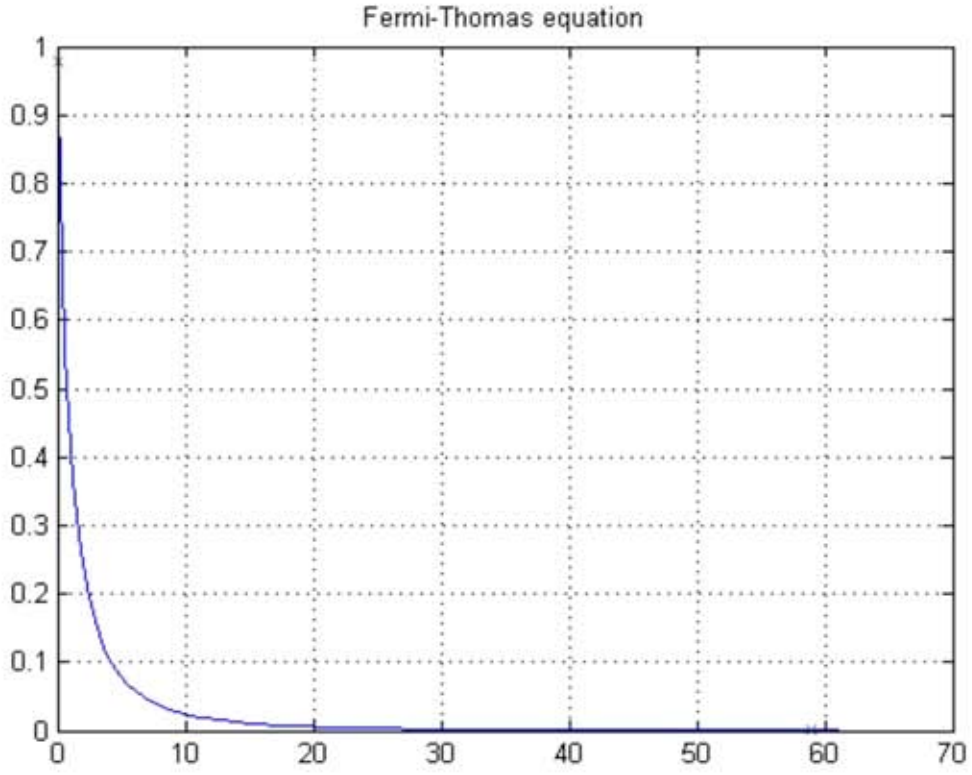
DANIEL N. POP

ASPECTS CONCERNING SOME NUMERICAL METHODS FOR APPROXIMATE SOLUTION OF TWO-POINT BOUNDARY VALUE PROBLEMS



Presa Universitară Clujeană

DANIEL N. POP

ASPECTS CONCERNING SOME NUMERICAL METHODS FOR APPROXIMATE SOLUTION OF TWO-POINT BOUNDARY VALUE PROBLEMS

Referenţi ştiinţifici:

Prof. univ. dr. Damian Trif

Conf. univ. dr. Radu T. Trîmbiţaş

ISBN 978-973-595-671-4

Universitatea Babeş-Bolyai
Presa Universitară Clujeană
Director: Codruţa Săcelean
Str. Hasdeu nr. 51
400371 Cluj-Napoca, România
Tel./fax: (+40)-264-597.401
E-mail: editura@editura.ubbcluj.ro
http://www.editura.ubbcluj.ro/

DANIEL N. POP

ASPECTS CONCERNING SOME NUMERICAL METHODS FOR APPROXIMATE SOLUTION OF TWO-POINT BOUNDARY VALUE PROBLEMS

PRESA UNIVERSITARĂ CLUJEANĂ /
CLUJ UNIVERSITY PRESS
2014

Dedication: To my wife, **Ligia**

Contents

List of Figures

Chapter 1

Introduction

The purpose of this work is to determinate the approximate solutions of boundary value problems with conditions inside the interval $(0, 1)$ using collocation method with global *B-spline functions* of degree k (order $k + 1$), orthogonal polynomials *Chebyshev* and combined methods with *B-spline functions* or *C.C method* and *Runge-Kutta* methods. This work is structured in five chapters:

- *Chapter 1- Introduction,*
- *Chapter 2- Main results,*
- *Chapter 3- Linear form,*
- *Chapter 4- Nonlinear form,*
- *Chapter 5- MATLAB* [1] *and MAPLE* [2] *Codes.*

In the second Chapter, we present main results on existence and uniqueness of two-point boundary value problems related to linear or nonlinear differential equations, basic issues on spline functions, numerical methods based on spline functions and collocation methods. It also present a series of known results in the theory of differential equations to be used as numerical examples studied by us in subsequent chapters. *In Chapter 3*, we study both problems with oscillatory and non-oscillatory solutions and numerical examples in this chapter are of great technical domains: theory of waves and environmental protection (pollutant transport in rivers). Under certain conditions imposed using arbitrary grid we determined a global approximation method for the oscillatory solutions based on cubic B-spline functions, in which the internal memory used and run time are smaller than if we use a global *C.C (Chebyshev Collocation)* method. In the case of existence of singularities at the ends of interval $(0, 1)$, we determine an approximate solution using a global method based on *B-spline functions* of degree k and *Runge-Kutta* methods. We dedicated *Chapter 4* to study the nonlinear problems with unique solution, and to determined the approximate solution using three methods:

1. a global collocation method with B-spline functions of degree k,
2. a collocation method based on combined *B-spline functions* of degree k and *Runge-Kutta* method,

[1] MATLAB is a trademark of Mathworks Inc., Natick M.A. United States of America
[2] MAPLE is a trademark Waterloo Maple Inc., Ontario Canada

3. a combined method based on collocation methods and *Runge-Kutta* method or *C.C (Chebyshev Collocation)* method.

We also showed that for the case when we have singularities at the endpoints of the interval $(0, 1)$ can not apply only to last two methods. The examples in this chapter are from the areas: of internal combustion model (*Bratu*'s problem), average temperature in advection-diffusion process (with applications in medical sciences) and in a semiclassical description of the charge density in atoms of high atomic number (*Thomas-Fermi* equation). We do a comparative study of these numerical methods, making their implementation of algorithms written in `MATLAB 2010a` and `Maple 13.0` and we was also concerned with the approximation errors and determine their implementation costs (run time and internal memory used). In *Chapter 5* we gave codes in `MATLAB` and `MAPLE` for examples used in paper.

The writing of this book has benefited enormously from a lot of discussion with prof. dr. Ion Păvăloiu, conf. dr. Radu T.Trîmbiţaş from " Babeş-Bolyai " University Cluj-Napoca and prof. dr. Eugen Drăghici from " Lucian Blaga " University Sibiu.

We appreciate the help provided by many experts who have commented on portions of this work, prof. dr. Damian Trif from " Babeş-Bolyai " University Cluj-Napoca and prof. dr. Alexandru I. Lupaş from " Lucian Blaga " University Sibiu have been especially helpful.

Daniel N.Pop
Romanian-German University of Sibiu
Faculty of economic engineering in electrical, energy, electronic
Calea-Dumbravii street nr: 28-32
Romania
E-mail:popdaniel31@yahoo.com

Chapter 2

Basic Notions and Main Results

2.1 Conditions of existence and uniqueness for two point boundary value problems in compact interval $[a, b]$.

The problem studied in this work has the form (PVPN)

$$\begin{cases} y'' = -f(x,y), & x \in [0,1] \\ y(a) = \alpha; & \\ y(b) = \beta, \quad a, b \in (0,1),\ b < a & \end{cases} \tag{2.1}$$

where $f(x,y) \in C([0,1] \times \mathbb{R})$, $a, b, \alpha, \beta \in \mathbb{R}$.
We also consider the problem (BVPN)

$$\begin{cases} y''(x) + f(x,y) = 0, & x \in [a,b] \\ y(a) = \alpha, & \\ y(b) = \beta. & \end{cases} \tag{2.2a}$$

The problem (BVPN) is equivalent to an integro-differential equation of *Fredholm* type:

$$y(x) = \frac{x-a}{b-a}\beta + \frac{b-x}{b-a}\alpha + \int_a^b G(x,s) f(s, \varphi(s)) ds,$$

where G (x, s) is *Green*'s function given by:

$$G(x,s) = \begin{cases} \frac{(s-a)(b-x)}{b-a}, \text{ if } a \le s \le x \le b \\ \frac{(x-a)(b-s)}{b-a}, \text{ if } a \le x \le s \le b \end{cases}.$$

Definition 1 *Let $[a,b] \subset R$, finite interval and a function $f : [a,b] \times R \to R$, f satisfies a* Lipshitz *condition in variable y, if there exist a real constant $K \in R$ so that:*

$$\begin{aligned} |f(x,y_1) - f(x,y_2)| &\le K\ |y_1 - y_2| \\ \forall (x,y_1), (x,y_2) &\in \Omega, \end{aligned} \tag{2.3}$$

and $\Omega \subset [a,b] \times R$ is a domain.

Theorem 2 ***(Picard) [40, page 33]****If $f : [a,b] \times R \to R$ satisfies the conditions*

1. *f is continuous in a domain $\Omega \subset [a,b] \times R$,*
2. *f satisfies a Lipschitz condition of the form (2.3),*
3. *$(b-a) \leq h$, where $h > 0$ is small enough,*
4. *if in addition h verifies the relationship:*

$$\frac{1}{2}K \cdot h^2 + 4 \cdot h \leq 1,$$

then there exist the unique solution of the problem (BVPN) on the interval $[a,b]$ of length h.

Next we give the following theorem which offers a sufficient condition of uniqueness of the solution of the problem (BVPN).

Theorem 3 ***(R.D Russel, L.F Shampine) [50, pag 10]*** *Suppose that $y(x)$ is a solution of the boundary value problem (BVPN), that the functions*

$$f(x,z) \text{ and } \frac{\partial f(x,z)}{\partial y}$$

are defined and continuous for $a \leq x \leq b$, and $|z-y| \leq \delta,\ \delta > 0$, and the homogeneous equation $y''(x) = 0$ subject to the homogeneous boundary conditions $y(a) = y(b) = 0$ has only the trivial solution. If the linear homogeneous equation

$$z''(x) + \frac{\partial f(x,y)}{\partial y} z(x) = 0,$$

has only trivial solution, then this is sufficient to guarantee that there exists a $\sigma > 0$, such $y(x)$ is the unique solution of the problem (BVPN) in the sphere

$$\{w : \|w - y''\| \leq \sigma\}.$$

Also we we recall the following result.

Theorem 4 *[30, pp: 112-113] Suppose that*

$$D = \{a \leq x \leq b, -\infty < y < \infty\},$$

and $f(x,y)$ is continuous on D. If f satisfies a Lipschitz *condition on D in the variable y, then the initial value problem (IVP)*

$$\begin{cases} y' = z, \\ z' = -f(x,y),\ a \leq x \leq b \\ y(a) = \alpha \\ y'(a) = \gamma \end{cases},$$

has a unique solution $y(x)$ for $a \leq x \leq b$.

If the problem (BVPN) has the unique solution, the requirement $y(x) \in C^1[0,1]$ ensure the existence and the uniqueness of the solution of the problem (PVPN).

We will consider the cases:

1. f linear, i.e $f(x,y) = q(x)y(x) - r(x)$, where q, $r \in C[0,1]$
2. f nonlinear (y' missing) .

Remark 5 *In the following we use the notations: for linear boundary value problem (BVPL), (PVPL) and (BVPN), (PVPN) if f is nonlinear.*

In linear case occurs:

Theorem 6 ***([14, pag 520], [37, pag 286])*** *Let the (BVPL) problem, and in addition:*

1. $q(x)$, $r(x) \in C[a,b]$,
2. $q(x) \geq 0$ *pe* $[a,b]$,

then the (BVPL) problem has unique solution.

Theorem 7 ***[1, pag 461]*** *The (BVPL) problem has unique solution, iff homogeneous problem (with $r = 0$, $\alpha = \beta = 0$) has only trivial solution.*

The question is how we determine an approximate solutions of the problem (PVPL), methods are considered from one of the following classes: discrete methods and collocation methods.

Since in linear case we have studied the oscillatory and non-oscillatory solutions, we recall some notions and basic results consisting in disconjugate criteria.

Definition 8 *[49, pag 79] Let the differential equation*

$$-y''(x) + q(x)y(x) = 0, \tag{2.4}$$

where $q \in C([a,b]$. A solution of differential equation (2.4) is called oscillatory if it has more than two zeros on the interval $[a,b]$.

Remark 9 *According to* Sturm*'s theorem [49, pag 76] that, if the differential equation (2.4) has an oscillatory solution, then all they are oscillatory solutions.*

Theorem 10 ***(Lyapunov)*** *[32, pag 14] The differential equation*

$$-y''(x) + q(x)y(x) = 0,$$

has oscillatory solutions on $[a,b]$, if

$$(b-a)\int_a^b |q(x)|\,dx > 4, \tag{2.5}$$

and non-oscillatory solutions, if

$$(b-a)\int_a^b |q(x)|\,dx \leq 4.$$

2.2 Numerical methods for boundary value problems

2.2.1 Discrete methods

The discrete methods are one step methods and multistep methods. Since in following chapters we use *Runge-Kutta* methods for initial value problem (IVP), we recall the following results.

One step *Runge-Kutta* scheme

A discrete numerical one step approximation solution $\{y_i : i = 0, 1, ...n\}$ can be calculated on nonuniform mesh Δ of the interval $[a, b]$

$$\Delta := \{a = x_0 < x_1 < ... < x_n = b\}, \tag{2.6}$$

defined so that

$$y_i \approx y(x_i),$$

where $y(x) \in C^1[a, b]$ is unique solution of (BVPN) problem. We denote the stepsize by

$$h_i = x_{i+1} - x_i \ ; \ i = 0, 1, 2, ..., n-1. \tag{2.7}$$

If

$$h_0 = ... = h_{n-1} = \frac{(b-a)}{n},$$

we have uniform mesh.

In the case of nonuniform mesh we denote by:

$$h := \max |x_{i+1} - x_i| \ , \ i = 0, 1, 2, ..., n-1, \tag{2.8}$$

$$\underline{h} := \min |x_{i+1} - x_i| \ , \ i = 0, 1, 2, ..., n-1. \tag{2.9}$$

We construct the following collocation points like in ([53], [54])

$$\xi_{ij} = x_i + h_i \rho_j; \ 1 \leq j \leq k, \ 0 \leq i \leq n-1, \tag{2.10}$$

on each subinterval $[x_i, x_{i+1}], \ i = 0, 1, ..., N-1$, where :

$$0 < \rho_1 < \rho_2 < ... < \rho_k < 1, \tag{2.11}$$

are the roots of k-th *Legendre* polynomial (see [4] for more details).

Definition 11 *[5, pag 69] The general form of a $q-$stage* Runge-Kutta *method is*

$$\begin{cases} y_0(a) = \alpha, \\ y_{i+1} = y_i + h_i \sum\limits_{j=1}^{k} \beta_j f_{ij}; \ 0 \leq i \leq n-1 \end{cases}, \tag{2.12}$$

where

$$f_{ij} = f(\xi_{ij}, y_i + h_i \sum_{l=1}^{k} \alpha_{jl} f_{il}); \ 1 \leq j \leq k.$$

The *Runge-Kutta* method (2.12) is called **explicit** if

$$\sum_{l=1}^{k} \beta_l = 1;\ \sum_{l=1}^{k} \alpha_{jl} = \rho_j;\ 1 \leq j \leq k, \tag{2.13}$$

where ρ_j are given by (2.11), it is called **implicit** otherwise. **Explicit** schemes have the advantage that the computation of new approximation y_{i+1} is straightforward. For **implicit** schemes we need to solve a nonlinear equation (if f is nonlinear). If :

$$\left| h\frac{\partial f}{\partial y} \right| \|A\| < 1,$$

where the matrix A is define as [5, page: 113], then this can be done by a simple fixed-point iteration and otherwise by the more expensive *Newton* method. Note that an implicit scheme always requires to predict a starting value for the iteration.

Theorem 12 *[5, pag 220] If* $f(x,y) \in C^{(p+1)}([a,b] \times \Omega),\ \Omega \subset \mathbb{R}$ *a* **Runge-Kutta** *scheme it is accurate of order p for a (BVPN) problem. Hence*

$$|y_i - y(x_i)| = \mathbb{O}(h^p),\ 0 \leq i \leq n-1. \tag{2.14}$$

Also at collocation points

$$\xi_{ij},\ i = 0, 1, ...n,\ j = 1, 2, ..., k,$$

it holds

$$\left|y_{ij} - y(\xi_{ij})\right| = \mathbb{O}(h^p) + \mathbb{O}(h_i^{k+1}); 1 \leq j \leq k,\ 0 \leq i \leq n-1,$$

where $h_i = x_{i+1} - x_i,\ h = \max\limits_{0 \leq\ i \leq\ n-1} \{h_i\}$.

Numerov scheme

Is a multi-step method with $p = 4$ [28, pag 461-464] for solving nonlinear (BVPN) problem.

