First International Conference
Modelling and Development of Intelligent Systems
Sibiu - Romania, 22-25 October, 2009

A comparison between two collocations methods for linear polylocal problems - a Computer Algebra based approach

Daniel N.Pop , Radu T. Trîmbiţaş

Abstract

Consider the problem:

$$\begin{aligned} -y''(t) + q(t)y(t) &= r(t), \qquad t \in [a, b] \\ y(c) &= \alpha \\ y(d) &= \beta, \qquad c, d \in (a, b). \end{aligned}$$

The aim of this paper is to present two approximate solutions of this problem based on B-splines and first kind Chebyshev polynomials, respectively. The first solution uses a mesh based on Legendre points, while the second uses a Chebyshev-Lobatto mesh. Using computer algebra techniques and a Maple implementation, we obtain analytical expression of the approximations and give examples. Chebyshev method has a smaller error, but for large number of mesh points the B-spline method is faster and requires less memory.

1 Introduction

Consider the problem:

$$-y''(t) + q(t)y(t) = r(t), \qquad t \in [a, b] \tag{1}$$
$$y(d) = \alpha \tag{2}$$
$$y(e) = \beta, \qquad d, e \in (a, b), d < e. \tag{3}$$

where $q, r \in C[a, b]$, $\alpha, \beta \in \mathbb{R}$. *This is not a two-point boundary value problem*, since $d, e \in (a, b)$.

If the solution of the two-point boundary value problem

$$\begin{aligned} -y''(t) + q(t)y(t) &= r(t), \qquad t \in [d, e] \\ y(d) &= \alpha \\ y(e) &= \beta, \end{aligned} \tag{4}$$

exists and it is unique, then the requirement $y \in C^2[a, b]$ assures the existence and the uniqueness of (1)+(2)+(3).

We have two initial value problems on $[a, d]$ and $[e, b]$, respectively, and the existence and the uniqueness for (4) assure existence and uniqueness of these problems. It is possible to solve this problem by dividing it into the three above-mentioned problems and to solve each of these problem separately, but we are interested to a unitary approach that solve it as a whole.

In 1966, two researchers from Tiberiu Popoviciu Institute of Romanian Academy, Cluj-Napoca, *Dumitru Ripianu* and *Oleg Arama* published a paper on a polylocal problem, see [9].

2 Principles of the method

The implementation is inspired from [4, 5].

2.1 B-spline method

Our first method is based on collocation with nonuniform cubic B-splines [2, 10]. For properties of B-spline and basic algorithms see [5].

Consider the mesh (see [1])

$$\Delta : a = x_0 < x_1 < \cdots < x_m < x_{m+1} = b, \tag{5}$$

and the step sizes

$$h_i := x_{i+1} - x_i, \qquad i = 0, \ldots, m.$$

Within each subinterval we insert k points

$$0 \le \rho_1 < \rho_2 < \cdots < \rho_k \le 1,$$

which are the roots of the kth Legendre's orthogonal polynomial on $[0, 1]$ [6, 8].

Finally, the mesh has the form

$$\xi_{i,j} := x_i + h_i \rho_j, \qquad j = 1, \ldots, k, \qquad i = 0, \ldots, m.$$

The number of mesh points is now $N = (m + 1)k$.

We shall choose the basis such that the following conditions hold:

- the solution verifies the differential equation (1) at $\xi_{i,j}$;
- the solution verifies the conditions (2), (3).

We need a basis having $N + 2$ cubic B-spline functions.

One renumbers the points such that the first point is x_0 and the last is x_{n+1}.

In order to impose the fulfillment of (1) at a and b we complete the mesh with points $x_{-k}, x_{-k+1}, \ldots, x_{-1}$ and $x_{n+2}, x_{n+3}, \ldots, x_{n+k+1}$.

The form of solution is

$$y(t) = \sum_{i=-1}^{n+2} b_i B_i(t), \tag{6}$$

where $B_i(t)$ is the B-spline with knots $x_{k-2}, x_{k-1}, x_k, x_{k+1}, x_{k+2}$.

