ELSEVIER

Journal of Computational and Applied Mathematics 83 (1997) 195–203

JOURNAL OF COMPUTATIONAL AND APPLIED MATHEMATICS

An optimal algorithm for certain boundary value problem

Krystyna Styś, Tadeusz Styś *

Department of Mathematics, University of Botswana, Private Bag 0022, Gaborone, Botswana

Received 13 January 1997; received in revised form 20 April 1997

Abstract

The $O(h^4)$ finite-difference scheme for the second derivative $u''(x)$ leads to a coherent pentadiagonal matrix which is factorized into two tridiagonal matrices. This factorization is used to derive an optimal algorithm for solving a linear system of equations with the pentadiagonal matrix. As an application, a nonlinear system of ordinary differential equations is approximated by an $O(h^4)$ convergent finite-difference scheme. This scheme is solved by the implicit iterative method applying the algorithm at each iteration. A *Mathematica* module designed for the purpose of testing and using the method is attached.

Keywords: Optimal algorithm; Boundary problem; Sparse matrices

AMS classification: 65L

1. Introduction

Let us consider the following system of differential equations:

$$-\frac{d^2u}{dx^2} = g(x,u), \quad a \leqslant x \leqslant b, \tag{1}$$

with the boundary conditions

$$u(a) = u_0, \quad u(b) = u_{n+1},$$

where $u(x) = (u^{(1)}(x), u^{(2)}(x), \ldots, u^{(p)}(x))$ is the unknown function, and $g(x,u) = (g^{(1)}(x,u), g^{(2)}(x,u), \ldots, g^{(p)}(x,u))$, is given.

This boundary value problem has a unique regular solution if the function $g(x,u)$ is continuous and possesses partial derivatives, such that, the Jacobian matrix $J(g^{(1)}, g^{(2)}, \ldots, g^{(p)})$ is nonpositive

* Corresponding author.

PII S0377-0427(97)00098-8

definite, i.e.,

$$(J(g^{(1)}, g^{(2)}, \ldots, g^{(p)})\lambda, \lambda) \leqslant 0,$$

for all $\lambda = (\lambda_1, \lambda_2, \ldots, \lambda_p)$, $x \in (a, b)$ and $-\infty < u^{(q)} < \infty$, $q = 1, 2, \ldots, p$.

There is a broad literature on numerical solution of Eq. (1) with either initial value conditions or boundary value conditions. The higher-order and stable methods given in [2, 3, 5, 6] are applicable to Eq. (1) with initial conditions. When boundary conditions are present, shooting method is used to transform boundary conditions into initial conditions. It is, however, expensive in terms of arithmetic operations because the initial value $u'(a)$ is obtained only after solving an initial value problem. Nevertheless, one can use shooting method to refine $u'(a) \approx (1/h)(u(a+h) - u(a))$, where $u(a+h)$ can be computed by the algorithm given in this paper. A higher-order method (cf. [2, 3, 5, 6]) for solving initial value problems combined with the algorithm and shooting method gives an approximate solution of the boundary problem $O(h^p)$, $(p \geqslant 6)$ accurate.

We shall approximate the system of Eq. (1) by the following finite-difference scheme with the global error $O(h^4)$ (cf. [1, 7]):

$$\begin{aligned} &-L_h v_i = g(x_i, v_i), \quad i = 1, 2, \ldots, n, \\ &v_0 = u_0, \quad v_{n+1} = u_{n+1}, \end{aligned} \tag{2}$$

where $v_i = v(x_i)$, $x_i = a + ih$, $i = 0, 1, \ldots, n+1$, $h = (b-a)/(n+1)$, and the finite-difference operator L_h is given by

$$L_h v_i \equiv \begin{cases} \dfrac{v_{i-1} - 2v_i + v_{i+1}}{h^2}, & i = 1, n, \\ \dfrac{-v_{i-2} + 16v_{i-1} - 30v_i + 16v_{i+1} - v_{i+2}}{12h^2} & i = 2, 3, \ldots, n-1. \end{cases}$$