Definition 13 *Let* Δ *be a uniform mesh (2.6) of the interval* $[a,b]$ *with stepsize h given by (2.7). We denote by*

$$h := x_k - x_{k-1}, y_k := y(x_k) \quad and \quad f_k := f(x_k, y_k),\ k = 1, 2, ...n-1,$$

then for nonlinear (BVPN) problem the Numerov *scheme is given by*

$$y_{k+1} - 2y_k + y_{k-1} + \frac{1}{12}h^2(f_{k+1} + 10f_k + f_{k-1});\ k = 1, 2, 3, ...n-1,$$

where the boundary conditions are

$$y_0 = \alpha;\ y_n = \beta.$$

2.2.2 Collocation methods

Let Δ be a arbitrary mesh of the interval $[a,b]$ given by (2.6) and we construct the collocation points like in (2.10).
One renumbers the collocation points, such that the first is $\xi_0 := a,\ \xi_1 := x_0 + h_0\rho_0$, and the last $\xi_{N-1} := x_{n-1} + h_{n-1}\rho_k,\ \xi_N = b$. So the grid Δ becomes

$$\overline{\Delta} := \{a = \xi_0 < \xi_1 < ... < \xi_N = b\}, \tag{2.15}$$

where $N = nk + 2$. We denote by $P_{k,\overline{\Delta},s}$, the set of functions $s(x) \in C^2[a,b]$ which have the following proper ties for any $i = 0,1,2,...,N-1$:

1. $s(x) = s_i(x),\ x \in [\xi_i, \xi_{i+1}]$,
2. $s_i(x)$ are polynomial of degree k.

We wish to find an approximate solution of the (BVPN) problem, having the following form

$$u_{\overline{\Delta}}(x) = \sum_{j=0}^{N} c_j \phi_j(x), \tag{2.16}$$

where $\phi_j(x) \in \mathbb{P}_{k,\overline{\Delta},s},\ j = 0,1,2,...N$ are linearly independent functions, and c_j are real parameters. These parameters are determined so that $u_{\overline{\Delta}}(x)$ satisfy the differential equation (2.2a) in $(N-1)$ points

$$\xi_i \in \overline{\Delta},\ i = 1,2,...,N-1,$$

and satisfies the boundary conditions

$$u_{\overline{\Delta}}(a) = \alpha,\ u_{\overline{\Delta}}(b) = \beta.$$

We impose the condition as for every $j = 0,1,2,...n,\ \phi_j(x)$ and any linear combination of these are $C^1[a,b]$ class (like the exact solution of (BVPN) problem).

In this paper we use for $\{\phi_j(x),\ j = 0,1,2,...N\ \}$ the following functions:

1. B-splines of order $k+1$ (degree k),
2. *Chebyshev* orthogonal polynomials of first kind.

B-splines of order k+1 (degree k)

For reason of efficiency, stability, flexibility in order and continuity, we choose B-splines as basis functions. Efficient algorithms for calculating with are given by *deBoor (*[11], [12]) and *Riesler* [48, page 18]. Let the grid $\overline{\Delta}$ given by (2.15) for $x \in [a,b]$ and $0 \le i \le N-k-1$, we define **B-splines functions of order k+1 (degree k)** so:

$$\begin{cases} B_{i,0}(x) = \begin{cases} 1 & \text{if } \xi_i \le x < \xi_{i+1},\ i = 0,1,...,N-1 \\ 0 & \text{otherwise} \end{cases} \\ B_{i,k}(x) = \frac{x-\xi_{\ i}}{\xi_{i+k}-\xi_i} B_{i,k-1}(x) + \frac{\xi_{i+k+1}-x}{\xi_{i+k+1}-\xi_{i+1}} B_{i+1,k-1}(x), \\ \qquad \text{if } \xi_i < \xi_{i+k} \text{ and } \xi_{i+1} < \xi_{i+k+1}. \end{cases} \tag{2.17}$$

We recall some properties of **B-splines functions of order k+1 (degree k)**:

Proposition 14 *[48, teorema 1.4.3, pag. 25] With the above notations:*

1. $B_{i,k}(x)$ *is a polynomial of degree* k,
2. $B_{i,k}(x) = 0$ *for* $x \notin \left[\xi_i, \xi_{i+k+1}\right]$,
3. $B_{i,k}(x) > 0$ *for* $x \in (\xi_i, \xi_{i+k+1})$,
4. $B_{i,k}(x)$ *form the basis in* $\mathbb{P}_{k,\overline{\Delta},s}$.

Definition 15 *[5, pag 218] Given a grid* $\overline{\Delta}$ *of the interval* $[a, b]$ *(2.15), a collocation approximate solution in k stages given by (2.16) is continuous piecewise polynomial function*

$$u_{\overline{\Delta}}(x) = \sum_{i=0}^{n} c_i B_{i,k}(x), \tag{2.18}$$

which reduces to a polynomial of degree at most k (order $k+1$*) on each mesh subinterval and satisfies (2.2a) in grid* $\overline{\Delta}$ *and boundary conditions.*

Since in this work we used combined methods: collocation method with B-splines functions and *Runge-Kutta* methods we recall some results [5, pp: 210-219].

Theorem 16 *[5,* ***Theorem 5.73, page 219****] Given a mesh* $\overline{\Delta}$ *of the interval* $[a, b]$ *given of (2.15), there is only one step explicit Runge-Kutta method with k stages (2.12) equivalent to the collocation method (2.18), (i.e. they have same order of convergence). Moreover*

$$u_{\Delta}(x_i) = y_i,\ u_{\Delta}(\xi_{ij}) = y_{ij},\ 0 \le i \le n\ , 1 \le j \le k,$$

where $x_i \in \Delta$, *and* ξ_{ij} *are collocation points given by (2.10).*

If $y(x)$ is the exact solution of (BVPL) problem and $u_{\overline{\Delta}}(x)$ is approximate collocation solution in k stages we denote by

$$e_{\overline{\Delta}}(x) := u_{\overline{\Delta}}(x) - y(x). \tag{2.19}$$

Theorem 17 *[5,* ***Theorem 5.75****, page 219] The k−stage collocation approximate solution exist for boundary nonlinear value problems (BVPN) and it is obtained in a stable way. Moreover, the error and its derivatives satisfy for every* $i = 0, 1, ..., n$

$$\max_{x\ i \le x \le x_{i+1}} |e_{\Delta}(x)| = \mathbb{O}(h^k),$$

$$\max_{x\ i \le x \le x_{i+1}} \left|e_{\Delta}^{(j)}(x)\right| = \mathbb{O}(h^{k+1-j}) \cdot \theta_i^{j-1},\ 1 \le j \le k,$$

where

$$\theta_i := \frac{h}{h_i},\ h_i = x_{i+1} - x_i,\ h = \max_{0 \le\ i \le\ n-1}\{h_i\},$$

and $e_{\Delta}^{(j)}(x)$ *is derivative of order* j.

Collocation spectral methods

1. Chebyshev polynomials-General properties. In their monograph [22], *Fox* and *Parker* collected the underlying principles of the *Chebyshev* theory. The polynomials whose properties and applications will be discussed were introduced more than a century ago by the Russian mathematician *P.L. Chebyshev* (1821-1894). Their importance for numerical analysis was rediscovered around the middle of the 20−th century by *C.Lanczos* [34] .

Definition 18 *The polynomials* $T_k(x),\ k \in \mathbb{N}$ *defined by:*

$$T_k(x) = \cos(k \arccos(x)),\ -1 \leq x \leq 1,\ k = 0, 1, ...N, \tag{2.20}$$

are called the Chebyshev *polynomials of the first kind.*

Definition 19 *The polynomials* $U_k(x),\ k \in \mathbb{N}$ *defined by*

$$U_k(x) = \frac{\sin((k+1) \cdot \arccos(x))}{(k+1) \cdot \sqrt{1-x^2}}, x \in [-1, 1]\,;\ k = 0, 1, ...N, \tag{2.21}$$

are called the Chebyshev *polynomials of the second kind.*

Proposition 20 *(***Orthogonality***) [24, page: 6] The polynomials* $T_k(x),\ k \in \mathbb{N}$ *are ortho gonal i.e,*

$$(T_n, T_m)_w = \frac{\pi}{2} \cdot c_n^1 \cdot \delta_{n,m};\ n, m \in \mathbb{N}, \tag{2.22}$$

relative to ***weight function***

$$w : I = [-1, 1] -> \mathbb{R}, \quad w\ (x) := \frac{1}{\sqrt[2]{1-x^2}} \tag{2.23}$$

where $\delta_{n,m}$ *stands for the* Kronecker *delta symbol and, throughout in this work, the coefficients are defined by*

$$c_n^1 := \begin{cases} 0, & \text{if } n < 0 \\ 2, & \text{if } n = 0 \\ 1, & \text{if } n \geq 1 \end{cases} \tag{2.24}$$

Definition 21 *[35, pp: 5-6] We say that the functions :*

$$\phi_0, ..., \phi_N : [a, b] \to \mathbb{R},$$

form a Chebyshev *system (or T-system) of order* N *on* $[a, b]$*, if any nonzero linear combination of these functions*

$$\sum_{k=0}^{N} \alpha_k \phi_k, \quad \left(\sum_{j=0}^{N} \alpha_j^2 > 0 \right),$$

has at most N *zeros on* $[a, b]$.

Remark 22 *Using the results, [35, page: 8] it follows that the set of* Chebyshev *orthogonal polynomials* $T_k(x),\ k = 0, 1, 2, ...N$*, form a* Chebyshev *system of order* N *on* $[-1, 1]$ *and moreover for every function:*

$$u(x) \in L_w^2(I)\ ,$$

where $L^2_w(I)$ *is the set of square integrable functions on* I *relative to weight* w*, can be developed in a* Chebyshev *series*

$$u(x) = \sum_{k=0}^{\infty} \overset{*}{u} \cdot T_k(x), \tag{2.25}$$

where

$$\overset{*}{u} = \frac{2}{\pi c_k^1}(u, T_k)_w,$$

and c_k^1 *are given by the relationship (2.24).*

2.Collocation derivative operator. Suppose we know the value of a function at several points (nodes) and we want to approximate its derivative at those points. One way to do this is to find the polynomial that passes through all of data points, differentiate it analytically, and evaluate this derivative at the grid points.

In other words, the derivatives are approximated by exact differentiation of the *Lagrange* interpolate. Since interpolation and differentiation are linear operations the process of obtaining approximations to the values of the derivative of a function at a set of points can be expressed as a matrix-vector multiplication. The matrix involved are called pseudo-spectral differentiation (derivation) matrices or simply differentiation matrices.

Thus, if

$$u := (u(\xi_0), ..., u(\xi_N))^T,$$

is the vector of function values of the grid $\overline{\Delta}$ given by (2.15) and

$$u' := (u'(\xi_0), ..., u'(\xi_N))^T,$$

is the vector of approximate nodal derivatives, obtained by this idea, then there exists a matrix, say $D^{(1)}$ such that

$$u' = D^{(1)}u.$$

We will deduce the matrix $D^{(1)}$ and the next differentiation $D^{(2)}$ defined by

$$u'' = D^{(2)}u.$$

To get the idea we proceed in the simplest way following closely the papers of *A.Sollomonoff* and *E.Turkel* in ([61], [62]).

Thus if:

$$L_N(x) \quad : \quad = \sum_{k=0}^{N} u(\xi_k) l_k(x), \text{ where} \tag{2.26}$$

$$l_k(x) \quad : \quad = \frac{1}{\alpha_k} \prod_{l=0, l\neq k}^{N} (x - \xi_l), \alpha_k := \prod_{l=0, l\neq k}^{N} (\xi_k - \xi_l) \tag{2.27}$$

We shall explicitly construct $D^{(1)}$ by demanding that *Lagrangian* basis $\{l_k(x)\}_{k=0}^N$

$$D^{(1)} l_k(\xi_j) = l_k'(\xi_j) \quad j, k = 0, 1, 2, ..., N, \tag{2.28}$$

where

$$(l_k(x))' = l_k(x) \sum_{l=0}^{N} \frac{1}{x - \xi_l}. \tag{2.29}$$

This equality implies the diagonal elements have the form

$$d_{kk}^{(1)} = \sum_{\substack{l=0 \\ l\neq k}}^{N} \frac{1}{\xi_k - \xi_l},\ k = 1, 2, 3, ..., N. \tag{2.30}$$

Using the definition of $l_k(x)$ (2.27), we get the off-diagonal elements so:

$$d_{jk}^{(1)} = \frac{\alpha_j}{\alpha_k(\xi_j - \xi_k)}$$

and matrix $D^{(2)}$ elements [24, page 21] are:

$$d_{jk}^{(2)} = \begin{cases} 2d_{jk}^{(1)}[d_{jj}^{(1)} - \frac{1}{\xi_j - \xi_k}], & \text{if} \quad j \neq k \\ \left[d_{kk}^{(1)}\right]^2 - \sum\limits_{\substack{l=0 \\ l\neq k}}^{N} \frac{1}{(\xi_j - \xi_k)^2}, & \text{if } j = k \end{cases} . \tag{2.31}$$

Spectral methods for o.d.e

The spectral methods are particulary attractive due to the following approximation pro perties. The distance between the exact solution $y(x)$ of the (BVPN) problem and its spectral approximation $u_N(x)$ is of order $\frac{1}{N^s}$ [24, **Remark 27,** page: 29] i.e,

$$\|y(x) - u_N(x)\| \leq \frac{C}{N^s},$$

where the exponent s depends only on the regularity (smoothness) of the exact solution $y(x)$. Moreover, if $y(x)$ is infinitely derivable, the above distance vanishes faster than any power of $1/N$, and this means spectral accuracy.

Let's consider the partition of the interval $[0, 1]$ so

$$0 = \xi_0 < \xi_1 < ... < \xi_N = 1,$$

where:

$$\xi_j = \frac{-\cos\frac{\pi j}{N} + 1}{2},\ j = 0, 1, ..., N.$$

We observe that the nodes are symmetrically located around $1/2$ if N is even.

This nodes are called *Gauss-Lobatto -Chebyshev* points (shortly *G.L.C)*. In this work we try to find the approximate solution $u_N(x)$ of the exact solution $y(x)$ of the (PVPN) or (PVPL) problem so:

$$u_N(x) = \sum_{k=0}^{N} u_j l_k(x), N \in \mathbb{N},$$

where $u_j := u_N(\xi_j)$ which satisfies (2.2a) in collocation points ξ_j, $j = 1, 2, ..., N-1$ and boundary conditions: $u(\xi_0) = \alpha,\ u(\xi_N) = \beta$.

2.2.3 Examples of technical problems

1. Reaction diffusion equation (linear form) [41] In the paper [36] *P.A.Domenico, F.W. Schwartz,* described the propagation of heat from chemical reactions through a bar of given length L, the axis Ox is chosen along the bar, which occupies the interval $[0, L]$. We consider the problem of non-stationary heat transfer in the bar, and if $L = 1$ we take the heat source as follows:

$$s(x) = 10e^{-5x^{\ 2}}.$$

The problem may change to the advection-diffusion equation on introducing the advection speed $U(x)$, constant or variable and considering the case of stationary and apply to the transport of pollutants in rivers, where $U(x)$ is the local speed of transport.

In this paper we consider the case $U(x) = m(constant)$. We treated the case $m = 4$, given a differential equation whose solution is non-oscillatory

$$-z'' + 4z'(x) = 10e^{(-5z^{\ 2})};\ 0 < z < 1 \tag{2.32}$$

Using a substitution of the form [49, page: 77]

$$z(x) := \varphi(x)y(x),$$

where

$$\varphi(x) = e^{\frac{1}{2}\int_{x_0}^{x} p(s)ds}, \varphi(x) \neq 0,$$

differential equation (2.32) can be transformed as follows

$$-y'' + 4y(x) = 10e^{-5x^{\ 2}-2x}; 0 < x < 1. \tag{2.33}$$

2. Burden and Faires problem [14, problem 5, page: 561] This problem appears in the resistance structure calculation of certain construction:

$$\begin{cases} -y'' - y = x, & x \in (0,1) \\ y\left(\frac{1}{4}\right) = \frac{\sin\frac{1}{4}}{\sin 1} - \frac{1}{4},\ y\left(\frac{1}{2}\right) = \frac{\sin\frac{1}{2}}{\sin 1} - \frac{1}{2} \end{cases}. \tag{2.34}$$

3. Frequency domain equation for the vibrating string. This problem is considered in the paper of *Greengard and Rokhlin* [27, page: 443]:

$$\begin{cases} y''(x) + (k^2+5)y(x) = 5\sin(kx),\ 0 < x < 1 \\ y(c) = \sin(kc),\ y(d) = \sin(kd),\ 0 < c < d < 1 \end{cases}, \tag{2.35}$$

The exact solution is: u(x)=sin(px). Here we solve a problem using B-splines functions and Chebyshev Collocation method of type which arises when dealing with a frequency domain equation for vibrating string and we compare the costs (Run-times and Internal Memory). In order to demonstrate the performance of B-splines method on large scale oscillatory cases, we solved the problem for p ranging from 9 to 630 and n between 10 and 1200. We observed that reliable results can only be expected under the assumption of scale resolution, is small, i.e that $p/n < 1$. In this situation the error in approximating the exact solution

$$u(x) = sin(px) \tag{2.36}$$

is of order $O(h^p)$. We treated the case: c=Pi/54, d=Pi/12, $k = 9$.