The conditions on solution yield a linear system with $n + 4$ equations and $n + 4$ unknowns (the coefficients b_i, $i = -1, \ldots, n + 2$).

The system matrix is banded with at most 4 nonzero elements on each line (3 nonzero at each mesh point and four at d and e).

2.2 Chebyshev method

Our second method is based on first kind Chebyshev polynomials [6, 8]. We consider the mesh

$$\frac{b-a}{2} \cos \frac{k\pi}{n} + \frac{a+b}{2}, \qquad k = 0, \ldots, n \tag{7}$$

(the extremes of Chebyshev #1 polynomials, or equivalently the roots of Chebyshev #2 polynomials) completed with inner points c and d. The form of the solution is

$$y(t) = \sum_{i=0}^{n+1} c_i T_i(t); \tag{8}$$

where $T_i(t)$ is the k-th degree first kind Chebyshev polynomial on interval $[a, b]$. As in the previous section, the fulfillment of (1), (2) and (3) leads us to a system of $n + 2$ equations and $n + 2$ unknowns (the coefficients c_i, $i = 0, \ldots, n + 1$). This time the matrix is dense.

3 Maple implementation

We implement our ideas in Maple 10. For necessary details on Maple see [7].
Both methods return the approximation in analytic form.

3.1 B-spline method

The basic functions are computed using the function `BSpline` of the package `CurveFitting`. The B-spline basis is obtained through Maple sequence

```
> S:=(x,u,k)->eval(BSpline(4,t,
> knots=[seq(u[i],i=k-2..k+2)]),
> t=x):
```

`S(x,u,k)` computes the cubic B-spline in variable `x`,
with knots `u[k-2],...,u[k+2]`.

The procedure `genspline` computes the B-spline solution. It accepts the mesh `x`, the number of points `n`, the functions `q` and `r`, the points `d`, `e` and the values at `d` and `e`, `alpha` and `beta`, respectively. It returns the solution `y`, given by (6). The matrix of the system and the right-hand side vector are constructed element by element and the solution is computed using the function `LinearSolve` from `LinearAlgebra` package. This is a fast and flexible solution, and allows the selection of the solution method and gaining additional information, like condition number. Here is the Maple code.

```
> genspline:=proc(x,n,q,r,d,e,
> alpha,beta)
> local k, i, A, y, poze, pozd, ii,p,xe,xd, Y;
> global S, b;
> A:=Matrix(n+4,n+4); y:=Vector(n+4):
> b:=Vector(n+4):
> ii:=1;
> for i from 0 to n+1 do
> for k from max(i-1,-1) to i+1 do
> A[ii,k+2]:=(-eval(diff(S(t,x,k),
> t$2), t=x[i])+q(x[i])*
> eval(S(t,x,k),t=x[i]));
> end do:
> y[ii]:=r(x[i]);
> if (x[i]<d and x[i+1]>d) then
> ii:=ii+1; pozd:=ii; xd:=i;
> end if:
> if (x[i]<e and x[i+1]>e) then
> ii:=ii+1; poze:=ii;  xe:=i;
> end if:
> ii:=ii+1;
> end do:
> p:=xd;
> for k from p-1 to p+2 do
> A[pozd,k+2]:=eval(S(t,x,k),t=d);
> end do;
> y[pozd]:=alpha;
> p:=xe;
> for k from p-1 to p+2 do
> A[poze,k+2]:=eval(S(t,x,k),t=e);
> end do;
> y[poze]:=beta;
> b:=LinearSolve(A,y);
> Y:=0:
> for k from -1 to n+2 do
> Y:=Y+b[k+2]*S(t,x,k):
> end do:
> return Y:
> end proc:
```

