.25$	$v_0 = 0.50$	$v_0 = 1.0$	$v_0 = 5.0$	$v_0 = 10.0$
1	0.2500000	0.5000000	1.0000000	5.0000000	10.0000000
2	−25.5091438	−25.7687836	−26.2918949	−30.6432896	−36.4224434
3	−15.3420372	−15.5771437	−16.0841179	−21.9890518	−32.9111176
4	−13.4423866	−13.3590727	−13.1948881	−13.1899252	−26.0706615
5	−14.2701044	−14.3122711	−14.3967123	−14.3993130	−15.8642664
6	−13.8769932	−13.8582811	−13.8211451	−13.8200245	−13.2634506
7	−14.0577354	−14.0665798	−14.0842857	−14.0848227	−14.3612375
8	−13.9732618	−13.9692125	−13.9610624	−13.9608164	−13.8366861
9	−14.0124502	−14.0143347	−14.0181427	−14.0182686	−14.0768642
10	−13.9942102	−13.9933300	−13.9915771	−13.9915133	−13.9644613
11	−14.0026922	−14.0030909	−14.0039244	−14.0039463	−14.0165644
12	−13.9987316	−13.9985657	−13.9981709	−13.9981642	−13.9923048
13	−14.0005941	−14.0006542	−14.0008345	−14.0008392	−14.0035868
14	−13.9997225	−13.9996872	−13.9995966	−13.9995918	−13.9983177
15	−14.0001192	−14.0001383	−14.0002003	−14.0001974	−14.0007696
16	−13.9999361	−13.9999123	−13.9998941	−13.9998932	−13.9996414
17	−14.0000381	−14.0000496	−14.0000582	−14.0000591	−14.0001764
18	−13.9999743	−13.9999733	−13.9999657	−13.9999638	−13.9999104
19	−14.0000219	−14.0000210	−14.0000181	−14.0000210	−14.0000477
20	−13.9999962	−13.9999952	−13.9999924	−13.9999952	−13.9999743
21	0.0000000	0.0000000	0.0000000	0.0000000	−14.0000219
22	0.0000000	0.0000000	0.0000000	0.0000000	−13.9999962

EXAMPLE 2. The boundary value problem

$$y''(x) = \frac{(32 + 2x^3 - yy')}{8}, \qquad \text{for } 1 < x < 3,$$

Table 5. The iterations for calculating v_k with various v_0 in Example 3.

k	$v_0 = 0.25$	$v_0 = 0.50$	$v_0 = 1.0$	$v_0 = 5.0$	$v_0 = 10.0$
1	0.2500000	0.5000000	1.0000000	5.0000000	10.0000000
2	−2.9779999	−2.7393000	−2.2535999	1.8135999	−3.7744999
3	−5.7396998	−5.5599999	−5.1805000	−1.4428000	−6.3042998
4	−7.4190998	−7.3365002	−7.1479998	−4.5093002	−7.6483002
5	−7.9450002	−7.9310999	0.0000000	−6.7719002	−7.9759002
6	0.0000000	0.0000000	0.0000000	−7.8002000	−7.9993000
7	0.0000000	0.0000000	0.0000000	−7.9899001	−8.0000000

Table 6. Approximated errors with each initial velocity in Example 3.