4. Bratu's problem ([24], [58], [59]). Spontaneous combustion occurs in the model

$$\begin{cases} -y'' + \lambda e^u = 0, \ 0 < x < 1, \\ y(0) = y(1) = 0, \end{cases} \tag{2.37}$$

where $\lambda > 0$ is a parameter. Substituting a function of the form:

$$y(x) = -2\log(\cosh((x - 0.5)\frac{\theta}{2})/\cosh(\frac{\theta}{4}),$$

which satisfies the boundary conditions, into differential equation, we find that $y(x)$ is a solution if:

$$\theta = \sqrt{2\lambda}\sinh\left(\frac{\theta}{4}\right). \tag{2.38}$$

This nonlinear algebraic equation for θ has two, one or no solutions when $\lambda < \lambda_c, \lambda = \lambda_c$ or $\lambda > \lambda_c$, respectively. The critical value λ_c satisfies :

$$1 = \frac{1}{4}\sqrt[2]{2\lambda_c}\sinh(\frac{\theta}{4})$$

We treated the case $\lambda = \lambda_c = 1$.

A diagram of bifurcation for this problem is also available in the monograph of *U.Ascher, B.Mattheij and R.D.Russel* [5, page: 491].

5. The average temperature in a reaction-diffusion process [24, page: 45]

$$\begin{cases} u'' + u^p = 0, 0 < x < L \\ u(0) = u(L) = 0. \end{cases} \tag{2.39}$$

D.Trif, C.Gheorghiu [25] prove that, for $p = 3$, the boundary value problem (2.39) has a unique positive solution and solve it using spectral *Galerkin* methods, convergence rate being $\mathbb{O}(10^{-8})$.The authors use Newton quasilinearization method, taking as starting solution the function

$$u_0 := (1 - x^2)^2;$$

further they demonstrates that this iterative process is convergent.

Remark 23 *For different values of parameter p the problem (2.39) may have a unique positive solution or multiple oscillatory solutions. Bifurcation diagram of this problem is found in the work [5, pag 491].*

6. The charge density in atoms of high atomic number. The Thomas-Fermi equation,

$$y'' = x^{-1/2}y^{3/2}, \tag{2.40}$$

is to be solved with boundary conditions

$$y(0) = 1, \quad y(\infty) = 0. \tag{2.41}$$

This (BVP) arises in a semiclassical description of the charge density in atoms of high atomic number. There are difficulties at both end points; these difficulties are discussed at length in *Davis* (1962) and in *Bender and Orszag* (1999). *Davis* discusses series solutions for

$$y(x) \text{ as } x \to 0.$$

It is clear that there are fractional powers in the series. That is because, with $y(0) = 1$, ODE requires that

$$y(x) \sim x^{-1/2} \text{ as } x \to 0,$$

and hence there be a term $\frac{4}{3}x^{3/2}$ in series for $y(x)$. Of course, there must also be lower-order terms so as to satisfy the boundary condition at $x = 0$.*Bender & Orszag* discuss the asymptotic behavior of $y(x)$ as $x \to 0$. Verify that trying a solution of the form

$$y''(x) \sim ax^{\alpha},$$

yields

$$y_0(x) = 144x^{-3},$$

as an exact solution of the ODE that satisfies the boundary condition at infinity.

Chapter 3

Linear form of boundary values problems

3.1 Formulating the problem

We consider the linear boundary value problem with conditions inside the interval $[0,1]$ (PVPL)

$$\begin{cases} -y''(x)+q(x)y(x)=r(x), & x\in[0,1] \\ y(a)=\alpha,\ y(b)=\beta, & a,b\in(0,1), \end{cases}, \tag{3.1}$$

where $q(x),\ r(x)\in C[0,1];\ y\in C^1[0,1],\ a,b,\ \alpha,\beta\in\mathbb{R}$.

If the solution of two points boundary value problem (BVPL)

$$\begin{cases} -y''(x)+q(x)y(x)=r(x), & x\in[a,b] \\ y(a)=\alpha,\ y(b)=\beta, \end{cases}, \tag{3.2}$$

exists and it is unique, then the requirement $y(x)\in C^1[0,1]$ assures the existence and the uniqueness exact solution of the problem (3.2).

We present three approximation methods:

1. Global method with cubic B-spline function,
2. C.C method,
3. Combined method *Runge-Kutta* and B-spline functions of degree k.

3.2 Global method with cubic B-spline function

Consider the linear boundary value problem (PVPL) given by (3.1) and further consider that occurs the conditions of existence and uniqueness of exact solutions $y(x)$. We recall some properties of orthogonal *Legendre* [35] polynomials.

Definition 24 *We define the orthogonal* Legendre *polynomials as follows*

$$P_n(y)=\frac{1}{2^n n!}\frac{d^n}{dy^n}(y^2-1)^n; n=0,1,2,...$$

Proposition 25 *The roots of orthogonal* Legendre *polynomials are contained in* $(-1,1)$.

Using the affine transformation

$$y := 2x - 1$$

we obtain the orthogonal *Legendre* polynomials on $(0,1)$ having all roots within $(0,1)$:

$$\bar{P}_n(x) = \frac{1}{2^n n!}\frac{d^n}{dx^n}[x^n(x-1)^n];\ n = 0,1,2,...$$

3.2.1 Construction of collocation points and the base of cubic B-spline functions

We construct the partition of $[0,1]$ like in (2.6), the step-size (2.7) and collocation points (2.10).

Collocation points form the following matrix

$$\Upsilon = \begin{bmatrix} h_0\rho_1 & h_0\rho_2 & \dots & h_0\rho_k \\ x_1 + h_1\rho_1 & x_1 + h_1\rho_2 & \dots & x_1 + h_1\rho_k \\ \vdots & \vdots & \ddots & \vdots \\ x_{n-1} + h_{n-1}\rho_1 & x_{n-1} + h_{n-1}\rho_2 & \dots & x_{n-1} + h_{n-1}\rho_k \end{bmatrix}$$

We note that the points x_i of grid (2.6) are not collocation points, so you must insert their heads interval, i.e. 0 and 1, and inside a, b and so we get $nk+4$ points .

We renumber the collocation points such that

$$\xi_0 = 0,\ \xi_1 = h_0\rho_1, \xi_2 = h_0\rho_2, ...\xi_{N+1} = x_{n-1} + h_{n-1}\rho_k,\ \xi_{N+2} = 1,$$

where $N = nk+2$. Therefore, the partition of $[0,1]$ becomes

$$\Delta := \left\{0 = \xi_0 < \xi_1 < ... < a... < b... < \xi_{N+2} = 1\right\}.$$

We augment the above partition Δ [60] to form

$$\overline{\Delta} : \xi_{-2} < \xi_{-1} < \xi_0 = 0 < \xi_1 < ... < \xi_{N+2} = 1 < \xi_{N+3} < \xi_{N+4}, \tag{3.3}$$

where

$$\begin{aligned} \xi_l &= a; \quad \xi_{l+p} = b;\ 0 < l < N+1;\ 1 < l+p < N+2, \\ \xi_{-1} - \xi_{-2} &= \xi_0 - \xi_{-1} = \xi_1 - \xi_0, \\ \xi_{N+4} - \xi_{N+3} &= \xi_{N+3} - \xi_{N+2} = \xi_{N+2} - \xi_{N+1.} \end{aligned}$$

To simplify the presentation, we use the abbreviations

$$q_i := q(\xi_i)\ ;\ h_i := \xi_{i+1} - \xi_i;\ h := \max_{0 \le\ i \le\ n} h_i;\ \underline{h} := \min_{0 \le\ i \le\ n} h_i.$$

Definition 26 *For* $\forall\ x \in [0,1]$ *and* $\forall\ 0 \le i \le N+1$, *the cubic B-spline functions with the five knots :* ξ_i, ξ_{i+1}, ξ_{i+2}, ξ_{i+3}, ξ_{i+4} *are given by*

$$\begin{cases} B_{i,0}(x) = \begin{cases} 1 & if \quad \xi_i \le x < \xi_{i+1} \\ 0 & if \quad x \in [0,1] - [\xi_i, \xi_{i+1}) \end{cases} \\ B_{i,3}(x) = \frac{x - \xi_{\ i-2}}{h_{i-2}+h_{i-1}+h_i} B_{i,2}(x) + \frac{\xi_{\ i+2} - x}{h_{i+1}+h_i+h_{i-1}} B_{i+1,2}(x) \\ if\ \ \xi_i < \xi_{i+3} \quad and\ \xi_{i+1} < \xi_{i+4} \end{cases} \tag{3.4}$$

3.2.2 The existence and uniqueness of approximate solution

We wish to find an approximate solution of the boundary value problem (PVPL) (3.1) in the following form

$$u_{\overline{\Delta}}(x) = \sum_{i=\ 0}^{N+2} c_i B_{i,3}(x), \tag{3.5}$$

where $B_{i,3}(x)$ is a cubic B-spline function with knots $\{\xi_{i+k}\}_{k\ =\ -2}^{2}$.

We impose the conditions:

(c1) The approximate solution $u_{\overline{\Delta}}(x)$ given by (3.5) verifies the differential equation of (3.1) at

$$\overline{\Delta} := \{\xi_j\ ,\ j = 1, ..., N+1,\ j \neq l,\ j \neq l+p\}.$$

(**c2**) The approximate solution $u_{\overline{\Delta}}(x)$ (3.5) verifies

$$u_{\overline{\Delta}}(\xi_l) = \alpha,\ u_{\overline{\Delta}}(\xi_{l+p}) = \beta.$$

We recall that $a = \xi_l$ and $b = \xi_{l+p}$.

Conditions **(c1)** and **(c2)** yield to a linear system with $N+3$ equations and $N+3$ unknows $c_i,\ i = 0, ..., N+2$:

$$A \cdot c = \gamma. \tag{3.6}$$

We denote by

$$f_i(x) := B''_{i,3}(x) - q(x)B_{i,3}(x), i = 0, 1, ...N,$$

then

$$A = \begin{bmatrix} f_i(\xi_j);\ i \in \{0, 1, 2..., N+2\} \quad j \in \{1, 2, ...N+1\} \backslash \{l.l+p\} \\ B_{i,3}(\xi_l); i = l-1, l, l+1 \\ B_{i,3}(\xi_{l+p}); i = l+p-1, l+p, l+p+1 \end{bmatrix}. \tag{3.7}$$

The system matrix A is tridiagonal with 3 nonzero elements on each row. The right hand side of (3.6) is

$$\gamma = [r(\xi_1), ...r(\xi_{l-1}), \gamma_1, r\left(\xi_{l+1}\right), ...r(\xi_{l+p-1}), \gamma_2, r(\xi_{l+p+1}), ...r(\xi_{n+2})].$$

Lemma 27 *[48, pag 23]For every $l > 0,\ x \in [0,1],\quad B_{i,l}(x) \in C^1[0,1]$ occurs*

$$B'_{i,l}(x) = l\ [\frac{B_{i,l-1}(x)}{\xi_{i+l} - \xi_i} - \frac{B_{i+1,l-1}(x)}{\xi_{i+l+1} - \xi_{i+1}}]. \tag{3.8}$$

Lemma 28 ***(Daniel N.Pop [42])*** *For $\forall\ x \in [0,1]$ and $B_{i,3}(x) \in C^2[0,1]\ 0 \leq i \leq N+2$ then*

$$\begin{aligned} B''_{i,3}(x)\ &= 3![\frac{B_{i,1}(x)}{(h_i + h_{i+1} + h_{i+2})(h_{i+1} + h_i)} - \\ &- \frac{B_{i+1,1}(x)(h_{i+2} + 2h_{i+1} + 2h_{i+2} + h_{i+3})}{(h_{i+2} + h_{i+1})(h_{i+2} + h_{i+1} + h_i)(h_{i+3} + h_{i+2} + h_{i+1})} + \\ &+ \frac{B_{i+2,1}(x)}{(h_{i+3} + h_{i+2} + h_{i+1})(h_{i+2} + h_{i+3})}]. \end{aligned} \tag{3.9}$$

Proof. For $l = 3$ we get from (3.8)

$$B'_{i,3}(x) = 3[\frac{B_{i,2}(x)}{h_{i+2} + h_{i+1} + h_i} - \frac{B_{i+1,2}(x)}{(h_{i+3} + h_{i+2} + h_{i+1}}],$$

then

$$B''_{i,3}(x) = 3\left[\frac{B'_{i,2}(x)}{h_{i+2} + h_{i+1} + h_i} - \frac{B'_{i+1,2}(x)}{h_{i+3} + h_{i+2} + h_{i+1}}\right]. \tag{3.10}$$

Using again (3.8) for $l = 2$ it results

$$B'_{i,2}(x) = 2\left[\frac{B_{i,1}(x)}{h_{i+1} + h_i} - \frac{B_{i+1,1}(x)}{h_{i+2} + h_{i+1}}\right], \tag{3.11}$$

$$B'_{i+1,2}(x) = 2\left[\frac{B_{i+1,1}(x)}{h_{i+2} + h_{i+1}} - \frac{B_{i+2,1}(x)}{h_{i+3} + h_{i+2}}\right]. \tag{3.12}$$

Replacing (3.11) and (3.12) in (3.10) we get (3.9). ■

Theorem 29 ***(Daniel N.Pop)[42]*** *If for every $x \in [0, 1]$*

$$q(x) < K \leq 0, \tag{3.13}$$

then the matrix A given by (3.7) is nonsingular.

Proof. In the following we denote by

$$f_i(x) := B''_{i,3}(x) - K \cdot B_{i,3}(x),\ \forall\ i = 0, 1, ...N + 2. \tag{3.14}$$

Since $B_{i,3}(x) > 0$ for every $x \in (\xi_{j+1}, \xi_{j+3})$ [48, pp: 18-20] resulting

$$\begin{aligned} B_{i,3}(\xi_{j+1}) &> 0,\ B_{i,3}(\xi_{j+2}) > 0,\ B_{i,3}(\xi_{j+3}) > 0. \\ B_{i,3}(\xi_{l+p}) &> 0; i = l + p - 1, l + p, l + p + 1. \end{aligned} \tag{3.15}$$

From (3.4) and (3.9) we obtained the following inequalities

$$\begin{aligned} B_{i,3}(\xi_{j+2}) &> B_{i,3}(\xi_{j+1}) + B_{i,3}(\xi_{j+3}),\ j = -2, -1, 0, 1, ...N, \\ B''_{i,3}(\xi_{j+1}) &= \frac{3!}{(h_{j+2} + h_{j+1} + h_j)(h_{j+1} + h_j)} > 0, \\ B''_{i,3}(\xi_j) &= \frac{3!(h_j + 2h_{j+1} + 2h_{j+2} + h_{j+3)}}{(h_{j+2} + h_{j+1} + h_j)(h_{j+2} + h_{j+1})(h_{j+3} + h_{j+2} + h_{j+1})} > 0, \\ B''_{i,3}(\xi_{j+3}) &= \frac{3!}{(h_{j+3} + h_{j+2} + h_{j+1})(h_{j+2} + h_{j+3})} > 0, \\ B''_{i,3}(\xi_{j+2}) &> B''_{i,3}(\xi_{j+1}) + B''_{i,3}(\xi_{j+3}),\ j = -2, -1, 0, 1, ...N. \end{aligned} \tag{3.16}$$

Using again (3.14) and (3.16) we get

$$f_i(\xi_{j+2}) > f_i(\xi_{j+1}) + f_i(\xi_{j+3}),\ \forall j = -2, -1, 0, 1, ...N.$$

From the above relationship we deduce that collocation matrix A given by (3.7) is strictly diagonally dominant, so nonsingular and $\det A \neq 0$.

So the coefficients c_i from the relation (3.5) can be determined uniquely. ■

To solve the system (3.6) we use *Crout*'s factorization algorithm for linear tridiagonal systems [14, page: 353], this algorithm requires only $(5N + 11)$ multiplication/division and $(3N + 6)$ addition / substraction and therefore has considerable computational advantages compared with methods, where the matrix are not tridiagonal and irreducible, especially for large values of N.

Conclusion 30 *In terms of costs (time U.C and internal memory), effectiveness of this method with cubic B-spline methods in comparison with a situation arises in pseudo-spectral oscillatory solutions, as this that collocation matrix is dense.*

3.2.3 Error study

Below we expose some considerations on the study of the error.

Theorem 31 *[42] If $y(x) \in C^1[0,1]$ is the exact solution of boundary value problem (PVPL), then there exits a unique cubic B-spline function*

$$u_{\overline{\Delta}}(x) \in S(\overline{\Delta}),$$

where

$S\left(\overline{\Delta}\right) = \{p(x) \in C^2[0,1]/p(x)$ *is a polynomial function of degree three for each subintervals*: $[\xi_{i-2},\ \xi_{i+2}];\ 0 \leq i \leq N+2\}$,*which verifies*

$$\max_{\xi_i \leq x \leq \xi_{i+4}} \left|y(x) - u_{\overline{\Delta}}(x))\right| \leq K \cdot H^2 \cdot \left\|y^{(2)}\right\|_{[\xi_i, \xi_{i+4}]},\ i = -2,-1,0,1,...,N, \tag{3.17}$$

where

$$H := \max\{h_i, h_{i+1}, h_{i+2}, h_{i+3}\},$$

and K is real constant independent of $\overline{\Delta}$ given by (3.3).

Proof. Using the results of *U. Ascher* [4] and *C.deBoor* [12], convergence of the method presented in the previous paragraph takes place for every division

$$\xi_j \in \overline{\Delta},\ j \in \{-2,-1,0,1,...,N+4\} - \{l, l+p\}$$

and more:

$$\left|y^{(\upsilon)}(\xi_j) - u^{(\upsilon)}_{\overline{\Delta}}(\xi_j)\right| = \mathbb{O}(H^{2k}), \forall\ 0 \leq \upsilon \leq 2,\ \forall\ k \geq 2, \tag{3.18}$$

where υ means order derivative index and the index k is the degree of *Legendre* polynomial.