The procedure genspline accepts the mesh given in array form. The procedure gendivLeg generates the mesh as shown in Section 2.1. It calls the procedure genpoints. It computes the Legendre polynomial, solve it, and generates mesh points using an affine transform. The Legendre polynomials are generated using the orthopoly package, and their roots are obtained via solve function. Here is the code for genpoints:

```
> genpoints:=proc(a,b,N,k)
> local L,i,j,xu,xc,pol,pol2,sol,
> h,nL,x;
> L:=[a];
> h:=(b-a)/(N+1);
> xc:=a-h; pol:=P(k,t);
> pol2:=expand(subs(t=2*x-1,pol));
> sol:=fsolve(pol2);
> for i from 0 to N+2 do
> xu:=xc+h;
> for j from 1 to k do
> L:=[op(L),xc+(xu-xc)*sol[j]];
> end do;
> xc:=xu;
> end do;
> L:=[op(L),b];
> L:=sort(L);
> return L;
> end proc:
```

The code for gendivLeg closes the section.

```
> gendivLeg:=proc(a,b,n,k)
> local h,x,Y,L,nn,j:
> L:=genpoints(a,b,n,k);
> L:=convert(L,rational,exact);
> nn:=nops(L)- 2*k;
> x:=Array(-k..nn+k-1,L):
> return x;
> end proc:
```

3.2 Chebyshev method

The Chebyshev polynomials are generated via the orthopoly package. The Maple sequence

```
> S:=(x,k,a,b)->T(k,
> ((b-a)*x+a+b)/2):
```

computes the k-th degree Chebyshev polynomial on interval $[a, b]$. The following Maple procedure genceb is the analogous of genspline. It uses solve to compute the Chebyshev coefficients.

```
> genceb:=proc(x,n,q,r,c0,d0,
> alpha,beta)
> local k, ecY, ecd, C, h, Y, c, a, b;
> global S;
> a:=x[0]; b:=x[n-1];
> Y:=0;
> for k from 0 to n+1 do
> Y:=Y+c[k]*S(t,k,a,b);
> end do;
> Y:=simplify(Y);
> ecY:=-diff(Y,t$2)+q(t)*Y=r(t):
> ecd:=Array(0..n+1);
> for k from 0 to n-1 do
> ecd[k]:=eval(ecY,t=x[k]):
> end do;
> ecd[n]:=eval(Y,t=c0)=alpha:
> ecd[n+1]:=eval(Y,t=d0)=beta:
> C:=solve({seq(ecd[k],k=0..n+1)},
> [seq(c[k],k=0..n+1)]);
> assign(C):
> return Y:
> end proc:
```

The mesh points are the roots of the n-th degree second kind Chebyshev polynomials (formula (7)) and the points c and d.

4 Numerical examples

We present two examples: one with a nonoscillating solution and the other with oscillating solution. A problem with a nonoscillating solution is simple and does not require a large computational effort. A problem with an oscillating solution is harder, and requires a mesh with a large number of points. The methods do not depend on conditions on $q(x)$. We solved our examples using both methods. For each example and method we plot the exact and the approximate solution and generate the execution profile (with the pair `profile` - `showprofile`). The first example is from [3, page 560]

$$\begin{aligned} -y'' - y &= x, \qquad x \in [0,1] \\ y\left(\frac{1}{6}\right) &= -\frac{1}{6}\frac{-6\sin\frac{1}{6} + \sin 1}{\sin 1}, \\ y\left(\frac{3}{4}\right) &= -\frac{1}{4}\frac{-4\sin\frac{3}{4} + 3\sin 1}{\sin 1}. \end{aligned}$$

The exact solution is $Z(t) = -\frac{-\sin(t) + t\sin 1}{\sin 1}$, and we computed it using `dsolve`. We chose $n = 10$ for both methods and $k = 3$ for the first method. Figure 1 shows the exact and the approximate solution computed using the first method. The error plot in a semilogarithmic scale is given in Figure 2.

The corresponding graphs for Chebyshev methods are illustrated in Figures 3 and 4.