Let us write the finite-difference scheme (2) in matrix form as

$$Av = f(v), \tag{3}$$

where $v = (v^1, v^2, \ldots, v^p)^{\mathrm{T}}$, with $v^{(q)} = (v_1^{(q)}, v_2^{(q)}, \ldots, v_n^{(q)})$, $f = (f^{(1)}, f^{(2)}, \ldots, f^{(p)})$, with $f^{(q)} = (f_1^{(q)}, f_2^{(q)}, \ldots, f_n^{(q)})$, $q = 1, 2, \ldots, p$, and

$$f_i^{(q)} = 12h^2 \begin{cases} g^{(q)}(x_i, v_i) + \dfrac{u_0}{h^2} & \text{if } i = 1, \\ g^{(q)}(x_i, v_i) - \dfrac{u_0}{12h^2} & \text{if } i = 2, \\ g^{(q)}(x_i, v_i) & \text{if } i = 3, 4, \ldots, n-2, \\ g^{(q)}(x_i, v_i) - \dfrac{u_{n+1}}{12h^2} & \text{if } i = n-1, \\ g^{(q)}(x_i, v_i) + \dfrac{u_{n+1}}{h^2} & \text{if } i = n. \end{cases}$$

The matrix A is a block diagonal matrix of the form $A = \text{diagonal}(M)$, where M is the pentadiagonal matrix

$$M = \begin{bmatrix} 24 & -12 & 0 & 0 & 0 & 0 & \cdots & 0 & 0 \\ -16 & 30 & -16 & 1 & 0 & 0 & \cdots & 0 & 0 \\ 1 & -16 & 30 & -16 & 1 & 0 & \cdots & 0 & 0 \\ 0 & 1 & -16 & 30 & -16 & 1 & \cdots & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & 0 & 0 & 0 & \cdots & 30 & -16 \\ 0 & 0 & 0 & 0 & 0 & 0 & \cdots & -12 & 24 \end{bmatrix}_{n\times n}.$$

2. Factorization of the pentagonal matrix

The matrix M satisfies the following equality (cf. [7]):

$$MG = hS, \tag{4}$$

where

$$S = \begin{bmatrix} 12 & 0 & 0 & 0 & \cdots & 0 & 0 & 0 \\ -1 & 14 & -1 & 0 & \cdots & 0 & 0 & 0 \\ 0 & -1 & 14 & -1 & \cdots & 0 & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & 0 & \cdots & -1 & 14 & -1 \\ 0 & 0 & 0 & 0 & \cdots & 0 & 0 & 12 \end{bmatrix}$$

and Green's matrix G has the entries

$$G_{ij} = \begin{cases} ih(1 - jh) & \text{if } i \leqslant j, \\ jh(1 - ih) & \text{if } i > j. \end{cases}$$

One can check that

$$M_0 G = hI, \tag{5}$$

where I is the unitary matrix, and

$$M_0 = \begin{bmatrix} 2 & -1 & 0 & 0 & \cdots & 0 & 0 & 0 \\ -1 & 2 & -1 & 0 & \cdots & 0 & 0 & 0 \\ 0 & -1 & 2 & -1 & \cdots & 0 & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & 0 & \cdots & -1 & 2 & -1 \\ 0 & 0 & 0 & 0 & \cdots & 0 & -1 & 2 \end{bmatrix}.$$

Hence,

$$G^{-1} = \frac{1}{h} M_0.$$

From (4) and (5), we obtain the following factorization of the matrix M:

$$M = SM_0. \tag{6}$$

3. An optimal algorithm

The matrix Eq. (3) splits into p decoupled systems of equations of the form

$$Mv^{(q)} = f^{(q)}, \quad q = 1,2,\ldots,p. \tag{7}$$

Applying factorization (6), we rewrite the above system as follows:

$$\begin{aligned} &Sw^{(q)} = f^{(q)}, \\ &M_0 v^{(q)} = w^{(q)}, \quad q = 1,2,\ldots,p. \end{aligned} \tag{8}$$

Since both tridiagonal matrices S and M_0 are diagonally dominant, each of the two systems of equations in (8), for fixed q, can be successfully solved by the Gaussian elimination. Below, we give an algorithm for solving the tridiagonal systems of equations in (8). The algorithm is optimal in terms of the number of arithmetic operations as it consists of $8n - 9$ operations. We note that to solve the corresponding pentadiagonal system in (7), without factorization, there are $16n - 35$ arithmetic operations required.

The forward elimination applied to the first system of equations in (8) leads to the following upper triangular form of the system:

$$\begin{bmatrix} 1 & -\beta_1 & 0 & 0 & 0 & 0 & \cdots & 0 & 0 \\ 0 & 1 & -\beta_2 & 0 & 0 & 0 & \cdots & 0 & 0 \\ 0 & 0 & 1 & -\beta_3 & 0 & 0 & \cdots & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & 0 & 0 & 0 & \cdots & 1 & -\beta_{n-1} \\ 0 & 0 & 0 & 0 & 0 & 0 & \cdots & 0 & 1 \end{bmatrix} \begin{bmatrix} w_1^{(q)} \\ w_2^{(q)} \\ w_3^{(q)} \\ \vdots \\ w_{n-1}^{(q)} \\ w_n^{(q)} \end{bmatrix} = \begin{bmatrix} \hat{f}_1^{(q)} \\ \hat{f}_2^{(q)} \\ \hat{f}_3^{(q)} \\ \vdots \\ \hat{f}_{n-1}^{(q)} \\ \hat{f}_n^{(q)} \end{bmatrix}, \tag{9}$$