More in points $\xi_l = a,\ \xi_{l+p} = b,$ according [5, page 222] occurs

$$\begin{aligned} |y(\xi_l) - u_{\overline{\Delta}}(\xi_l)|_{[\xi_{l-2}, \xi_{l+2}]} &\leq K_1 \cdot H^2 \cdot \left\|y^{(2)}\right\|_{[\xi_{l-2}, \xi_{l+2}]}, \\ \left|y(\xi_{l+p}) - u_{\overline{\Delta}}(\xi_{l+p})\right|_{[\xi_{l+p-2}, \xi_{l+p+2}]} &\leq K_2 \cdot H^2 \cdot \left\|y^{(2)}\right\|_{[\xi_{l+p-2}, \xi_{l+p+2}]}, \end{aligned} \tag{3.19}$$

where K_1, K_2 are real constants independent of division $\overline{\Delta}$ given by (3.3) and $y(x) \in C^1[0,1]$ the exact solution of the bylocal linear problem (PVPL). Using the relations (3.18) and (3.19), that the proposed method is convergent with order of convergence $\mathbb{O}(H^2)$. ■

3.2.4 Numerical results

In the works [43], [44], we prove the effectiveness of the method presented in above paragraph for the case $q(x) < K \leq 0,$ by two examples: one with oscillatory solution and other with non-oscillatory solution. For implementation of costs (run times and internal memory) we used a pair of commands `Maple 13` ([2, page: 236], [29]) `profile-show-profile`, and for error control the command `plots[log-plot]`.

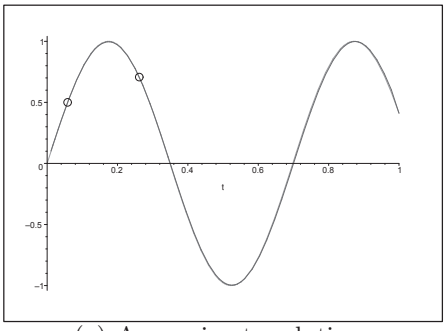
To make a comparative study of computational costs and errors we took two different divisions of the interval $[0,1]$ so : $N \in \{10, 36\}$.

1. Greengard and Rokhlin problem (oscillatory solution) We solve the bylocal problem (2.35) for $k = 9$ using *cubic B-spline* functions on two meshes $\overline{\Delta}$ of the interval $(0, 1)$ given by (3.3). We obtain the following results

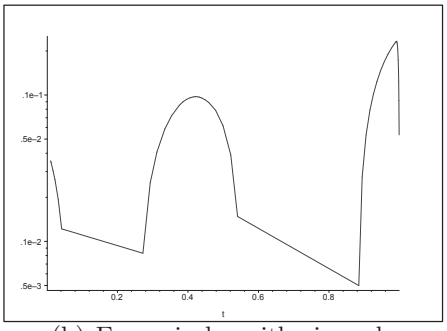
N	Run times	Internal memory	Tolerance
10	2.955	64944668	$0.5e^{-1}$
36	12.668	132688208	e^{-2}

.

For $N = 36$ we represented approximate solution in Figure 3.1(a) and errors in logarithmic scale in Figure 3.1(b).



(a) Approximate solution



(b) Errors in logarithmic scale

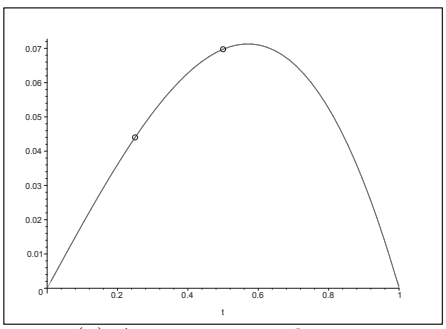
Figure 3.1: Greengard-Rokhlin problem, non-uniform mesh, Cubic B-spline

2. Burden and Faires problem(non-oscillatory solution) We solve the two-points boundary value linear problem (2.34) using *cubic B-spline* functions on two meshes $\overline{\Delta}$ of the interval $(0, 1)$ given by (3.3). We obtain the following results

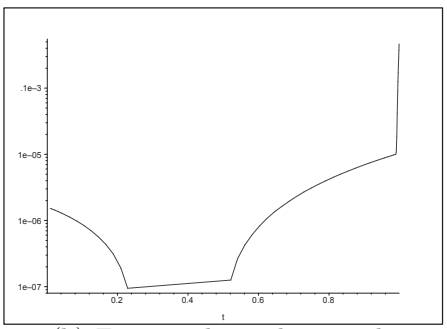
N	Run times	Internal memory	Tolerance
10	2.724	68700548	$0.1e^{-25}$
36	11818	235934564	$0.1e^{-40}$

.

For $N = 36$ we represented approximate solutions in Figure 3.2(a) and errors in logarithmic scale in Figure 3.2(b).



(a) Approximate solution



(b) Errors in logarithmic scale

Figure 3.2: Burden-Faires problem, non-uniform mesh, Cubic B-spline

3.3 Pseudo-spectral (C.C) method

Formulating the problem

We consider the linear boundary value problem (PVPL) with conditions inside the interval $[0,1]$. This problem will be solved numerically by dividing them into three two-points boundary value problems as follows:

- (BVPL1)

$$\begin{cases} -y''(x)+q(x)y(x)=r(x),\ a\le x\le b, \\ y(a)=\alpha,\ y(b)=\beta \\ a,b,\alpha,\beta\in\mathbb{R}, \end{cases}, \tag{3.20}$$

- (BVPL2)

$$\begin{cases} -y''(x)+q(x)y(x)=r(x),\ 0\le x\le a, \\ y(a)=\alpha,\ y'(a)=\gamma,\ a,\alpha,\gamma\in\mathbb{R}, \end{cases}, \tag{3.21}$$

- (BVPL3)

$$\begin{cases} -y''(x)+q(x)y(x)=r(x),\ b\le x\le 1, \\ y(b)=\beta,\ y'(b)=\delta,\ b,\beta,\delta\in\mathbb{R}. \end{cases} \tag{3.22}$$

We determined the approximate solutions of the problems (3.20), (3.21) and (3.22) using the $C.C$ method [24, page: 36-38].

Choosing the mesh of the interval $[0,1]$ and define the *Chebyshev'* polynomials of first kind on $[a,b]$.

Definition 32 *If* $T_k(x)=\cos(k\arccos(x))\ \ k=0,1,2,..N$ *are the* Chebyshev' *polynomials* of first kind *on* $[-1,1]$, *we define the* Chebyshev' *polynomials* of first kind *on* $[a,b]$ *so*

$$T_k(\frac{(b-a)x+(a+b)}{2}):=\cos(k\arccos\frac{(b-a)x+(a+b)}{2}) \tag{3.23}$$

We choose to solve numerically the bylocal linear problem (PVPL) the following mesh of the interval $[0,1]$

1. on $[0,a]$

$$x_j^1=\frac{a}{2}(-\cos\frac{\pi j}{N}+1),\ j=0,1,...,N, \tag{3.24}$$

2. on $[a,b]$

$$x_j^2=\frac{(b-a)\cos\frac{\pi j}{N}+a+b)}{2},\ j=0,1,...,N, \tag{3.25}$$

3. and on $[b,1]$

$$x_j^3=\frac{(b-1)\cos\frac{\pi j}{N}+1+b)}{2},\ j=0,1,...,N. \tag{3.26}$$

The points of the mesh (3.24) are symmetrical to $\frac{a}{2}$, the points of the mesh(3.25) are symmetrical to $\frac{a+b}{2}$, and those mesh-points (3.26) are symmetrical to $\frac{b+1}{2}$.

Solving numerical two-points boundary value problems (BVPL1), (BVPL2) and (BVPL3).

In the paper [15], *C.Canuto* suggests to find approximate solution $u_N(x)$ using Lagrange interpolation polynomial. We use *C.C method* for determine the approximate solutions of two-points boundary value problems (BVPL1), (BVPL2) and (BVPL3) on collocation points given by (3.24), (3.25) and (3.26) as shown below.

Determination of the unknowns

$$c_j^2 := u_N(x_j^2),\ j = 0, 1, ..., N,$$

consists in solving the following linear algebraic system

$$\begin{cases} L_N(u_N(x_j^2)) - r((x_j^2)) = 0;\ j = 1, ..., N-1 \\ u_N(a) = \alpha;\ u_N(b) = \beta, \end{cases}, \tag{3.27}$$

where x_j^2 are given by (3.25) and

$$L_N(x) = \sum_{k=1}^{N} u(x_k) l_k(x), \tag{3.28}$$

is the *Lagrangian* interpolation polynomial.

Determination of the unknowns

$$c_j^1 := u_N(x_j^1),\ j = 0, 1, ..., N,$$

consists in solving the following linear algebraic system

$$\begin{cases} L_N(u_N(x_j^1)) - r((x_j^1)) = 0;\ j = 1, ..., N-1 \\ u_N(a) = \alpha;\ u_N(0) = \eta, \end{cases}, \tag{3.29}$$

where x_j^1 are given by (3.24). We build $u_N(0)$ so

$$u_N(0) = \alpha - a u'_N(a),$$

where the value $u'_N(a)$ is obtained by deriving *Lagrange* interpolation polynomial on $x = a$.

$$u'_N(a) = \sum_{j=0}^{N} \frac{1}{a - x_j}. \tag{3.30}$$

Determination of the unknowns

$$c_j^3 := u_N(x_j^3),\ j = 0, 1, ..., N,$$

consists in solving the following linear algebraic system:

$$\begin{cases} L_N(u_N(x_j^3)) - r((x_j^3)) = 0;\ j = 1, ..., N-1 \\ u_N(b) = \beta;\ u_N(1) = \lambda, \end{cases}, \tag{3.31}$$

where x_j^3 are given by (3.26). We build $u_N(1)$ so:

$$u_N(1) = \beta + (1 - b) u'_N(b),$$

where the value $u'_N(b)$ is obtained by deriving *Lagrange* interpolation polynomial on $x = b$.

$$u'_N(b) = \sum_{j=0}^{N} \frac{1}{b - x_j}. \tag{3.32}$$

Proposition 33 *If in collocation matrices of the systems (3.27), (3.29), (3.31) remove lines which are obtained from boundary conditions, first and last column we obtain a centrosymmetric matrices, which according ([17], [69]) are nonsingular.*

Using the above proposition, it follows that if we use the method presented can uniquely determine the values:

$$u_N(x_j^1),\ u_N(x_j^2) \text{ and } u_N(x_j^3).$$

3.3.1 Numerical results

The `Maple` code to determinate the approximate solutions of the two-points boundary value problems (2.34) and (2.35) using $C.C$ methods is in [21]. For establish the running costs (run time and internal memory) we set the pair of commands using Maple 13 `profile-show-profile [29]` and for errors in logarithmic scale we use the command `plots[plot-log ]`.

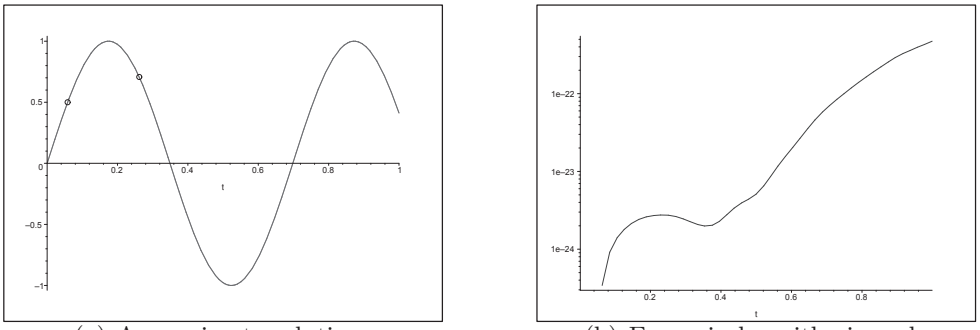
1. Greengard and Rokhlin problem (oscillatory solution)

For $k = 9$ and for different values of division: $N \in \{10, 36\}$ we solve the two-points boundary value problem (2.35) using $C.C$ method on the collocation points of the interval $[0, 1]$ given by (3.24), (3.25) and (3.26).

We got the following results

N	Run times	Internal memory	Tolerance
10	3.925	68943558	$0.1e^{-3}$
36	53.767	997685852	$8e^{-25}$

For $N = 36$ we represented the approximate solution in Figure 3.3(a) and the errors in logarithmic scale in Figure 3.3(b).



(a) Approximate solution (b) Errors in logarithmic scale

Figure 3.3: Greengard-Rokhlin problem, non-uniform-mesh, C.C method

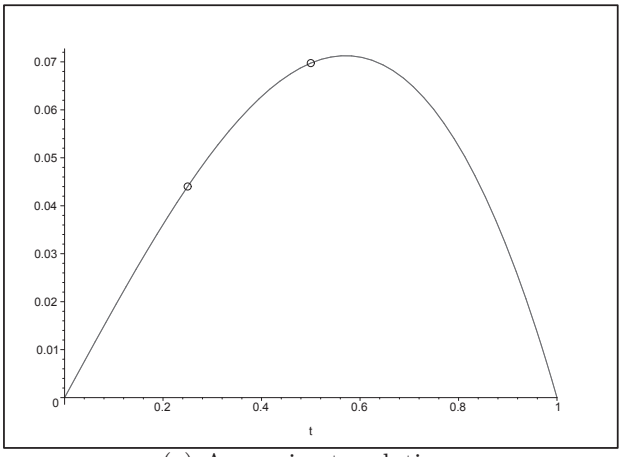
2. Burden and Faires problem

We determined the approximate solution of the problem (2.34) for two values of grid : $N \in \{10, 36\}$ using the collocation points of the interval $[0, 1]$ (3.24), (3.25) and (3.26).

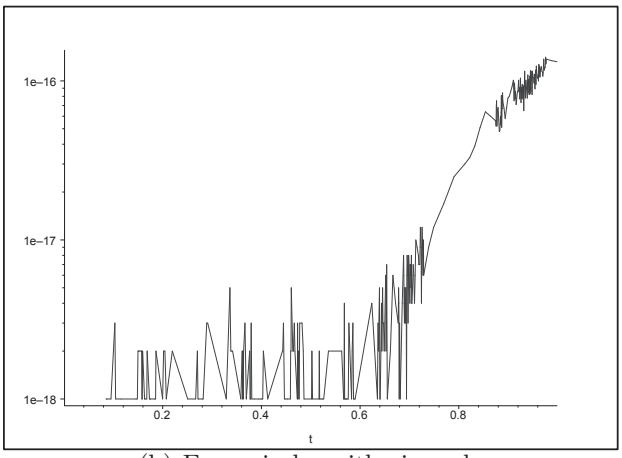
We obtained the following results

N	Run times	Internal memory	Tolerance
10	0.320	8014024	e^{-10}
36	6.469	135614356	e^{-16}

For $N = 36$, we represented the approximate solution in Figure 3.4(a), and the errors in Figure 3.4(b).



(a) Approximate solution



(b) Errors in logarithmic scale

Figure 3.4: Burden-Faires problems, non-uniform-mesh, C.C method

3. Reaction diffusion equation

For equation (2.33), we determined an approximate solution using *C.C* method combined with a pair *Runge-Kutta* method. With conditions inside the interval $(0, 1)$, we obtained the following (BVPL) problem

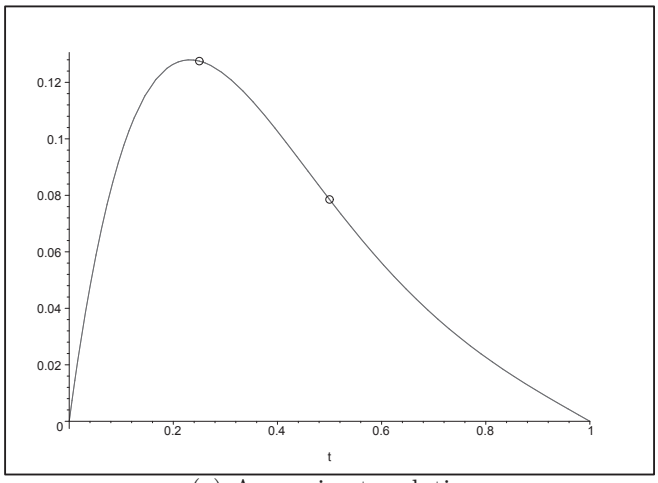
$$\begin{cases} -y'' + 4y(x) = 10e^{-5x\ ^2-2x}; 0 \leq x \leq 1, \\ y(1/4) = \gamma_1,\ y(1/2) = \gamma_2, \end{cases} \tag{3.33}$$

where the values $\gamma_1,\ \gamma_2$ are obtained using Maple 13.