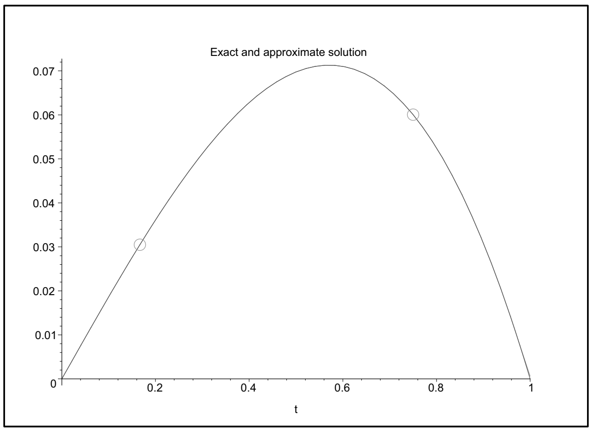

Figure 1. **The graph of exact and approximate solution, nonoscillating problem, B-spline method,** $n = 10$**,** $k = 3$

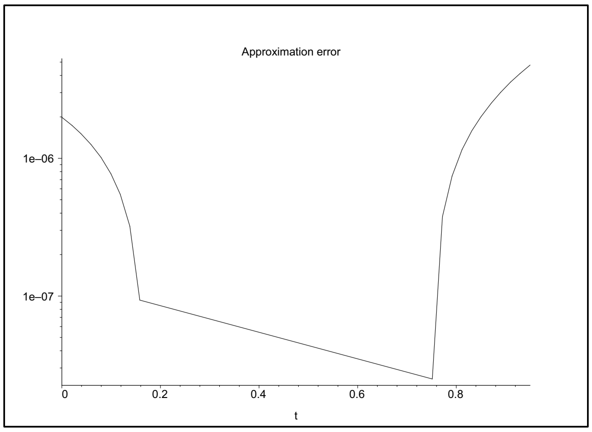

Figure 2. **Error plot, nonoscillating problem, B-spline method,** $n = 10$**,** $k = 3$

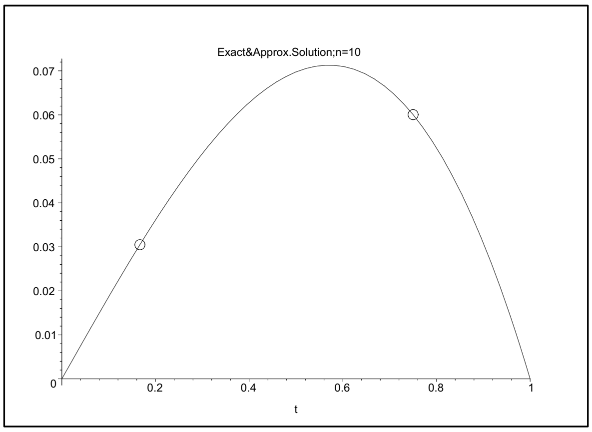

Figure 3. **Exact and approximate solution, nonoscillating problem,Chebyshev method,**$n = 10$

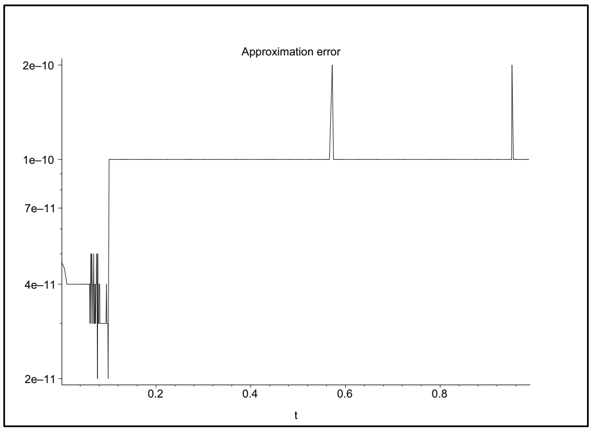

Figure 4. Error plot, nonoscillating problem, Chebyshev method, $n = 10$

Here are the profiles for the procedures in the case of nonoscillating problem. The function `showprofile` for the B-spline method gives the following results:

```
function            depth    calls     time    time%         bytes   bytes%
---------------------------------------------------------------------------
genspline               1        1    2.496   100.00      92768440   100.00
---------------------------------------------------------------------------
total:                  1        1    2.496   100.00      92768440   100.00
```

The profile for Chebyshev method is:

```
function            depth    calls     time    time%         bytes   bytes%
---------------------------------------------------------------------------
genceb                  1        1    0.249   100.00       9291004   100.00
---------------------------------------------------------------------------
total:                  1        1    0.249   100.00       9291004   100.00
```