x	$v_0 = 0.25$	$v_0 = 0.50$	$v_0 = 1.0$	$v_0 = 5.0$	$v_0 = 10.0$
0.00	0.0000000	0.0000000	0.0000000	0.0000000	0.0000000
0.05	−0.0000020	−0.0000020	−0.0000020	−0.0000020	−0.0000020
0.10	−0.0000030	−0.0000030	−0.0000030	−0.0000030	−0.0000030
0.15	−0.0000040	−0.0000040	−0.0000040	−0.0000030	−0.0000030
0.20	−0.0000050	−0.0000040	−0.0000050	−0.0000040	−0.0000040
0.25	−0.0000050	−0.0000040	−0.0000050	−0.0000040	−0.0000040
0.30	−0.0000050	−0.0000040	−0.0000050	−0.0000030	−0.0000030
0.35	−0.0000050	−0.0000040	−0.0000050	−0.0000030	−0.0000030
0.40	−0.0000050	−0.0000040	−0.0000050	−0.0000030	−0.0000030
0.45	−0.0000050	−0.0000040	−0.0000050	−0.0000030	−0.0000030
0.50	−0.0000050	−0.0000040	−0.0000050	−0.0000020	−0.0000020
0.55	−0.0000050	−0.0000030	−0.0000050	−0.0000020	−0.0000020
0.60	−0.0000050	−0.0000030	−0.0000050	−0.0000020	−0.0000020
0.65	−0.0000050	−0.0000030	−0.0000050	−0.0000010	−0.0000010
0.70	−0.0000050	−0.0000030	−0.0000060	−0.0000010	−0.0000010
0.75	−0.0000050	−0.0000030	−0.0000060	0.0000000	0.0000000
0.80	−0.0000050	−0.0000030	−0.0000060	0.0000000	0.0000000
0.85	−0.0000050	−0.0000030	−0.0000060	0.0000000	0.0000000
0.90	−0.0000050	−0.0000030	−0.0000070	0.0000000	0.0000000
0.95	−0.0000060	−0.0000030	−0.0000070	0.0000010	0.0000010
1.00	−0.0000060	−0.0000030	−0.0000070	0.0000010	0.0000010

with the boundary conditions

$$y(1) = 17, \qquad y(3) = \frac{43}{3}.$$

The problem has exact solution $y(x) = x^2 + 16/x$.

For the numerical solution, we chose $h = 0.01$ and tried initial velocities $v_0 = 0.25, 0.50, 1.0, 5.0, 10.0$. Using the same manner of Example 1, we always had converging final initial velocity $v_k = -13.99999$. In this example, we used error bound 10^{-5}. Table 3 shows approximated errors of each initial velocity. We calculated at 201 node points, but we show the errors for 21 node points in Table 3. We found that the midpoint of the x-interval has a smaller error than the endpoint. Table 4 shows the procedure for calculating v_k. It shows that it took more iteration times than in Example 1.

EXAMPLE 3. Consider the boundary value problem

$$y''(x) = \frac{3}{2}y^2, \qquad \text{for } 0 < x < 1,$$

with the boundary conditions

$$y(0) = 4, \qquad y(1) = 1.$$

The problem has exact solution $y(x) = 4/(1+x)^2$.

Table 7. Using $v_0 = 0.6975$ and $\omega = 1.24899$ in Example 4.

$x(i)$	Exact Solution	Approx. Solution	Error
1.00	2.0000000	2.0000000	0.0000000
1.05	2.0023808	2.0020661	0.0003147
1.10	2.0090909	2.0084186	0.0006723
1.15	2.0195651	2.0185144	0.0010507
1.20	2.0333333	2.0318995	0.0014338
1.25	2.0500000	2.0481904	0.0018096
1.30	2.0692306	2.0670624	0.0021682
1.35	2.0907407	2.0882378	0.0025029
1.40	2.1142855	2.1114786	0.0028069
1.45	2.1396549	2.1365800	0.0030749
1.50	2.1666665	2.1633656	0.0033009
1.55	2.1951609	2.1916831	0.0034778
1.60	2.2249997	2.2214017	0.0035980
1.65	2.2560601	2.2524097	0.0036504
1.70	2.2882349	2.2846136	0.0036213
1.75	2.3214281	2.3179369	0.0034912
1.80	2.3555551	2.3523207	0.0032344
1.85	2.3905399	2.3877246	0.0028152
1.90	2.4263151	2.4241302	0.0021849
1.95	2.4628198	2.4615438	0.0012760
2.00	2.4999993	2.5000041	−0.0000048

The final v_k is -0.00606 and iteration number is 7996.