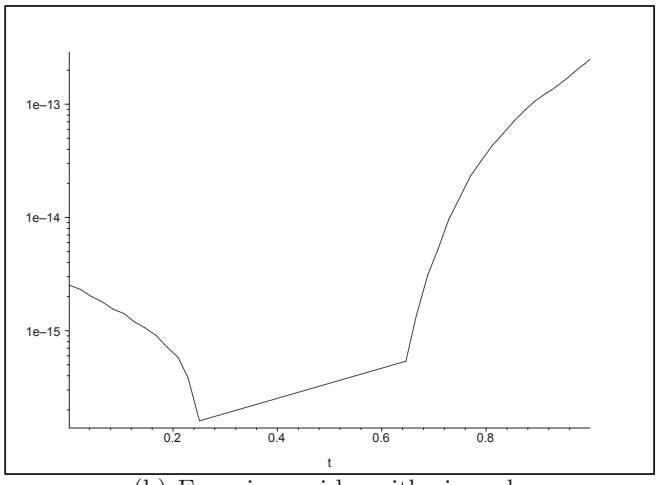
For $N \in \{10, 36\}$, we determined an approximate solution of boundary value problem (3.33), and evaluate the costs (Run times and Internal memory used) and the errors, achieving the results:

N	Run times	Internal memory	Tolerance
10	1.402	23586232	e^{-07}
36	13.610	249685652	e^{-13}

The approximate solution for $N = 36$ is in Figure 3.5(a), and the error in semi-logarithmic scale in Figure 3.5(b).



(a) Approximate solution



(b) Error in semi-logarithmic scale

Figure 3.5: Reaction diffusion equation, C.C method

3.4 Combined Runge-Kutta methods and B-spline functions

Principle method

The two-points boundary values problem (PVPL)(3.1) can be solve by dividing it in three above problems so:

- A two-points *Dirichlet* boundary values problem (BVPL)

$$\begin{cases} -y''(x) + q(x)y(x) - r(x) = 0, & x \in (a,b) \\ y(a) = \alpha,\ y(b) = \beta \end{cases}, \tag{3.34}$$

- Two initial value problems (*Cauchy* problems):

1. (IVP-Left)

$$\begin{cases} y' = z \\ z' = -f(x,y), & x \in [0,a] \\ y(a) = \alpha,\ y'(a) = \gamma \end{cases}, \tag{3.35}$$

2. and (IVP-Right)

$$\begin{cases} y' = z \\ z' = -f(x,y), & x \in [b,1] \\ y(b) = \beta,\ y'(b) = \delta, \end{cases}, \tag{3.36}$$

where $f(x,y) = -q(x)y(x) + r(x)$.

The values $y'(a)$, $y'(b)$ necessary to determine solutions of the problems (IVP-Left) and (IVP-Right) are determined by differentiating the approximate solution of two-points boundary values problem (3.34) in the points $x = a$ and $x = b$.

For the two-points boundary values problem (BVPL) we use the collocation method with B-splines of order $k+1$, $k \in \mathbb{N}^*$ and to determine the approximate solutions of the problems (IVP-Left) and (IVP-Right) we use *Runge-Kutta* methods. We use division $\overline{\Delta}$ of the interval $[a,b]$ given by (3.3), where the multiplicity of points a, b are $k+1$ and inside collocation points have multiplicity k.

We find the approximate solution $u_{\overline{\Delta}}(x)$ in the form

$$u_{\overline{\Delta}}(x) = \sum_{i=0}^{N+2} c_i B_{i,k}(x),$$

where $B_{i,k}(x)$ are B-splines functions of degree k, and real coefficients $c_i \in \mathbb{R}$ are determined by imposing the following conditions:

- **(C1)** the approximate solution $u_{\overline{\Delta}}(x)$ verifies the differential equation given by (3.34) inside the grid $\overline{\Delta}$ (multiplicity k).
- **(C2)** the approximate solution $u_{\overline{\Delta}}(x)$ verifies the conditions

$$u_{\overline{\Delta}}(a) = \alpha, u_{\overline{\Delta}}(b) = \beta,\ a, b \in (0,1),$$

where the points a and b have the multiplicity $k+1$.

The conditions **(C1) +(C2)** lead to a linear system $Ac = \gamma$, with $(N+3)$ equations and $(N+3)$ unknowns.

The collocation matrix is

$$A = [a_{ij}]_{i,\ j=0,...,N+2} = \begin{cases} -B''_{i,k}(\xi_j) + q(\xi_j)B_{i,k}(\xi_j), & \text{for } \ j = 1, 2, ..., N+1, \\ B_{i,k}(a), & \text{for } \ j = 0, \\ B_{i,k}(b), & \text{for } \ j = N+2 \end{cases} . \tag{3.37}$$

The right hand size of this system is

$$\gamma = \left[\gamma_1, r(\xi_1), \ldots, r(\xi_{N-1}), \gamma_2\right]^T .$$

We denote by $\overline{u}_{\overline{\Delta}}(x)$ the approximate solution of the problem (PVPL) (3.1), obtained by combined method B-splines and Runge-Kutta methods.

Theorem 34 ***(Daniel N.Pop)*** *[45] Suppose that there exists $p \geq k \geq 2$, such that:*

(a) The linear problem (3.34) is well-posed, in the sense that the collocation matrix A given by (3.37) has a condition number $\kappa_k \equiv cond(A)$, of moderate size ([63]) and

$$q, r \in C^p[a, b];$$

(b) The linear problem (3.34) has a unique solution and $y(x) \in C^p[a, b]$,

(c) The collocation points ρ_1, ..., ρ_k given by (2.11) satisfy the orthogonality condition

$$\int_0^1 \Phi(x) \prod_{\ell=1}^{k} (x - \rho_\ell) dx = 0, \qquad \Phi \in \mathbb{P}_{p-k}, \ (p \leq 2k),$$

where $\mathbb{P}_{p-k}$ is the set of polynomials at most degree $p-k$.

Then for

$$h = \max_{i=1,...,N} h_i,$$

sufficiently small our method (Collocation and two Runge-Kutta methods) is stable with constant $C \cdot \kappa_k N$ (where C is real constant) and leads to a unique solution $y(x)$ of two points boundary values problem (3.2). Furthermore at mesh-points $\xi_j \in \overline{\Delta}$ it holds

$$\left| y^{(\nu)}(\xi_j) - \overline{u}_{\overline{\Delta}}^{(\nu)}(\xi_j) \right| = O(h^p), \qquad \nu = 0, 1; \ j = 0, 1, ..., N+2, \tag{3.38}$$

while, on the other hands , for $\forall x \in [\xi_j, \xi_{j+1}]$, $j = 0, 1 \ldots, N+1$ occurs

$$\left| y^{(\nu)}(x) - \overline{u}_{\Delta}^{(\nu)}(x) \right| = O(h_j^{k+2-\nu}) + O(h^p), \qquad \nu = 0, 1, \tag{3.39}$$

where $h_j = \left|\xi_{j+1} - \xi_j\right|$, $j = 0, 1, 2, ...N+1$, and $h = \max\limits_{0 \leq\ j \leq\ N+1} h_j$.

Proof. Using the results prove by *U. Ascher, R. M. Mattheij şi R. D. Russel* in ([5, **Theorem 5.140,** page: 253]) for linear boundary values problem (BVPL) (3.34), we obtained estimates (3.38) and (3.39). Errors obtained for the approximations $\alpha = y'_{\overline{\Delta}}(a)$ and $\beta = y'_{\overline{\Delta}}(b)$ are of order $O(h^p)$, ([23, **Theorem** 5.4.1, page: 293]). If we choose and embedded pair of Runge-Kutta method of order at least $(p, p+1)$, and conditions in hypothesis of theorem are fulfilled, then the final error have the order $O(h^p)$. The stability of this method results from the equivalence of collocation method with a *Runge-Kutta* method [5, **Theorem** 5.73, page: 219].

So if the mesh is sufficiently fine, the embedded pair of *Runge-Kutta* methods does not increase the order of error. ■

Remark 35 *The condition number may grow rapidly when h is small. In the paper [5, page:129]* U.Ascher *give the following estimation:*

$$\kappa_\Delta \approx K \sum_{i=0}^{N+1} h_i^{-2} \max_{i=0,\ldots,N+1} \int_{x_i}^{x_{i+1}} |G(x_i, t)|\, dt,$$

where K is a generic constant and G is the Green's *function for (BVPL) problem.*

Remark 36 *We suggest using the combined approach method (Runge-Kutta and B-spline of degree k) if the problems (PVPL) (3.1) have singularities on the endpoints of interval* $[0, 1]$ *(see the numerical example 3.4.1).*
Detection can be done using the step control algorithm ([1, pp: 376-380], [66]).

Conclusion 37 *The error estimation does not depend on the number of collocation points. Nevertheless, the* Runge-Kutta *methods requires an order greater or equal to the order of error for derivatives at a and b. We conclude: collocation method combined with* Runge-Kutta *method is an effective method to determinate an approximate solution of (PVPL) problem.*

3.4.1 Numerical example

We consider the two-points boundary values linear problem:

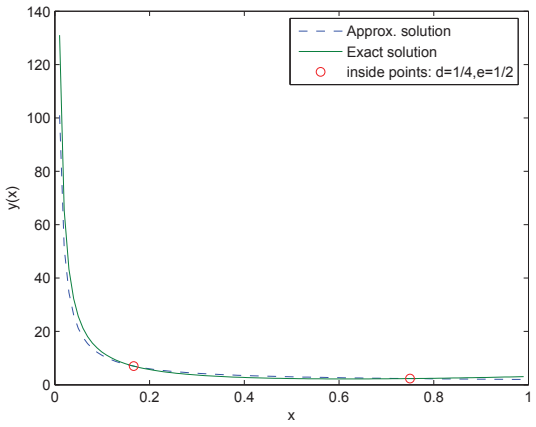
$$\begin{cases} y'' = \frac{2}{x^2}(y-1), & x \in (0,1) \\ y\left(\frac{1}{4}\right) = 5,\ y\left(\frac{1}{2}\right) = 3. \end{cases} \tag{3.40}$$

The above problem has the exact solution:

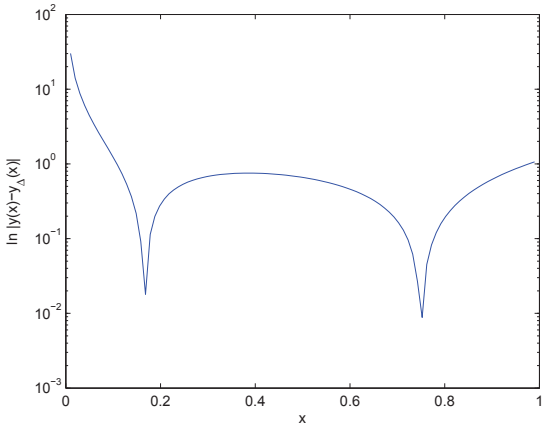
$$y(x) = 1 + 1/x. \tag{3.41}$$

Using the step control algorithm ([1, pp: 376-380], [66]), we set that the exact solution has a singularity in $x = 0$. Applying the method described in this paragraph for $N = 11$ and $k = 3$, we got the approximate solution and we represent it in Figure 3.6(a) and the errors in Figure 3.6(b). We wrote the programs in `MATLAB 10a` and run times given by functions `tic-toc`:

```
Elapsed time is 0.150000  seconds.
```



(a) Approximate solution



(b) Errors in logarithmic scale

Figure 3.6: Singularity in $x = 0$

Chapter 4

Nonlinear Boundary Value Problem

4.1 The statement of problem

Consider the second order nonlinear boundary value problem (PVPN) so

$$\begin{cases} y''(x) + f(x,y) = 0, & x \in [0,1] \\ y(a) = \alpha;\ y(b) = \beta, & a, b \in (0,1),\ a < b, \end{cases} \tag{4.1}$$

where $a, b,\ \alpha, \beta \in \mathbb{R}$ and $f : [0,1] \times \mathbb{R} \to \mathbb{R}$.

We try to solve the above problem using three methods:

1. a global collocation method based on B-splines of order $k+1$.
2. a combined method based on B-splines (order $k+1$) and a *Runge-Kutta* methods.
3. a pseudo-spectral collocation method (*C.C* method) with *Chebyshev* extreme points combined with a *Runge-Kutta* method.

Our choice to use these methods is based on the following reasons:

1. We write the code using the function `spcol in MATLAB Spline Toolbox ([39],[55],[58])`, and in `MATLAB dmsuite [68].`
2. It is the most suitable method, for a general purpose code, among the finite element ones. See ([7], [51], [52]), where complexity comparisons which support the above claim are made and collocation, when efficiently implemented, is shown to be competitive using extrapolation.
3. Theoretical results on the convergence of collocation method are given in ([33], [67]).
4. Several representative test problems demonstrate the stability and flexibility in [11].
5. For each *Newton* iteration, the resulting linear algebraic system of equations (after using *Newton* method with quasilinearization) is solved using methods given in [13].
6. There exists elegant theoretical results on the convergence of collocation method (see for example [31]).

We also consider the (BVPN) problem

$$\begin{cases} y''(x) + f(x,y) = 0, & x \in [a,b] \\ y(a) = \alpha;\ y(b) = \beta, & a,b,\alpha,\beta \in \mathbb{R} \end{cases} \tag{4.2}$$

To apply the collocation theory, we need to have an isolated solution $y(x)$ of the the problem (4.2), and this occurs if the above linearized problem for $y(x)$ is uniquely solvable. This is assuring by (**Theorem** 3) (see [50]).

4.2 Global collocation method based on B-splines function of order (k+1)

First we are interested to a global approach for the solution of the problem (PVPN) (4.1). We consider the grid $\overline{\Delta}$ of $[0,1]$ given by (3.3), and a number of such partitions $\overline{\Delta}_n$ $(n = 1,2,3,...)$ which satisfy the condition

$$\lim_{n\to\infty} h(\overline{\Delta}_n) = 0, \text{ where } h := \max\{h_i\},\ h_i := x_{i+1} - x_i.$$

Definition 38 *A function $v(x)$ is in family $L(\overline{\Delta}_n, k, 2)$ if $v(x)$ is a polynomial of degree k on each subinterval of $\overline{\Delta}_n$, $v(x) \in C^p[0,1]$ and the subfamily*

$$L'(\overline{\Delta}_n, k, 2) \subset L(\overline{\Delta}_n, k, 2),$$

consists of all functions that satisfied the boundary value conditions

$$y(a) = \alpha;\ y(b) = \beta,\ a,b \in (0,1),\ a < b.$$

The next lemma gives a sufficient condition for convergence.

Lemma 39 *([50], [67]) Let $\overline{\Delta}$ be a grid (3.3) and if we form a set of points $S_n (n = 1,2,3,...)$ like in (2.6) then for a large n, there is a unique element $u_{\overline{\Delta}_n}(x) \in L'(\overline{\Delta}_n, k, 2)$ satisfying (4.2) at each point of S_n and*

$$\left\| u^{(j)}_{\overline{\Delta}_n}(x) - y^{(j)} \right\| \le \delta,\ j = 0,1.$$

The approximate solution $u_{\overline{\Delta}_n}(x)$ and its derivatives up to the order two converge uniformly to exact solution $y(x)$. Moreover the rate of convergence is bounded by:

$$\left\| u^{(j)}_{\overline{\Delta}_n}(x) - y^{(j)} \right\| \le \theta F_n(y''),\ j = 0,1,$$

where θ is a constant independent of n and $F_n(y'')$ is the error of the best uniform approximation to $y''(x)$ in $L(\overline{\Delta}_n, k, 2)$.

For reasons of efficiency, stability, continuity, we choose as the basis B-splines of order $k+1$, where $k =$ degree of *Legendre* polynomial. We wish to find an approximate solution of (PVPN) problem (4.1) in $L(\overline{\Delta}_n, k, 2)$, having the following form:

$$u_{\overline{\Delta}_n}(x) = \sum_{i=0}^{N+2} c_i B_{i,k}(x), \tag{4.3}$$

where $B_{i,k}(x)$ is a B-spline of order $(k+1)$ with mesh-points :

$$\overline{\Delta} := \{\xi_i \ , i = -k+1, ..., N+k+1\},$$

and c_i are real constants to be determined.

Remark 40 *Our approximation method is inspired from [8, chap. 2,5].*

Let

$$J = \{0, 1, ..., N+2\} \backslash \{l, l+p\}.$$

We impose the conditions:

- **(c1)** The approximate solution (4.3) satisfies the differential equation (4.1) at ξ_j, $j \in J$, where ξ_j are the collocation points.
- **(c2)** The solution satisfies:

$$u_{\overline{\Delta}_n}(\xi_l) = \alpha, \ u_{\overline{\Delta}_n}(\xi_{l+p}) = \beta \ , \ a = \xi_l, \ b = \xi_{l+p}.$$

The conditions **(c1)** and **(c2)** yield a nonlinear system with $(N+3)$ equations

$$\begin{cases} \sum\limits_{i=0}^{N+2} c_i B_{i,k}(a) = \gamma_1, \\ \sum\limits_{i=0}^{N+2} c_i B_{i,k}(b) = \gamma_2, \\ \sum\limits_{i=1}^{N+1} c_i B_i''(\xi_j) + f(\xi_j, \sum\limits_{i=1}^{N+1} c_i B_i(\xi_j)) = 0, \end{cases} \tag{4.4}$$

with unknowns $\{c_i\}_{i=0}^{N+2}$. If $F = [F_0, F_1, ..., F_{N+2}]^T$ are the functions defined by equations of the nonlinear systems, using the quasilinearization of *Newton* [5, pp. 52-55], we find the next approximation by means

$$c^{(\upsilon+1)} = c^{(\upsilon)} - w^{(\upsilon)} \tag{4.5}$$

where $c^{(\upsilon)}$ is the vector of unknowns obtained at the $\upsilon-$th step and $w^{(\upsilon)}$ is the solution of the linear system:

$$F\,'(c^{(\upsilon)})w = F(c^{(\upsilon)}). \tag{4.6}$$

The Jacobian matrix $F\,' = (J_{ij})$ is banded and it is given by:

$$J_{ij} = \begin{cases} B_j(a), & \text{for } i = 0 \\ B_j(b), & \text{for } i = N+2 \\ B_j''(\xi_i) + \frac{\partial f}{\partial y}(\xi_i, \sum\limits_{i=1}^{n+1} c_i B_j(\xi_i)) y(\xi_i), & \text{for } i = 1, 2, 3, ...N+1 \end{cases} . \tag{4.7}$$

To solve (4.1) we used the method presented in [18]; an initial approximation $u^0 \in C^1[0,1]$ is required. The successful **stopping criterion** [6] is

$$\left\| u^{(k+1)} - u^{(k)} \right\| \leq abstol + \left\| u^{(k+1)} \right\| reltol,$$

where *abstol* and *reltol* is the absolute and the relative error tolerance, respectively and the norm is the usual uniform convergence norm. The reliability of error-estimation procedure being used for stopping criterion was verified in [3]. Papers on this topics exploit the almost block diagonal structure of collocation matrix and recommend LU factorization ([3], [13]).