The second example has an oscillating solution:

$$-y'' - 243y = x, \qquad x \in [0,1]$$
$$y\left(\frac{1}{6}\right) = -\frac{1}{1458}\frac{-6\sin\frac{3}{2}\sqrt{3} + \sin 9\sqrt{3}}{\sin 9\sqrt{3}}$$
$$y\left(\frac{3}{4}\right) = -\frac{1}{972}\frac{-4\sin\frac{27}{4}\sqrt{3} + 3\sin 9\sqrt{3}}{\sin 9\sqrt{3}}.$$

The exact solution, provided by `dsolve` is $Z(t) = -\frac{1}{243}\frac{-\sin 9\sqrt{3}t + t\sin 9\sqrt{3}}{\sin 9\sqrt{3}}$. We chose $n = 100$ for both methods and $k = 3$ for the first method. Figure 5 gives the graph of exact and approximate solution for the oscillating problem. The error plot appear in Figure 6. The corresponding graphs for Chebyshev methods are given in Figures 7 and 8, respectively.

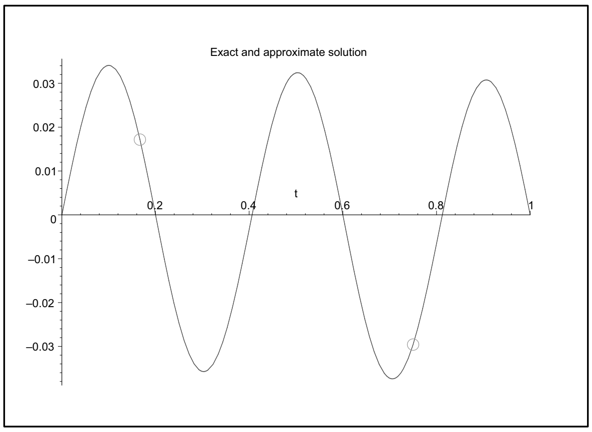

Figure 5. **Exact and approximate solution, oscillating problem, B-spline method,** $n = 100$**,** $k = 3$

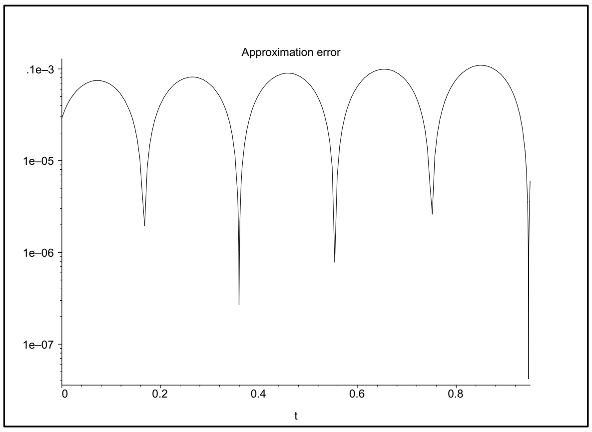

Figure 6. **Error plot, oscillating problem, B-spline method,** $n = 100$**,** $k = 3$

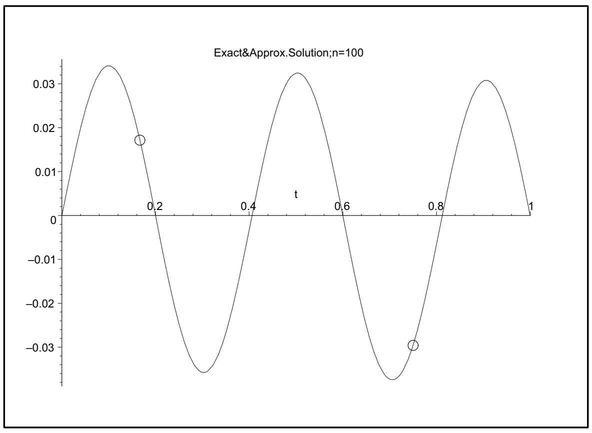

Figure 7. **Exact and approximate solution, oscillating problem, Chebyshev method,** $n = 100$