Table 8. Numbers of iteration for calculating v_k with various ω.

	$v_0 = 0.6975$
$\omega = 0.50$	19949
$\omega = 0.75$	13311
$\omega = 1.00$	9982
$\omega = 1.15$	8734
$\omega = 1.20$	8324
$\omega = 1.24899$	7996
$\omega = 1.25$	exceed 100000

	$v_0 = 0.6980$
$\omega = 0.50$	21146
$\omega = 0.75$	14073
$\omega = 1.00$	10654
$\omega = 1.15$	9187
$\omega = 1.20$	8993
$\omega = 1.195$	8845
$\omega = 1.20$	exceed 100000

For the numerical solution, we chose $h = 0.05$ and tried initial velocities $v_0 = 0.25, 0.50, 1.0, 5.0, 10.0$. We had converging final initial velocity value as $v_k = -7.99$, as shown in Table 5. Table 6 shows that the larger initial velocities have more accurate approximated values. In this case, there is not a unique solution [5]; the numerical solution converges always to a nontrivial solution.

EXAMPLE 4. Consider the boundary value problem

$$y''(x) = 2y^3 - 6y - 2x^3, \qquad \text{for } 1 < x < 2,$$

with the boundary conditions

$$y(1) = 2, \qquad y(2) = \frac{5}{2}.$$

The problem has exact solution $y(x) = x + 1/x$.

For the numerical solution, we chose $h = 0.05$ and tried initial velocities $v_0 = 0.25$ and $v_0 = 0.50, 1.0, 5.0, 10.0$. We have converging final initial velocity $v_k = -0.00606$ and good numerical results. We could not get suitable numerical solutions when we tried initial velocities with $v_0 = 1.0, 5.0, 10.0$. In this example, we found that it took many iterations to find v_k for convergence

of numerical solutions near $v_0 = 0.7$. It took 9982 iterations when $v_0 = 0.6975$, and it took more than 10000 iterations when $v_0 = 0.6980$. In this example, we used the generalized Newton's method [4]. It is more effective than Newton's method whenever one can find a suitable relaxation parameter ω. Table 7 shows a good numerical result, and Table 8 shows iteration numbers to find suitable v_k corresponding to each relaxation parameter ω for each initial velocity. We used error bound 10^{-5}.

EXAMPLE 5. Consider the boundary value problem

$$y''(x) = y^3 - yy', \qquad \text{for } 1 < x < 2,$$

with the boundary conditions

$$y(1) = \frac{1}{2}, \qquad y(2) = \frac{1}{3}.$$

The problem has exact solution $y(x) = 1/(x+1)$.

For the numerical solutions, we chose $h = 0.05$ and tried initial velocities $v_0 = 0.25, 0.50, 1.00$, and 3.988. We have converging final initial velocity $v_k = -0.24935$. At first, we found a number near $v_0 = 3.99$ by using the PC (Dell; mpx Pentium III). We wanted to get more accurate results so we ran the programs again by using VAX. We got the same result with PC and VAX. That means, we have the same converging initial velocity $v_0 = -2.4935$, but the critical value is a little different between the PC and VAX. In this example, we ran programs using single precision and double precision by generalized Newton's method on VAX. Table 9 shows the errors by using single precision programs with various initial velocities on VAX. Table 10 shows the errors by using double precision programs with various initial velocities on VAX.

In this example, we found that a number exists between 4.15 and 4.20 which we would like to call the *critical value*. Using a number which is smaller than the critical value as an initial velocity v_0, then numerical solutions converge to an exact solution. And, using a number which is larger than the critical value as an initial velocity v_0, then numerical solutions do not converge to an exact solution.