4.3 A combined method using B-splines and Runge-Kutta methods

The nonlinear boundary value problem (PVPN) can be solved by dividing it in three problems like:

1. A two-points nonlinear boundary values (BVPN) (4.2) on $[a,b]$,
2. Two nonlinear problems *Cauchy* (IVPN-Left) on $[0,a]$

$$\begin{cases} y'(x) = z(x) \\ z'(x) = -f(x,y) \\ y(a) = \alpha;\ y'(a) = \gamma,\ \gamma \in \mathbb{R} \end{cases} \tag{4.8}$$

3. and (IVPN-Right) on $[b,1]$

$$\begin{cases} y'(x) = z(x) \\ z'(x) = -f(x,y) \\ y(b) = \beta;\ y'(b) = \delta,\ \delta \in \mathbb{R} \end{cases} \tag{4.9}$$

or treating it globally.

Assuming that the two-points nonlinear boundary values problem (BVPN) given by (4.2) has the unique solution, the requirement $y \in C^1[0,1]$ ensure the existence and the uniqueness of the exact solution of (4.1). For the existence and the uniqueness of the exact solutions of the problems (IVPN-Left) and (IVPN-Right) we use **Theorem** 4 [30, **Theorem 3.1,** page: 113].

We use the mesh $\overline{\Delta}$ of the interval $[0,1]$ given by (3.3). To solve numerically the two-points nonlinear problem (BVPN), we use global collocation method with B-spline functions of degree k presented in the previous paragraph, and for the two problems (IVPN-Left) and (IVPN-Right) we use *Runge-Kutta* methods of order k.

Since $f(x,y) \in C^2\{[0,1]\times\mathbb{R}\}$ and it must ensure continuity and differentiability in $x=a$ and $x=b$, we choose the order of multiplicity k for knots, $\xi_j \in \overline{\Delta},\ j \neq l, l+p$, and for inner points $\xi_l = a$ şi $\xi_{l+p} = b$ the order of multiplicity $k+2$. The values $y\,'(a)$ şi $y'(b)$ can be computed by differentiating the approximate solution of the two-points boundary values problem (BVPN) (4.2) in points $x=a$ and $x=b$, and let $u_{\overline{\Delta}}(x)$ be the approximation computed by the combined method.

Theorem 41 *[47, section 3] If u is the isolated solution of (PVPN) problem, f has continuous second order partial derivative and its accuracy is $\mathbb{O}\ (h^{k+1})$, where h is the norm of the partition $\overline{\Delta}$.*

Proof. For the (PVPN) problem (4.1) we apply **Theorem** 5.147, page: 257 in [5].

We conclude that *Newton* method, applied to Δ, converges quadratically to the restriction of u to Δ, and the accuracy for the approximation and its derivative is $\mathbb{O}(h^{k+1})$, that is:

$$\left| u_{\overline{\Delta}}^{(j)}(x) - y^{(j)}(x) \right| = \mathbb{O}(h^{k+1}), \quad x \in [a,b],\ j = 0,1.$$

We extend convergence and the accuracy to the whole interval $[0,1]$ by using the stability and the convergence of *Runge-Kutta* methods. A $(k+1)$-order explicit *Runge-Kutta* methods is consistent and stable, so it is convergent, and its accuracy is $\mathbb{O}(h^{k+1})$. Thus the final solution has the same accuracy. The stability and convergence of *Runge-Kutta* method are guaranteed by **Theorem 5.3.1, page 285** and **Theorem 5.3.2, page 288** in [23]. ■

4.4 Some considerations on complexity

We will give a rough estimation of the complexity of above methods. We start with the first method. In the sequel, B will be the cost for B-spline evaluation and f the time for a function evaluation. The time required to construct the collocation matrix is

$$C_0 = 2(Nk+1)(k+1)B.$$

To construct the Jacobian we need $Nk(k+1)(B+f)$. The construction of the right-hand side requires

$$(Nk+2)B + NkB + Nkf.$$

So, for the linear system construction we obtain:

$$W_1 = ((B+f)k^2 + (4B+3f)kN + B.$$

For a banded linear system with bandwidth w the total cost for solution, using LU, with pivoting is

$$n\left(\frac{w^2}{2} + w\right),$$

(see [20, pp. 79-82]). In our case, $n = Nk+2$, and $w = \frac{3}{2}(k+2)$, and the cost for the solution of linear system will be:

$$W_2 = \left(\frac{21}{2}k + \frac{9}{8}k^3 + \frac{15}{2}k^2\right)N + 15k + 21 + \frac{9k^2}{4}.$$

The cost of Newton step is $W_s = W_1 + W_2$, that is

$$W_s = \left[\frac{9k^3}{8} + \left(f + B + \frac{15}{2}\right)k^2 + \left(3f + 4B + \frac{21}{2}\right)k\right]N + 2B + 15k + 21 + \frac{9k^2}{4}.$$

The total cost is $IW_s + C_0$, where I is the number of steps required in *Newton* methods. Since the convergence is quadratic, if the final tolerance is ε, assuming $\delta_{i+1} = c\delta_i$, is the error at $i-th$ step, we obtain [38, pp 295-297]:

$$I = \frac{1}{\log 2}\log\frac{\log|c| + \log\varepsilon}{\log|c| + \log|\delta_0|}.$$

For the second method, the same analysis works for (BVPN) solution part. We have an additional amount of work for *Runge-Kutta* method. If the number of stages is s and the number of points is p, the cost is $\mathbb{O}(psf)$.

4.5 A combined method with pseudo-spectral and *Runge-Kutta* methods

Consider the grid

$$0 = x_{-q} < ... < x_{-1} < a = x_0 < x_1 < ... < x_N = b < x_{N+1} < ... < x_{N+p} = 1 \tag{4.10}$$

We choose *Gauss-Lobatto-Chebyshev* type mesh:

on $[a,b]$

$$x_i = \frac{(b-a)\cos\frac{\pi i}{N} + a + b}{2}, i = 0,1,...N, \tag{4.11}$$

on $[0,a]$

$$x_i = \frac{a}{2}(\ -\cos\frac{\pi i}{N} + 1), i = -1,-2,... - q, \tag{4.12}$$

and on $[b,1]$

$$x_i = \frac{(b-1)\cos\frac{\pi i}{N} + 1 + b}{2}, i = N+1, N+2, ..., N+p. \tag{4.13}$$

To approximate the solution of two points boundary nonlinear problem (PVPN) follows the ideas in [17]. Considering the *Lagrange* basis (l_k) we have

$$y(x) = \sum_{k=0}^{N} \ell_k(x) y(x_k) + (R_N y)(x), \qquad x \in [a,b], \tag{4.14}$$

where

$$(R_N y)(x) = \frac{y^{(N+1)}(\xi)}{(N+1)!}(x-x_0)\cdots(x-x_N),$$
$$\min\{x_0, x_1, ..., x_N\} < \xi < \max\{x_0, x_1, ..., x_N\},$$

is the remainder of *Lagrange* interpolation.

Since $y(x)$ fulfills the differential equation given by (4.2), we obtain:

$$\sum_{k=0}^{N} \ell_k''(x_i) y(x_k) + (R_N y)''(x_i) = -f(x_i, y(x_i)), \qquad i = 1, \ldots, N-1. \tag{4.15}$$

Setting $y(x_k) := y_k$ and ignoring the rest, one obtains the nonlinear system:

$$\sum_{k=0}^{N} \ell_k''(x_i) y_k = -f(x_i, y(x_i)), \qquad i = 1, \ldots, N-1, \tag{4.16}$$

with unknowns y_k, $k = 1,2,...,N-1$, here $y_0 := y(x_0) = \alpha$,and $y_N := y(x_N) = \beta$. The approximate solution of (BVPN) problem given by (4.2) is in fact the *Lagrange* interpolation polynomial at nodes $\{x_k\}, k = -q, ..., 0, 1, ...N, ..., N+p$. The nonlinear system (4.16) can be rewritten as:

$$AY_N = F(Y_N) + b_N$$

where:

$$A = [a_{ik}], \quad a_{ik} = \ell_k''(x_i), \quad k, i = 0, 1, \ldots, N-1, N, \tag{4.17}$$

$$F(Y_N) = \begin{bmatrix} -f(x_1, y_1) \\ \vdots \\ -f(x_{N-1}, y_{N-1}) \end{bmatrix}, \qquad b_N = \begin{bmatrix} -\alpha\ell_0''(x_1) - \beta\ell_N''(x_1) \\ \vdots \\ -\alpha\ell_0''(x_{N-1}) - \beta\ell_N''(x_{N-1}) \end{bmatrix}. \tag{4.18}$$

Since:

$$\ell_k(x_i) = \ell_{N-k}(x_{N-i}); \ell_k''(x_i) = \ell_{N-k}''(x_{N-i}), \quad k, i = 0, 1, \ldots, N-1, N,$$

the matrix A is centro-symmetric (see [16] for proof), so nonsingular.

We introduce

$$G(Y) = A^{-1}F(Y) + A^{-1}b_N, \tag{4.19}$$

where the solution Y will be fixed point of G. To solve numerically the problem(4.1) on mesh given by (4.11) we use *C.C* method on $[a, b]$ and on $[0, a]$, $[b, 1]$ *Runge-Kutta* methods, where on $[0, a]$ we use the knots given by(4.12) and on $[b, 1]$ the knots given by (4.13).The derivative $y'(a)$ şi $y'(b)$ can be computed using the relations (3.30) and (3.32).

In [17, **Theorems: 2.1, 2.2**] *F.A. Costabile* and *A.Napoli* prove the following theorems.

Theorem 42 *If f verifies a* Lipschitz *condition with respect to second variable:*

$$|F(x, y_1) - F(x, y_2)| \leq L\,|y_1 - y_2|$$

and $\left\|A^{-1}\right\| L < 1$, then the system (4.16) has a unique solution which can be calculated by successive approximation method

$$Y^{(n+1)} = G(Y^{(n)}), n \in N^*, \tag{4.20}$$

where $Y^{(0)}$ fixed and G given by (4.19), and the matrix A is (4.17).

Theorem 43 *If $Y = [y(x_1), y(x_2), ..., y(x_{N-1})]^T$, where $y(x) \in C^1[0, 1]$ is the exact solution of (BVPN) problem and:*

$$Y_N = [y_1, y_2, ..., y_{N-1}]$$

where y_i are the values of approximated solution at $x_i \in \Delta_1$ computed by (4.16) and

$$R = [-(R_N y)''(x_1), ..., -(R_N y)''(x_{N-1})]^T,$$

then for error: $\|Y - Y_N\|$ it holds:

$$\|Y - Y_N\| \leq \frac{\left\|A^{-1}\right\| \|R\|}{1 - \|A^{-1}\| L}. \tag{4.21}$$

Combining the results obtained by *F.A. Costabile, A.Napoli* [17, **Theorems: 2.1, 2.2**] with the stability and convergence of *Runge-Kutta* methods [23, **Theorem** 5.3.1 page 285, **Theorem** 5.3.2 page 288] we obtain

Theorem 44 ***(Daniel N.Pop, Radu T.Trâmbiţaş , I.Păvăloiu [46])*** *If*

$$\frac{\left\|A^{-1}\right\| \|R\|}{1 - \|A^{-1}\| L} \leq \mathbb{O}(h^k) \text{ and}$$
$$\left|y_N(a) - y'(a)\right| = \mathbb{O}(h^k),$$
$$\left|y_N(b) - y'(b)\right| = \mathbb{O}(h^k),$$

then for each point $x_i \in \Delta_1$, $i = -q, ..., N + p$ occurs

$$|y(x_i) - y_i| = \mathbb{O}(h^k).$$

Proof. The condition $\left\|A^{-1}\right\| L < 1$, assures us that G is a contraction. From *Banach's* fix-point theorem it follows that $(Y^{(n)})$ given by (4.20) is convergent to the exact solution $\overline{Y}$ of the system (4.16) and the following estimation holds:

$$\left\|\overline{Y} - Y^{(n)}\right\| \leq \frac{(\left\|A^{-1}\right\| \|L\|)^n}{1 - \|A^{-1}\| L} \left\|Y^{(1)} - Y^{(0)}\right\|.$$

If accuracy of the collocation method for the problem (4.2) is $\mathbb{O}(h^k)$ (i.e the approximate solutions y and its derivative $y'(a), y'(b)$ are within this accuracy limit), and if *Runge-Kutta* method is stable and has the order k, then the final solution has same accuracy. The stability and convergence of *Runge-Kutta* method are guaranteed by [23, **Theorems 5.3.1 page 285 and 5.3.2 page 288**]. ■

4.6 Numerical examples

In the following we make a comparative computational study by examples of the three numerical methods:

1. global method with B-spline functions,
2. combined collocation method with B-splines and *Runge-Kutta* methods,
3. *C.C* method combined with *Runge-Kutta* methods.

Using the idea given in ([8], [10], [19]) we write the code using `MATLAB Spline Toolbox and sparse matrices`, since the function `spcol` allows us to easily calculate the inverse of collocation matrix. We write and implemented two functions in `MATLAB` :

1. `polycollocnelin`, for global collocation method with B-spline functions,
2. `polycalnlinRK`, for combine method (B-spline collocation and *Runge-Kutta*).

The case where we know the exact solution

We gave in the paper [47], two numerical examples with know exact solution:

1. *Goldner'* ***problem*** [26]

$$\begin{array}{l} y''(x) + y^3(x) + \dfrac{4 - (x - x^2)^3}{(x+1)^3} = 0;\ x \in (0,1) \\ y(1/4) = 3/20;\ y(1/2) = 1/6 \end{array}, \tag{4.22}$$

   with the exact solution

$$y(x) = \frac{x - x^2}{x + 1}.$$

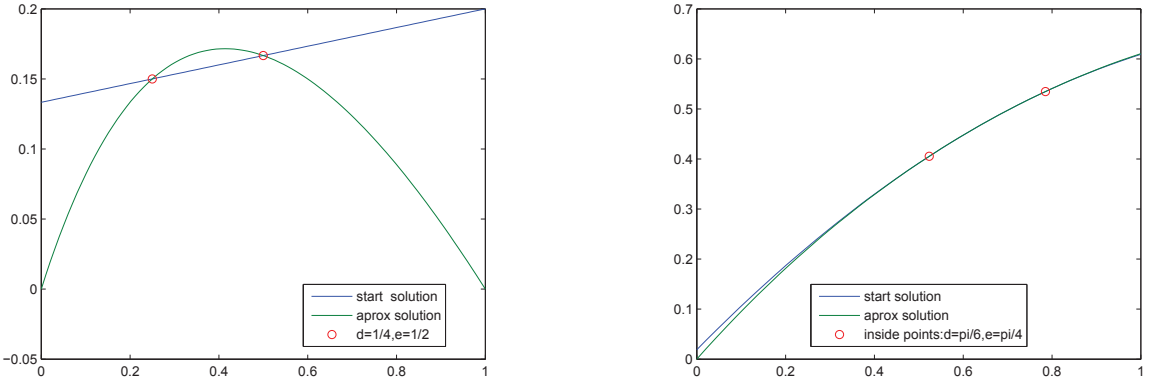
2. *Costabile'* ***problem*** [15]

$$\begin{array}{l} y''(x) + e^{-y(x)} = 0;\ x \in [0,1] \\ y(\pi/6) = \ln(3/2),\ y(\pi/4) = \ln((2+\sqrt{2})/2) \end{array}, \tag{4.23}$$

   with the exact solution

$$y = \ln(\sin(x) + 1).$$

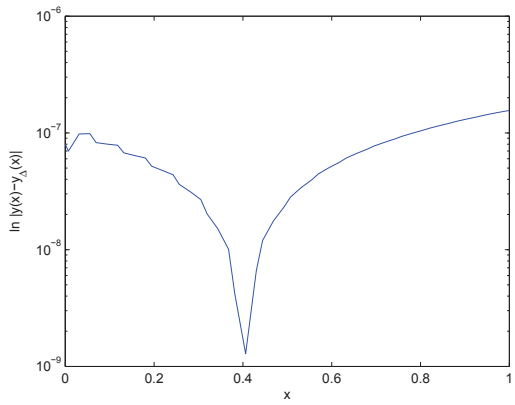
For each example we applied methods: global with *B-spline* of degree k and combined *B-spline* with *Runge-Kutta*. Figure 4.1 present approximate solution of the two-points boundary value problems (4.22) and (4.23). Errors for the two methods are represented in Figure 4.2 for the first example and Figure 4.3 for the second example. We have chosen as initial approximation *Lagrange* interpolation polynomial with the values α and β at a, $b \in (0, 1)$, computed using the `MATLAB` function `polyfit`.