where $\hat{f}^{(q)} = (\hat{f}_1^{(q)}, \hat{f}_2^{(q)}, \ldots, \hat{f}_n^{(q)})$, with

$$\hat{f}_1^{(q)} = \frac{f_1^{(q)}}{12}, \qquad \hat{f}_i^{(q)} = (f_i^{(q)} + \hat{f}_{i-1}^{(q)})\beta_i, \quad i = 2,3,\ldots,n-1, \quad \hat{f}_n^{(q)} = \frac{f_n^{(q)}}{12}.$$

The coefficients β_i are given by the formula

$$\beta_1 = 0, \qquad \beta_i = \frac{1}{14 - \beta_{i-1}}, \quad i = 2,3,\ldots,n-1.$$

We notice that the sequence $\{\beta_i\}$, as $i \to \infty$, converges rapidly to the fixed point $\beta = 7 - 4\sqrt{3}$ of the iteration function

$$s(x) = \frac{1}{14 - x}, \quad 0 \leqslant x \leqslant 1.$$

Since, for $0 \leqslant x \leqslant 1$,

$$|s'(x)| = \frac{1}{(14-x)^2} \leqslant \frac{1}{13^2},$$

by the fixed point theorem, we obtain the following error estimate:

$$|\beta_i - \beta| \leqslant \frac{1}{169^i}|\beta_1 - \beta| = \frac{7-4\sqrt{3}}{169^i}, \quad i = 2,3,\ldots,n-1.$$

We can assume that $\beta_i = \beta$ for $i \geqslant 4$, with the accuracy of $(7-4\sqrt{3})/169^4 \approx 8.8 * 10^{-11}$. By the backward substitution, we find the solution $w^{(q)}$ to the first system of equations in (8) as follows:

$$w_n^{(q)} = \hat{f}_n^{(q)}, \qquad w^{(q,i)} = \hat{f}_i^{(q)} + \beta_i w_{i+1}^{(q)}, \quad i = n-1, n-2, \ldots, 1.$$

Now, the Gaussian elimination applied to the second system of equations in (8) leads to the reduced system of equations of the form (9) with $w^{(q)}$ replaced by $v^{(q)}$, $\hat{f}_i^{(q)}$ replaced by $\overline{f}_i^{(q)}$, and β_i replaced by γ_i. We have

$$\gamma_1 = \frac{1}{2}, \qquad \gamma_i = \frac{1}{2-\gamma_{i-1}} = \frac{i}{i+1}, \quad i = 2,3,\ldots,n-1.$$

$$\overline{f}_1^{(q)} = \frac{w_1^{(q)}}{2}, \qquad \overline{f}_i^{(q)} = (w_i^{(q)} + \overline{f}_{i-1}^{(q)})\gamma_i, \quad i = 2,3,\ldots,n.$$

Again, by the backward substitution, we compute the solution $v^{(q)}$ to the decoupled system of equations in (8)

$$v_n^{(q)} = \overline{f}_n^{(q)}, \qquad v_i^{(q)} = \overline{f}_i^{(q)} + \gamma_i v_{i+1}^{(q)}, \quad i = n-1, n-2, \ldots, 1.$$

The above algorithm contains $8n - 9$ arithmetic operations.

4. Implicit iterations

We shall solve the system of algebraic Eq. (3) using the following implicit iterative method:

$$Av^{(m+1)} = f(v^{(m)}), \quad m = 0,1,\ldots,s, \tag{10}$$

where the starting value of v, $v^{(0)}$, is given.

Since A is a block diagonal matrix, the above system splits into p decoupled systems of equations

$$Mv^{(q,m+1)} = f^{(q)}(v^{(q,m)}), \quad q = 1,2,\ldots,p. \tag{11}$$

Because the pentagonal matrix M is monotonic, the iterative method is convergent, (cf. [8, 9]).

The algorithm for solving (11) is as follows:

For $m = 1,2,\ldots,s$,

(1) set $v = v^{(m-1)}$,

(2) compute $\boldsymbol{f} = (f^{(1)}(v), f^{(2)}(v), \ldots, f^{(p)}(v))$,

(3) solve (8) by the algorithm given in the previous section.