Table 9. Approximated errors with each initial velocity in Example 5.

x	$v_0 = 3.99$	$v_0 = 4.00$	$v_0 = 4.10$	$v_0 = 4.15$	$v_0 = 4.175$
1.00	0.0000000	0.0000000	0.0000000	0.0000000	0.0000000
1.05	−0.0000300	−0.0000300	−0.0000300	−0.0000300	−0.0000300
1.10	−0.0000560	−0.0000550	−0.0000540	−0.0000540	−0.0000560
1.15	−0.0000760	−0.0000760	−0.0000730	−0.0000730	−0.0000760
1.20	−0.0000910	−0.0000910	−0.0000880	−0.0000880	−0.0000920
1.25	−0.0001030	−0.0001030	−0.0001000	−0.0000990	−0.0001030
1.30	−0.0001110	−0.0001110	−0.0001070	−0.0001070	−0.0001120
1.35	−0.0001170	−0.0001160	−0.0001110	−0.0001110	−0.0001170
1.40	−0.0001190	−0.0001180	−0.0001130	−0.0001120	−0.0001190
1.45	−0.0001190	−0.0001180	−0.0001120	−0.0001110	−0.0001190
1.50	−0.0001160	−0.0001160	−0.0001090	−0.0001080	−0.0001170
1.55	−0.0001120	−0.0001110	−0.0001040	−0.0001030	−0.0001120
1.60	−0.0001060	−0.0001050	−0.0000970	−0.0000960	−0.0001060
1.65	−0.0000980	−0.0000970	−0.0000890	−0.0000870	−0.0000980
1.70	−0.0000880	−0.0000880	−0.0000780	−0.0000770	−0.0000890
1.75	−0.0000780	−0.0000770	−0.0000670	−0.0000660	−0.0000780
1.80	−0.0000660	−0.0000650	−0.0000540	−0.0000530	−0.0000660
1.85	−0.0000520	−0.0000520	−0.0000410	−0.0000390	−0.0000530
1.90	−0.0000380	−0.0000370	−0.0000260	−0.0000240	−0.0000390
1.95	−0.0000230	−0.0000220	−0.0000100	−0.0000080	−0.0000240
2.00	−0.0000070	−0.0000060	0.0000060	0.0000080	−0.0000080

Table 10. Approximated errors with each initial velocity in Example 5.

x	$v_0 = 3.99$	$v_0 = 4.00$	$v_0 = 4.10$	$v_0 = 4.15$	$v_0 = 4.175$
1.00	0.0000000	0.0000000	0.0000000	0.0000000	0.0000000
1.05	−0.0000300	−0.0000300	−0.0000300	−0.0000300	−0.0000300
1.10	−0.0000560	−0.0000550	−0.0000540	−0.0000540	−0.0000540
1.15	−0.0000760	−0.0000760	−0.0000740	−0.0000730	−0.0000730
1.20	−0.0000910	−0.0000910	−0.0000880	−0.0000880	−0.0000880
1.25	−0.0001030	−0.0001030	−0.0001000	−0.0000990	−0.0000990
1.30	−0.0001110	−0.0001110	−0.0001070	−0.0001070	−0.0001070
1.35	−0.0001170	−0.0001160	−0.0001120	−0.0001110	−0.0001110
1.40	−0.0001190	−0.0001180	−0.0001130	−0.0001130	−0.0001130
1.45	−0.0001190	−0.0001180	−0.0001120	−0.0001120	−0.0001120
1.50	−0.0001160	−0.0001160	−0.0001090	−0.0001080	−0.0001090
1.55	−0.0001120	−0.0001110	−0.0001040	−0.0001030	−0.0001030
1.60	−0.0001060	−0.0001050	−0.0000970	−0.0000960	−0.0000960
1.65	−0.0000980	−0.0000970	−0.0000890	−0.0000880	−0.0000880
1.70	−0.0000880	−0.0000880	−0.0000790	−0.0000770	−0.0000780
1.75	−0.0000780	−0.0000770	−0.0000670	−0.0000660	−0.0000660
1.80	−0.0000660	−0.0000650	−0.0000540	−0.0000530	−0.0000530
1.85	−0.0000520	−0.0000520	−0.0000410	−0.0000390	−0.0000400
1.90	−0.0000380	−0.0000370	−0.0000260	−0.0000240	−0.0000250
1.95	−0.0000230	−0.0000220	−0.0000100	−0.0000080	−0.0000090
2.00	−0.0000070	−0.0000060	0.0000060	0.0000070	0.0000070