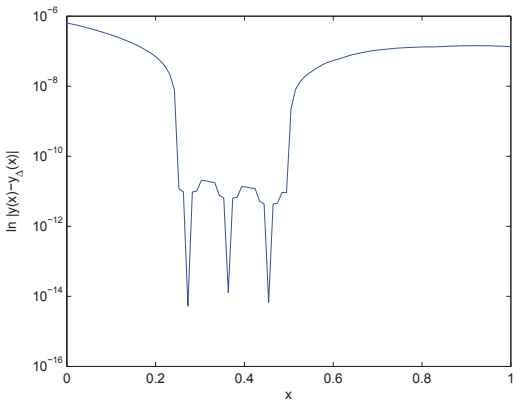
(a) Goldner-problem (4.22)

(b) Costabile-problem (4.23)

Figure 4.1: Approximate solution and initial start value in Goldner and Costabile problems

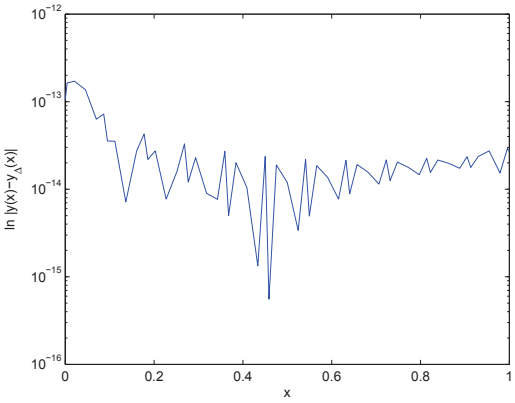


(a) Global collocation method with B-spline functions

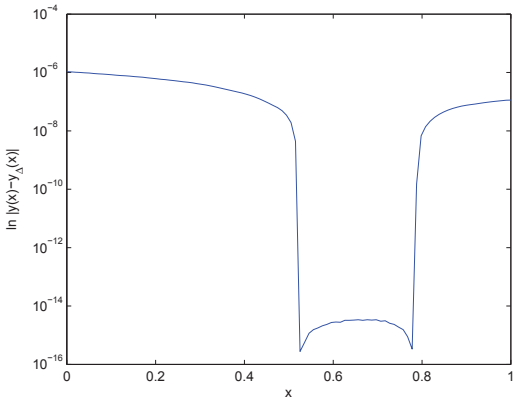


(b) Combined collocation method B-splines with Runge-Kutta

Figure 4.2: Errors in Goldner-problem



(a) Global collocation method with B-spline



(b) Combined method B-spline and Runge-Kutta

Figure 4.3: Errors in Costabile problem

Table 4.1 gives the residuals

$$e_{\overline{\Delta}}^{(j)} \left\| y^{(j)} - u_{\overline{\Delta}}^{(j)} \right\|, \text{ for } j = 0, 1, 2,$$

for the global method based on B-splines. For the residuals it holds

$$\left\| y^{(j)} - u_{\overline{\Delta}}^{(j)} \right\| = \mathbb{O}(\|\Delta\|)^{k+2-j}, \text{ for } j = 0, 1, 2$$

In order to compare the costs (run times) experimentally we used `MATLAB` functions `tic` and `toc`.The results are given in Table 4.2. The time for combined method is a bit larger.

N	$\left\|y - u_{\overline{\Delta}}\right\|$	$\left\|y' - u'_{\overline{\Delta}}\right\|$	$\left\|y^{"} - u^{"}_{\overline{\Delta}}\right\|$	$\left\|y - u_{\overline{\Delta}}\right\|$	$\left\|y' - u'_{\overline{\Delta}}\right\|$	$\left\|y^{"} - u^{"}_{\overline{\Delta}}\right\|$
5	$6e-05$	0.000123	0.002290	$1.04e-05$	$2.15e-05$	0.000172
6	$6e-05$	0.000123	0.002290	$3.72e-06$	$8.47e-06$	0.000124
7	$6e-05$	0.000123	0.002290	$1.59e-06$	$3.96e-06$	0.000106
8	$6e-05$	0.000123	0.002290	$7.37e-07$	$2.02e-06$	$5.65e-05$
9	$6.9e-06$	$1.63e-05$	0.000769	$3.84e-07$	$1.16e-06$	$3.61e-05$
10	$6.9e-06$	$1.63e-05$	0.000769	$2.04e-07$	$6.81e-07$	$2.42e-05$
11	$6.9e-06$	$1.63e-05$	0.000769	$1.21e-07$	$4.46e-07$	$2.14e-05$
12	$6.9e-06$	$1.63e-05$	0.000769	$1.82e-08$	$2.02e-07$	$1.81e-05$
13	$1.4e-06$	$3.64e-06$	0.000346	$1.30e-08$	$1.44e-07$	$9.77e-06$
14	$1.4e-06$	$3.64e-06$	0.000346	$7.54e-09$	$1.09e-07$	$1.51e-05$
15	$1.4e-06$	$3.64e-06$	0.000346	$5.54e-09$	$8.32e-08$	$6.08e-06$
16	$1.4e-06$	$3.64e-06$	0.000346	$3.54e-09$	$6.22e-08$	$7.92e-06$
17	$3.99e-07$	$1.22e-06$	0.000148	$2.81e-09$	$4.65e-08$	$4.65e-06$
18	$3.99e-07$	$1.22e-06$	0.000148	$1.89e-08$	$3.51e-08$	$4.76e-06$
19	$3.99e-07$	$1.22e-06$	0.000148	$1.43e-09$	$2.98e-08$	$4.14e-06$
20	$3.99e-07$	$1.22e-06$	0.000148	$1.02e-09$	$2.50e-08$	$3.70e-06$

Table 4.1: Error table for example (1) left, and example (2) right

	Method 1	Method 2
First example	0.0117501	0.022751
Second example	0.0163870	0.021535

Table 4.2: Run times

Case of unknown exact solution

To implement C.C method combined with Runge-Kutta methods we used the MATLAB 10.0 functions:

`cebdif,cebint,cebdiff,`

from the library `dmsuite` described in ([68], [59]), and for *Cauchy* problems on the intervals $[0,a]$, $[b,1]$ we use `ode45.` We chose $\{x_k, k=0,1,2,...,n\}$ as extreme *Chebyshev* points on the interval $[0,1]$. Since the successive approximation method is slow, we solve the nonlinear system by *Newton' method.* For this purpose we wrote the function `solvepolylocalceb` and then we called solver `ode45` for *Runge Kutta* methods, and the derivatives $y'(a), y'(b)$ are computed by the function `cebdifft.`

- The first example is *Bratu'* problem (2.37) for $\lambda = 1$.

$$\begin{cases} y''(x) + e^{y(x)} = 0, \ 0 < x < 1 \\ y(0.2) = 0.08918993462883 \\ y(0.8) = 0.08918993462883 \end{cases} \tag{4.24}$$

We took $N = 128$. We solve the two-points boundary values nonlinear problem (4.24) using three methods:

1. *C.C* method combined with *Runge-Kutta* methods, where we chose initial start value

$$y^{(0)}(x) = x(1-x), \tag{4.25}$$

2. global collocation method with B-splines, where we chose initial start value

$$y^{(0)}(x) = 3.9x(1-x)/7,$$

3. combined method B-splines with *Runge-Kutta* methods ,where we chose initial start value

$$y^{(0)}(x) = x(1-x).$$

We used to compare the cost, the MATLAB functions `tic` and `toc` and we obtained the following results

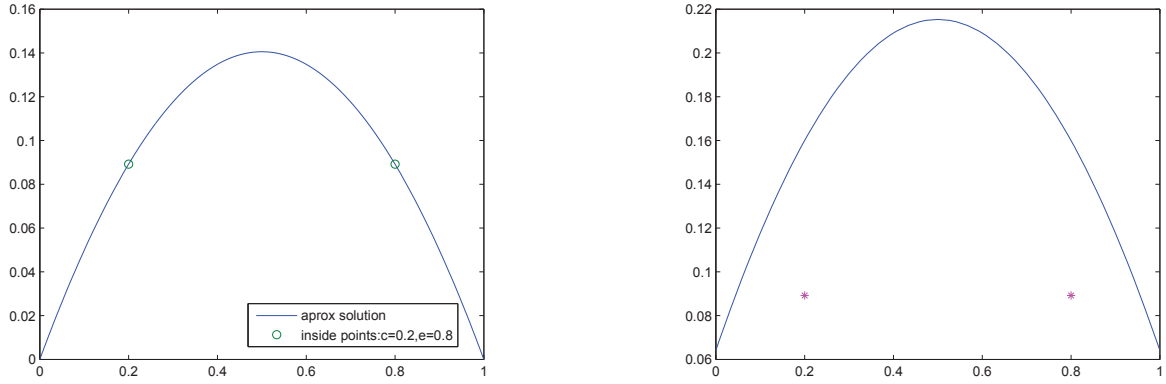
ε	*C.C* and *R.K*	*B.S* and *R.K*	*Global B.S*
10^{-5}	0.0540	0.035	0.021
10^{-6}	0.077	0.043	0.023
10^{-7}	0.049	0.025	0.024
10^{-8}	0.055	0.031	0.031
10^{-9}	0.054	0.036	0.030
10^{-10}	0.058	0.028	0.026

,

where ε is the tolerance in stopping criterion.

The graph of approximate nonlinear solution of the problem (4.24) is obtained after 4 iterations, and we plotted it in Figure 4.4(a).

Remark 45 *For the global method with B-splines we did not choose initial value (4.25), because that method is not convergent, fact visible in the graph given in Figure 4.4(b).*



(a) Convergence (b) Divergence

Figure 4.4: Bratu-problem for $\lambda = 1$

- The second example is a variant of the problem in the average temperature of reaction-diffusion problem (2.39) for $p = 3$.

$$\begin{cases} u'' + u^3 = 0;\ 0 < x < 1 \\ u(0.2) = 1.929990320692795 \\ u(0.8) = 1.929990320692795 \end{cases} \tag{4.26}$$

In this numerical example we took $N = 128$, the tolerance $\epsilon = 10^{-8}$ and for the initial solution we chose

$$u^{(0)} = \frac{12\pi}{\sqrt[2]{2}} x(1-x), \tag{4.27}$$

for C.C method combined with *Runge-Kutta*, *B-spline* with *Runge-Kutta* and for the initial start value we built a spline that passes through the points $[0;\ 0.2;\ 0.5;\ 0.8; 1]$. We have determined the values $u(0.2),\ u(0.5)$ and $u(0.8)$ writing code in `Maple13.0`.

After 8 iterations, we obtained the approximate nonlinear solution of the problem (4.26) in Figure 4.5(a).

We solved the two-points boundary nonlinear problem (4.26) using three methods:

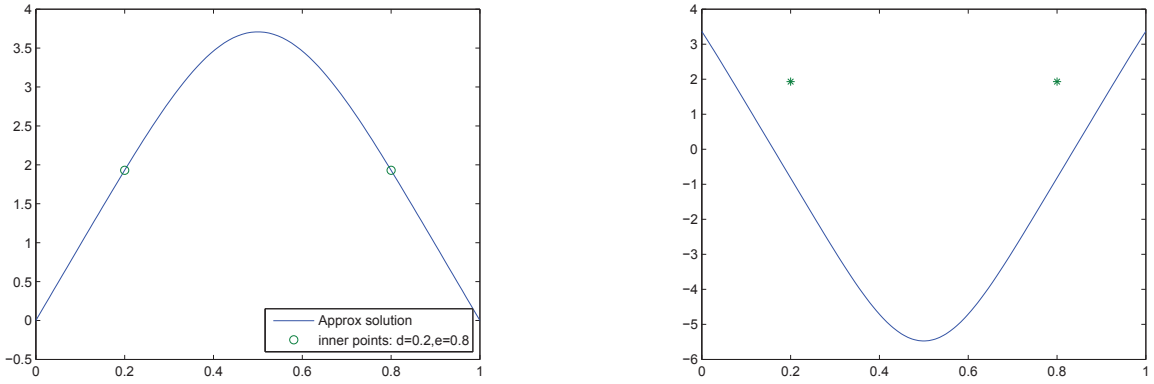
1. combined method C.C with Runge-Kutta,
2. global collocation method with B-splines,
3. combined method B-splines with Runge-Kutta.

Using again the functions `MATLAB tic` and `toc` to compare the cost (run-times), we obtained the following results

ε	$C.C + R.K.$	$B.S + R.K.$	$Global - B.S.$
$1.e-3$	0.221242	0.040019	0.034571
$1.e-4$	0.230269	0.046225	0.022515
$1.e-5$	0.238405	0.054828	0.032165

,

where ε is again the tolerance in stop criterion.



(a) Convergence (b) Divergence

Figure 4.5: The average temperature in reaction-diffusion process $p = 3$

Conclusion 46 *In both examples studied global B-splines method is faster.*

Remark 47 *Cost and number of iterations depend on the number of division points N. The tolerance used in stop criterion ϵ and the initial value $u^{(0)}$ is essential because it ensures convergence method. For example, a choice of initial solution (4.27) for global method with B-splines does not ensure convergence, and this is visible in the graph of approximate nonlinear solution of the problem (4.26) in Figure 4.5(b).*

Case when we have singularities in end points of the interval

Global collocation method with B-splines can not be applied for the solution of nonlinear problems which has singularities at the end points of the interval $[0, 1]$. In this case we must choose the combined B-spline with Runge-Kutta methods. Since near the end points we have singularities we chose the solver `ODE23tb` for Runge-Kutta methods.We exemplified this by *Fermi-Thomas* problem (2.40)with boundary conditions (2.41).

It should be noted that this problem has been studied by *Shampine, Gladwell and Thomson* [57, pp 155-156] using the solver `bvp4c`. We use for start solution

$$y(x) = 144x^{-3},$$

and for inner points $d = 0.015$, and $e = 59$.

Method	Run-Times
Shampine-Gladwell-Thomson	0.838735 seconds.
B-spline and Runge-Kutta	0.493448 seconds.

The graph of approximate nonlinear solution of *Fermi-Thomas* problem is presented in Figure 4.6.

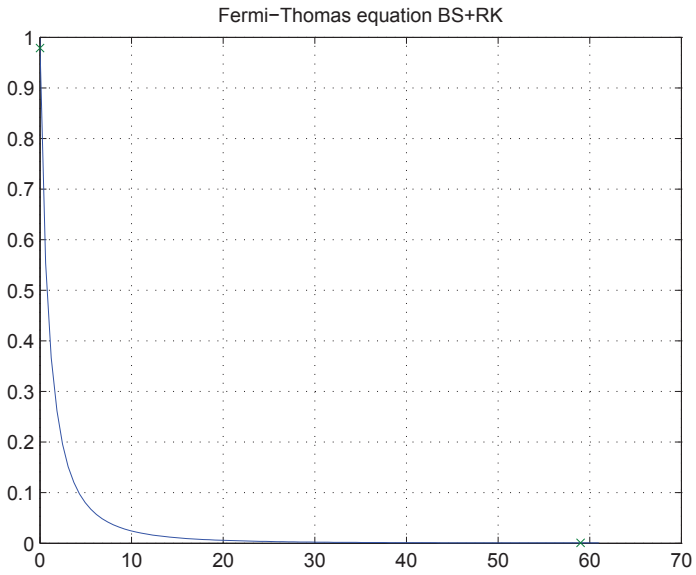


Figure 4.6: The charge density in atoms of high atomic number

Remark 48 *The running time for B-spline collocation is the shortest, since its collocation matrix is banded.*

Chapter 5

MATLAB and MAPLE codes

The first ODE solver of MATLAB was based on a FORTRAN programs written by *Carl de Boor* [8], *Lary Shampine* and *H.A. (Buddy) Watts* [56]. Also *Carl de Boor* write a library in MATLAB named Spline Toolbox([9], [10]). In this library there is a function *spcol,* which provide us a simple way to calculate the inverse of B-splines collocation matrix. The theoretical results on the convergence of collocation method ([11], [12]), together with those on error estimation and mesh selection [53], are more general than for the other methods. This, and the basic simplicity of the collocation procedure [19], also make programming of the method reasonably straightforward. *U. Ascher, S. Pruess* and *R.D. Russell* ([3], [4], [6]), shows that for low continuity piecewise polynomials, some of B-splines and their derivatives are largely mesh independent and hence cheap to evaluate. Unfortunately, the mesh matrices have condition numbers which grow very rapidly as the mesh is refined. The ODE Suite has evolved considerably as a result of further work by *L. Shampine, J. Kierzenka* and *M.W.Reichelt* [58]. *L.N. Trefeten* introduces in his monograph [64], some fairly useful MATLAB codes which implements spectral method. *Weideman* and *Reddy* present in their paper [68], a software suite which consists of 17 MATLAB functions for solving differential problems by the spectral collocation methods. These functions enable us to generate spectral differential matrices based on *Chebyshev* interpolates, and we use these functions in implementation of our *C.C* methods.