5. Numerical examples

Several systems of differential equations of the form (1) have been solved using the algorithm with *Mathematica*. The *Mathematica* module `solveBVP` is attached. The module `solveBVP` takes three optional parameters `bound`, `startv` and `iters`, in addition to the parameters g, p, a, b, and n. The parameter `bound` specifies the boundary conditions $\{u_0, u_{n+1}\}$ as a $p \times 2$ array, `startv` $= v^{(0)}$ is a $p \times n$ array, and `iters` $= s$. The default values of the optional parameters are as follows:

`bound` — $p \times 2$ array of zeros corresponding to the homogeneous boundary conditions,

`startv` — a $p \times n$ array of zeros,

`iters` $= 2n$.

Now, we present two examples of the boundary value problem (1) solved by the method given above.

Example 1. Find the solution of the equation that represents rotation of a heavy string (cf. [4])

$$\frac{d^2u}{dx^2} + \frac{u}{4\sqrt{x^2+u^2}} = 0, \quad 0 < x < 1, \tag{12}$$

with the boundary value conditions $u(0) = 0$, and $u(1) = 1$.

We consider the partition of the interval $[a, b] = [0, 1]$ into $n = 9$ subintervals by $n + 1$ points. So that, $p = 1$, $a = 0$, $b = 1$, $n = 9$, and the list of the boundary value conditions $\{u_0, u_{n+1}\} = \{\{0, 1\}\}$.

To solve Eq. (12), we define the function

```
g1[x_,u_]:= {u[[1]]/(4 Sqrt[x^2+u[[1]]^2])},
```

and call the module

```
solveBVP[g1,1,0,1,9,bound->{{0,1}}].
```

In Table 1, we present numerical results.
One can easily find the interpolating polynomial

$$P(x) = 1.09093x - 0.092147x^2 + 0.001197x^3 - 0.0000155x^4 + 0.000131x^5 - 0.000239x^6 + 0.000293x^7 - 0.0002295x^8 + 0.0001044x^9 - 0.0000209x^{10},$$

through the data points given in Table 1 using *Mathematica*. This polynomial is the approximate solution that satisfies the Eq. (12) in the interval $(0, 1)$, with the residual error less than 10^{-5}. Because the finite difference scheme (2) is $O(h^4)$ convergent, the global error $u(x) - P(x) \approx O(h^4)$, when $0 < x < 1$.

Table 1

x	$v(x)$
0	0
0.1	0.108172
0.2	0.214509
0.3	0.319017
0.4	0.421704
0.5	0.522577
0.6	0.621644
0.7	0.718912
0.8	0.814389
0.9	0.908082
1	1

Example 2. Find the solution of the system of two nonlinear equations that represent the equilibrium state of a rotating rod (cf. [4]).

$$\begin{aligned} \frac{d^2u^{(1)}}{dx^2} &= \sin u^{(2)}(x), \\ \frac{d^2u^{(2)}}{dx^2} &= u^{(1)}(x)\cos u^{(2)}(x), \quad 0<x<1, \end{aligned} \tag{13}$$

with the nonhomogeneous boundary value conditions

$$u^{(1)}(0)=0, \quad u^{(1)}(1)=1, \qquad u^{(2)}(0)=0, \quad u^{(2)}(1)=1.$$

In this example, we define the function

```
g2[x_,u_] := {-Sin[u[[2]]],-u[[1]]*Cos[u[[2]]]}
```

and execute the command

```
solveBVP[g2,2,0,1,9,bound->{{0,1},{0,1}}].
```

The numerical results are given in Table 2

The interpolating polynomials

$$\begin{aligned} P(x) = {} & 0.854845x + 0.0000458x^2 + 0.144202x^3 + 0.0052374x^4 - 0.0191007x^5 \\ & + 0.0507075x^6 - 0.0794568x^7 + 0.0738018x^8 - 0.0392807x^9 + 0.0089985x^{10} \end{aligned}$$

and

$$\begin{aligned} Q(x) = {} & 0.869691x + 0.0006005x^2 + 0.137016x^3 + 0.0301239x^4 - 0.114618x^5 \\ & + 0.240532x^6 - 0.356843x^7 + 0.323749x^8 - 0.168064x^9 + 0.0378118x^{10}, \end{aligned}$$

correspond to data $x, v1, v2$ in Table 2. One can check with use of *Mathematica* that these polynomials satisfy equilibrium equations in the interval $(0,1)$, with the residual error less than 10^{-4}. So that, the global errors $u^{(1)}(x) - P(x) = O(h^4)$, and $u^{(2)}(x) - Q(x) = O(h^4)$.