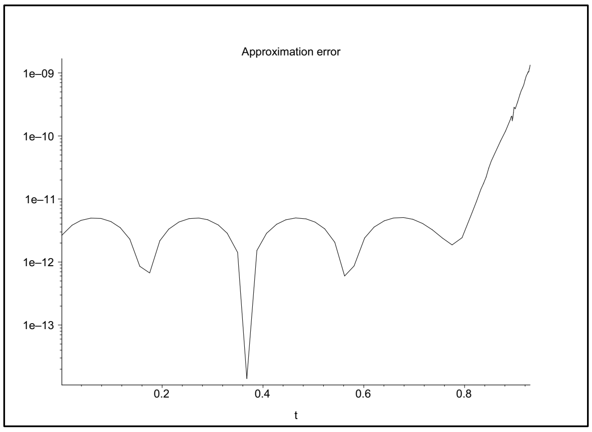

Figure 8. Error plot, oscillating problem, Chebyshev method, $n = 100$

Here are the profiles for the procedures in the case of oscillating problem.

function	depth	calls	time	time%	bytes	bytes%
genspline	1	1	36.691	100.00	990061444	100.00
total:	1	1	36.691	100.00	990061444	100.00

function	depth	calls	time	time%	bytes	bytes%
genceb	1	1	174.814	100.00	4365866236	100.00
total:	1	1	174.814	100.00	4365866236	100.00

5 Conclusions

The Chebyshev method has a smaller error (see error plots, Figures 2, 4, 6, 8). For the nonoscillating solution and a mesh with a small number of subintervals Chebyshev method is faster and requires less memory. If the number of points increases the B-spline method is faster and requires less memory. The reason is that for the B-spline method the matrix of the system that provides the coefficients is a band matrix with at most 4 nonzero elements per line, while for Chebyshev method the matrix is dense. The example with oscillating solution supports this conclusion.

Our approach based on computer algebra has the following advantages:

- The choice of mesh points is arbitrary.
- The degree of Legendre polynomial can be changed.
- We need not bother with differentiation, equation building, ordering and so on.
- The analytic form of the solution allow to compute the approximation at any point, to plot it and to use it further as input for other problems.

References

[1] U.Ascher,J.Christiansen and R. Russel , A collocation Solver for Mixed Order Systems Of Boundary Value Problems ,*Mathematics of computations.*33 nr:146,pp 659-679,1979

[2] U.Ascher,S.Pruess and R.D. Russel , On spline basis selection for solving differential equations, *SIAM,J.Numer.Anal.*20,no.1,pp 121-142, 1983

[3] L.R. Burden,J.D. Faires , *Numerical analyses*, PWS-Kent Publishing Company,New York,1985.
[4] C.deBoor,Package for calculating with B-splines,*SIAM J.Numer.Anal.*14(1977)no.3,pp.441-472,1977.
[5] C.deBoor,*A practical guide to splines,* Springer-Verlag,Berlin,Heidelberg,NewYork,1978.
[6] W.Gautschi, *Numerical Analysis,An Introduction*,Birkhauser,Basel,1997.
[7] A.Heck,*Introduction to Maple*,Springer-Verlag ,Berlin,Heidelberg,1997.
[8] Al. Lupas,*Numerical Methods*, Constant,Sibiu,2000,(in Romanian).
[9] D. Ripeanu,O.Arama,Qulque recherche actuelles consernant l'equtions de Ch.de la Valee Poussin relative au problem polylocal dans la theorie des equations differentials,*Studia Mathematica* 8(31), pp 19-28,1966.
[10] M.H Schultz ,*Spline Analysis,*Prentice Hall ,Inc.Englewood Clifs,New Jersey,1972

Daniel N.Pop
Romanian-German University Sibiu
Faculty of Economics and Computers
Calea Dumbravii nr: 28-32
Romania
E-mail: *danielnicolaepop@yahoo.com*

Radu T. Trîmbiţaş
Babe ş-Bolyai University,Cluj-Napoca
Department of Applied Mathematics
Mihail Kogalniceanu street nr:1
Romania
E-mail: *tradu@math.ubbcluj.ro*