Table 11. A numerical result by using single precision and double precision on VAX.

v_0	Single Precision	Double Precision
$v_0 = 3.99$	Converge	Converge
$v_0 = 4.10$	Converge	Converge
$v_0 = 4.15$	Converge	Converge
$v_0 = 4.20$	Diverge	Diverge

Table 11 shows numerical results by using single precision and double precision with various initial velocities near the critical value.

4. REMARKS

With regard to other numerical methods, note the following. Ritz's method [6] is useful in linear boundary value problems, but it has difficulty in nonlinear boundary value problems. In [7], using Green's function, boundary value problems are changed to an integral equation and iteration is applied. Then one must solve a large scale nonlinear system of equations. Keller introduces Gaussian quadrature, but this needs delicate conditions and is practically difficult.

With regard to our examples, note the following. In Examples 1 and 2, we show that our numerical solutions with any choice of initial velocities converge to an exact solution. And, in Examples 4 and 5, we show that we obtained numerical solutions with some initial velocities converging to an exact solution, but our numerical solution with some special choice of initial velocities did not converge to an exact solution. Example 3 shows numerical solutions converging to a nontrivial solution of the problem even though it does not have a unique solution. Finally, we found there exists some special initial velocity value v_0. It cannot make converging suitable initial velocity v_k (v_0 located near by 0.698 in Example 4). Also, it cannot make converging numerical solutions (v_0 located at between 4.15 and 4.20 in Example 5).

For nonlinear differential equations, shooting methods have certain advantages for the problem solver. First of all, the methods are quite general and are applicable to a wide variety of differential equations. It is not necessary for applicability of shooting methods that the equations be of special types such as even-order self-adjoint. Second, shooting methods require a minimum of problem analysis and preparation.

Despite their advantages, shooting methods, like all methods, have their limitations. Shooting methods sometimes fail to converge for problems which are sensitive to the initial conditions. In some problems, modest changes in the initial conditions result in numerical difficulties such as machine overflow. We need to study these matters theoretically, and this is in progress. We ran programs in Examples 1–5 by using Visual FORTRAN 6.0 with DELL(mpx) Pentium III, and Example 5 was run again by VAX at UTA ACS.

REFERENCES

1. P.B. Bailey, L.F. Shampine and P.E. Waltman, *Nonlinear Two Point Boundary Value Problems*, Academic Press, New York, (1968).
2. S.M. Roberts and J.S. Shipman, *Two Point Boundary Value Problems: Shooting Methods*, American Elsevier, New York, (1972).
3. R.L. Burden and J.D. Faires, *Numerical Analysis*, PWS, Boston, (1993).
4. D. Greenspan and V. Casulli, *Numerical Analysis for Applied Mathematics, Science, and Engineering*, Addison-Wesley, (1988).
5. S.T. Pohozaev, The Dirichlet problem for the equation $\Delta u = u^2$, *Soviet Math.* **1**, 1143–1146 (1960).
6. L. Collatz, *The Numerical Treatment of Differential Equations*, Third Edition, Springer-Verlag, Heidelberg, (1960).
7. H.B. Keller, *Numerical Methods for Two Point Boundary Value Problems*, Blaisdell, London, (1968).
8. R.K. Nagle and E.B. Saff, *Fundamentals of Differential Equations and Boundary Value Problems*, Addison-Wesley, New York, (1993).
9. I. Stakgold, *Green's Functions and Boundary Value Problems*, John Wiley & Sons, New York, (1979).