Table 2

x	$v1$	$v2$
0	0	0
0.1	0.0856296	0.0871142
0.2	0.172129	0.175081
0.3	0.260371	0.264742
0.4	0.351229	0.356912
0.5	0.445582	0.452367
0.6	0.544304	0.551823
0.7	0.648268	0.655904
0.8	0.758329	0.765108
0.9	0.875311	0.879765
1	1	1

Mathematica module solveBVP.

```
Options[solveBVP] = {bound  -> homogeneous,
                     startv -> vzero,
                     iters  -> twon};

solveBVP[g_, p_, a_, b_, n_, opts___]:= Module[
        { h, x, boundary, v0, iterNumber, beta, gamma, solution,
          bcorrections, useboundary, solve3, solve5, oneStep  },

    h = (b-a)/(n+1);
    x = Table[ N[a + i h], {i, 1, n}];
    homogeneous = Table[0, {p}, {2}];
    vzero = Table[0,{p},{n}];
    twon = 2n;

    boundary=bound/.{opts}/.Options[solveBVP];
    v0=startv/.{opts}/.Options[solveBVP];
    iterNumber = iters/.{opts}/.Options[solveBVP];

    beta={0};
    Do[AppendTo[beta, 1/(14-Last[beta]) ], {3}];
    Do[AppendTo[beta, N[7 - 4 Sqrt[3]] ], {n-3}];
    gamma=Table[i/(i+1), {i, 1, n}];
    bcorrections=Table[{{12 *h^2 *boundary[[q,1]], - h^2*boundary[[q,1]]},
                        {-h^2*boundary[[q,2]], 12*h^2*boundary[[q,2]]}}/h^2,
                  {q,1,p}];

    useboundary[f_, bcorr_]:=
       Join[Take[f,2] + bcorr[[1]], Take[f, {3,-3}], Take[f,-2] + bcorr[[2]]];
```

```
solve3[f_, d11_, alpha_]:= Module[{ f1, sol },
   f1[1]=f[[1]]/d11;
   f1[i_]:=f1[i]=(f[[i]]+f1[i-1])*alpha[[i]];
   sol[n]=If[d11==12, f[[n]]/12, f1[n]];
   sol[i_]:=sol[i]=f1[i]+alpha[[i]]*sol[i+1];
   Table[sol[i],{i,1,n}]];

solve5[f_]:=With[
   {w= solve3[f, 12, beta]},
   solve3[w, 2, gamma] ];

oneStep[v_]:=Module[ {ff, bff},
   ff=12*h^2* N[Transpose[MapThread[g,{x,Transpose[v]}]]];
   bff=MapThread[useboundary, {ff,  bcorrections}];
   Map[solve5, bff] ];

solution = Nest[oneStep, v0, iterNumber];
PrependTo[solution, x];
With[  {   header = Prepend[Table[Subscripted [v[i]], {i,1,p}], "x"],
           asol= Prepend[Table[boundary[[q,1]], {q,1,p}], a],
           bsol= Prepend[Table[boundary[[q,2]], {q,1,p}], b] },
        TableForm[Join[{header}, {asol}, Transpose[solution], {bsol}]]].
```

References

[1] J.H. Bramble, B.E. Hubbard, On a finite difference analogue of an elliptic boundary problem which is neither diagonally domonant nor non-negative type, J. Math. Phys. 43 (1964) 117–132.

[2] M.M. Chawla, R.S. Rao, High accuracy methods for $y'' = f(x, y)$, IMA, J. Numer. Anal. 5 (1985) 215–220.

[3] J.M. Franco, M. Palacios, High-order P-stable multistep methods, J. Comput. Appl. Math. 30 (1990) 1–10.

[4] S. Fucik, A. Kufner, Nonlinear Differential Equations, Elsevier, Amsterdam, 1980.

[5] T.E. Simos, An explicit four-step phase-fitted method for the numerical integration of second-order initial-value problems, J. Comput. Appl. Math. 55 (1994) 125–133.

[6] T.E. Simos, Explicit two-step methods with minimal phase-lag for the numerical integration of special second-order initial-value problems and their application to the one-dimensional Schrodinger equation, J. Comput. Appl. Math. 39 (1992) 89–94.

[7] T. Styś, K. Styś, An $O(h^4)$ locally overconvergent finite difference scheme for the equation $u_t = u_{xx} + f(t,x,u)$, J. Comput. Appl. Math. 34 (1991) 221–231.

[8] R.S. Varga, Matrix Iterative Analysis, Prentice-Hall, Englewood Cliffs, NJ, 1962.

[9] D.M. Young, Iterative Solution of Large Linear Systems, Academic Press, New York, 1971.