MAPLE is a general-purpose commercial computer algebra system.It was first developed in 1980 by the Symbolic Computation Group at the University of Waterloo in Waterloo, Ontario, Canada. Since 1988, it has been developed and sold commercially by Waterloo Maple Inc. (also known as Maplesoft), a Canadian company also based in Waterloo, Ontario. The current major version is version 15 which was released in April 2011. Users can enter mathematics in traditional mathematical notation. There is extensive support for numeric computations, to arbitrary precision, as well as symbolic computation and visualization.Maple is based on a small kernel, written in C, which provides the Maple language. Most functionality is provided by libraries, which come from a variety of sources. Many numerical computations are performed by the NAG Numerical Libraries, ATLAS libraries, or GMP libraries. Different functionality in Maple requires numerical data in different formats. Symbolic expressions are stored in memory as directed acyclic graphs. The standard interface and calculator interface are written in *Java*. The language permits variables of lexical scope.

However, in this **Section** we introduce some MATLAB and MAPLE codes which implement **Global B-splines methods, Combined methods based on B-splines and Runge-Kutta, and C.C methods combined with Runge-Kutta methods.**

Remark 49 *The following programs solve also the classical problems(BVP), if $a = d$ and $e = b$ but might apply to delay differential equations (DDE).*

5.1 Linear Case

5.1.1 MATLAB Codes

=======

```
%Linear case
%Standard equation :y"+q(x)y+r(x)=0, a<x<b
%Inner conditions: y(d)=alpha, y(d)=beta, a<d<e<b
%number of collocation points N=nk+2, k=degree of B-splines

q = @(x) -ones(size(x));
r = @(x) x;
%exact solution
solex= @(x) -x+sin(x)/sin(1);
a=0; b=1;
d=1/4;
e=1/2;
alpha=solex(d);
beta=solex(e);
t=linspace(a,b,100);
n=36;
k=3;
tic;
[x,y]=BVPcollocRK(q,r,a,b,d,e,alpha,beta,n,k,t);
toc;
figure(1)
plot(t,solex(t),x,y,[d,e],[alpha,beta],'o')
title('PVP problems .... ')
legend('Approximate solution','d=1/4,e=1/2',0)
grid on
figure(2)
semilogy(x,abs(solex(x)-y),'-')
title('Errors in semilogarithmic scale ')
grid on
```

=======

```
function [x,y]=BVPcollocRK(q,r,a,b,d,e,alpha,beta,N,k,t)
%BVPCOLLOCRK - collocation for linear polylocal problem
%-y"(x)+q(x)*y(x)=r(x);
%call [sp]=BVPcolloc(q,r,a,b,alpha,beta,N,k)
%q,r - functions, define BVP
%a,b - endpoints
%d,e - inner points
```

```
%alpha, beta - values at endpoints
%N - number of subintervals
%k -degree of B-splines
%t - evaluation points
%y - solution values at t
%ode23tb for stiff problem and ode45 for nonstiff

%prepare meshes
ti=t(t>=d & t<=e);
tleft=t(t<d);
tright=t(t>e);
%solve BVP
sp=BVPcolloc(q,r,d,e,alpha,beta,N,k);
yi=fnval(sp,ti);
ydi=fnval(fnder(sp),[d,e]);
x=ti';
y=yi';
%solver options
opts=odeset('AbsTol',1e-8,'Reltol',1e-7);
%solve right IVP
if ~is empty(tright)
    tright=[e,sort(tright)];
    [tr,wr]=ode23tb(@rhs,tright,[beta,ydi(2)],opts);
    x=[x;tr(2:end)];
    y=[y;wr(2:end,1)];
end
%solve left IVP
if ~is empty(tleft)
    tleft=[d,sort(tleft,'descend')];
    [tl,wl]=ode23tb(@rhs,tleft,[alpha,ydi(1)],opts);
    x=[tl(end:-1:2);x];
    y=[wl(end:-1:2,1);y];
end

% nested function for RK method
    function dy=rhs(x,u)
        dy=[u(2); q(x)*u(1)-r(x)];
    end
end
```

5.1.2 MAPLE codes

=======

B-splines method

```
> restart; with(CurveFitting): with(orthopoly):
Here we solve a problem (using B-splines functions)
```

```
In this situation the error in approximating the exact solution
u(x)=sin(p*x) is of order O(h^p).
We study the case p=9,n =10.
> Digits:=30:
> genpoints:=proc(a,b,N,k)
> local L,i,j,xu,xc,pol,pol2,sol,h,nL:
> 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): nL:=nops(L):
> return L:
> end proc:
> S:=(x,u,k)->eval(BSpline(4,t,knots=[seq(u[i],i=k-2..k+2)]),t=x):
> genspline:=proc(x,n,q,r,e,d,alpha,beta)
> #x - mesh
> #n - number of points
> #q, r - functions of differential equation
> #e,d - inner points
> local k, ecY,ecd, Y,a0, b0, b,B,u:
>   global S, bcopy:
> Y:=0:
> for k from -1 to n+2 do
>      Y:=Y+b[k]*S(t,x,k):
> end do:
> Y:=simplify(Y):
> ecY:=diff(Y,t$2)-q(t)*Y=r(t):
> ecd:=Array(-1..n+2):
> for k from 1 to n+1 do
>      ecd[k]:=eval(ecY,t=x[k]):simplify(ecd[k]):
> end do:
> ecd[-1]:=eval(Y,t=e)=alpha:simplify(ecd[-1]):
> ecd[n+2]:=eval(Y,t=d)=beta:simplify(ecd[n+2]):
> B:=solve({seq(ecd[k],k=-1..n+2)},[seq(b[k],k=-1..n+2)]):
> print(nops({seq(ecd[k],k=-1..n+2)}));
> print(nops([seq(b[k],k=-1..n+2)])):
> assign(B):
> for u from -1 to n+2 do bcopy[u]:=b[u]:
```

```
> end do:
> return Y:
> end proc:
> q:=t->-86: r:=t->5*sin(9*t):
> ecz:=diff(Z(t),t$2)-q(t)*Z(t)=r(t):
> dsolve({ecz,Z(0)=0,Z(1)=sin(9)},Z(t)):simplify(%):
> assign(%):
> gendiv:=proc(a,b,e,d,alpha,beta,n)
> local h,x,Y:
> h:=(b-a)/(n+1):
> x:=Array(-2..n+3,[seq(a+i*h,i=-2..n+3)]):
> Y:=genspline(x,n,q,r,e,d,alpha,beta):return Y:
> end proc:
> 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; print(nn):
> x:=Array(-k..nn+k-1,L):
> return x;
> end proc:
> b:=1: a:=0: e:=Pi/54: d:=Pi/12:
> alpha:=simplify(eval(Z(t),t=e)):beta:=simplify(eval(Z(t),t=d)):
> n:=10:alpha:beta:
> plot(Z(t),t=0..1,color=[BLUE],title="Exact solution"):
> k:=3:
> x:=gendivLeg(a,b,n,k):
> nn:=ArrayNumElems(x)-2*k-3:
> bcopy:=Array(-k..nn+2):
> profile(genspline):
> Y:=genspline(x,nn,q,r,e,d,alpha,beta):showprofile(genspline):
> p1:=plot([Y,Z(t)],t=0..1,title=" Approximate solution"):
> p2:=plots[pointplot]([[Pi/54,eval(Z(t),t=Pi/54)],[Pi/12,eval(Z(t),t=Pi/12)]],
symbol=circle,symbolsize=30,color=[BLACK]):
> plots[display]({p1,p2}):
> plots[logplot](Y-Z(t),t=0.001..0.999,title="Errors in semi-logarithmic scale",
color=[BLUE],axis=[gridlines=[100, color=blue]]):
> plot([seq(S(t,x,k),k=-1..nn+k)],t=x[-k]..x[nn+k+2],title=" B-splines bases"):
```

=======

Chebyshev Collocation

```
> restart; with(orthopoly):with(CodeTools):with(plots):
The (BVP) Greengard and Rochlin  solve with Collocation Chebishev method.
We study the case $p=9$,$n=10$.
In this situation the error in approximating the exact solution u(x)=sin(p*x)is
of order O(10^-8).
```

```
> S:=(x,k,a,b)->T(k,((b-a)*x+a+b)/2):
> Digits:=30:
> 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:
> n:=34:
> q:=t->-86: r:=t->5*sin(9*t):
> ecz:=diff(Z(t),t$2)-q(t)*Z(t)=r(t):
> dsolve({ecz,Z(0)=0,Z(1)=sin(9)},Z(t)):
> assign(%):
>  b:=1: a:=0: c0:=Pi/54: d0:=Pi/12:
> u:=[Pi/54,seq((b-a)/2*cos(k*Pi/n)+(a+b)/2,k=1..n),Pi/12]:
> u:=sort(evalf(u)):
> n:=nops(u):alpha:=simplify(eval(Z(t),t=Pi/54)):
> beta:=simplify(eval(Z(t),t=Pi/12)):
> x:=Array(0..n-1,u):
> eval(x):profile(genceb):
> Y:=genceb(x,n,q,r,Pi/54,Pi/12,alpha,beta) :showprofile(genceb):
> p1:=plot([Y,Z(t)],t=0..1,title="Approximate solution"):
>p2:=plots[pointplot]([[Pi/54,eval(Z(t),t=Pi/54)],
[Pi/12,eval(Z(t),t=Pi/12)]],
symbol=circle,symbolsize=30,color=[BLACK]):
> plots[display]({p1,p2}):
> plots[logplot](Y-Z(t),t=0.001..0.999,title="Errors in semi-logarithmic scale",
color=[BLUE],axis=[gridlines=[10, color=blue]]):
```

=======

5.2 Nonlinear form

5.2.1 Global B-splines

=======

```
%Nonlinear equation Goldner(Global B-splines)
%y"+y^3+(4-(t-t^2)^3)/(t+1)^3=0
%inner conditions: y(1/4)=3/20;y(1/2)=1/6;
%exact solution y=(t-t^2)/(t+1);
f=@(t,y) -y.^3-(4-(t-t.^2).^3)./(t+1).^3;
a=0; b=1;
% alpha=0;
% beta=0;
fe=@(t) (t-t.^2)./(t+1);
fd=@(x,y) -3*y.^2;
startv=@(t) (t+2)/15;
d=1/4; e=1/2;
alpha=fe(d); beta=fe(e);
N=15;
k=3;
t=linspace(a,b,100);
%Tolerance
err=1e-6;
[x,y]=polycollocnelin(f,fd,a,b,d,e,alpha,beta,N,k,startv,t,err);
close all;
figure(1)
plot(x,startv(x), x,y, [d,e],[alpha,beta],'o');
title('Goldner nonlinear equation-Global B-spline','FontSize',10)
legend('Start sol.','Approx sol.','inner points: d=1/4,e=1/2',0);
grid on
figure(2)
semilogy(x,abs(fe(x)-y))
xlabel('x','FontSize',10)
ylabel('ln |y(x)-y_{\Delta}(x)|','FontSize',10)
title('Error in semilogarithmic scale','FontSize',10)
grid on
```

5.2.2 Combined Method B-splines and Runge-Kutta

=======

```
%Costabile combined method B-Spline and Runge-Kutta
%y''+exp(-y)=0; in [0,1]
%inner conditions: y(pi/6)=ln(3/2), y(pi/4)=ln((2+sqrt(2))/2)
%exact solution y=ln(sin(x)+1)
%startv =start solution
%N=number of meshpoints
%k= degree of B-splines
```

```
%err=tolerance

f=@(x,y) -exp(-y);
fd=@(x,y) exp(-y);
fe=@(x) log(sin(x)+1);
N=11;
k=5;
a=0; b=1;
d=pi/6; e=pi/4;
p3=(d+e)/2;
alpha=fe(d); beta=fe(e);
t=linspace(a,b,100);
err=1e-6;
%construction of start  solution using polyfit on :d,e,(d+e)/2
u=[d,e,p3]; v=fe(u);
c=polyfit(u,v,2);
startv=@(x) c(1)*x.^2+c(2)*x+c(3);
tic;
[x,y]=polycalnlinRK(f,fd,a,b,d,e,alpha,beta,N,k,startv,t,err);
toc;
close all;
figure(1)
plot(x,startv(x), x,y, [d,e],[alpha,beta],'o');
title('Nonlinear eq. Costabile-Combined method Bs+Rk','FontSize',10)
legend('start sol.','approx. sol.','inner points: d=pi/6,e=pi/4',0);
grid on
figure(2)
semilogy(x,abs(fe(x)-y))
xlabel('x','FontSize',10)
ylabel('ln |y(x)-y_{\Delta}(x)|','FontSize',10)
title('Error in semilogarithmic scale','FontSize',10)
grid on
```

=======

```
function [x,y]=polycollocnelin(f,fd,a,b,d,e,alpha,beta,N,k,startv,t,err)
%POLYCOLLOCNELIN solve a nonlinear polylocal problem
%y"(x) =f(x,y)
% inner conditions: y(d)=alpha, y(e)=beta;
%call spr=polycalnlin(f,a,b,d,e,alpha,beta,N,k,startv,t)
%f - function, define  by (BVP)
%fd - derivative of  f against the second variable
%a,b - endpoints
%d,e - inner points
%alpha, beta - values at inner-points
%N -number of subintervals
%k - number of Legendre points
%startv - starting value (function)
```

```
%t - evaluation points
%err - tolerance
%x,y - solution (output)
%prepare mesh
%n=k*N+2; number of mesh-points

ok=0; while(~ok)
breaks=linspace(a,b,N+1);
degree=k+1;
order=degree+1;
knots = augknt(breaks,order,k);
rho=LegendrePoints(k);
tau=[];
for i=1:N
    tau=[tau,(breaks(i+1)+breaks(i)+rho*(breaks(i+1)-breaks(i)))/2];
end
%check d,e
tauv=sort([tau,d,e]);
pozdv=find(tauv = = d);
pozev=find(tauv = = e);
if (length(pozdv)~=1 ||
length(pozev)~=1)
warning(' d or e illegal! increment N');
    N=N+1;
else
    ok=1;
end
end
%prepare iteration
x=sort([a,tau,b,d,e])';
tl=(x~=d & x~=e);
y=startv(x);
% compute inverse of matrix collocation
Minit=spcol(knots,order, x, 'sparse');
coeffin = Minit\y; size(coeffin);
spi=spmak(knots,coeffin.');
yd2=fnval(fnder(spi,2),x(tl));
%build collocation matrix
xdd=sort([a,b,tau]);
tauxx=sort([d,e,brk2knt(xdd,3)])';
Mcol1=spcol(knots,order,tauxx,'sparse');
v=1:length(tauxx); pozd=find(tauxx==d);
poze=find(tauxx==e);
v([pozd,poze])=[];
vf=v(1:3:end); vd2=v(3:3:end);
vfc=sort([vf,pozd,poze])';
x=x(tl); y=y(tl);
```

```
% iteration while 1
Valjac=repmat(fd(x,y),1,length(x));
Mcol=[Mcol1(vd2,:)-Valjac.*Mcol1(vf,:);
Mcol1([pozd,poze],:)];
rhs=[yd2-f(x,y);0;0];
%solve the nonlinear system
coeffs=coeffin-Mcol\rhs;
if norm(coeffin-coeffs,inf)\<= err
    sp=spmak(knots,coeffs');
return
end
coeffin=coeffs; y=Mcol1(vf,:)*coeffin;
yd2=Mcol1(vd2,:)*coeffin;
end
sp = spmak(knots,coeffs.');
x=t;
y = fnval(sp,x);
%roots of \ Legendre polynomial
function r=LegendrePoints(k)
switch k
    case 2 p=[3/2,0,-1/2];
    case 3 p=[5/2,0,-3/2,0];
    case 4 p=[35/8,0,-15/4,0,3/8];
    case 5 p=[63/8,0,-35/4,0,15/8,0];
    case 6 p=[231/16,0,-315/16,0,105/16,0,-5/16];
    case 7 p=[429/16,0,-693/16,0,+315/16,0,-35/16,0];
otherwise
    p=[6435/128,0,-3003/32,0,+3465/64,0,-315/32,0,+35/128];
end
r=sort(roots(p))';
```

=======

```
function [x,y]=polycalnlinRK(f,fd,a,b,d,e,alpha,beta,N,k,startv,t,err)
%POLYCALNLIN solve a nonlinear polylocal problem B-splines combined with
%Runge-Kutta methods
%y"=f(x,y)
% inner conditions: y(d)=alpha, y(e)=beta;
%call spr=polycalnlin(f,a,b,d,e,alpha,beta,N,k,startv,t)
%f - function, define BVPN or BVPL
%fd - derivative of f depending on second variable
%a,b - endpoints
%d,e - inner points
%alpha, beta - values at inside points
%N - number of subintervals
%k - number  of Legendre points
%startv - starting value (function)
%t - evaluation points
```

```
%err - error
%x,y - solution (output)
%prepare meshes
ti=t(t>=d & t<=e);
tleft=t(t<d);
tright=t(t>e);
%solve BVP
[sp]=BVPcollocnelin(f,fd,a,b,alpha,beta,N,k,startv,err);
yi=fnval(sp,ti);
ydi=fnval(fnder(sp),[d,e]);
x=ti'; y=yi';
%solver options
opts=odeset('AbsTol',1e-8,'Reltol',1e-7);
%solve right (IVP)
if ~is empty(tright)
    tright=[e,sort(tright)];
    [tr,wr]=ode45(@rhs,tright,[beta,ydi(2)],opts);
    x=[x;tr(2:end)]; y=[y;wr(2:end,1)];
end
%solve left (IVP)
IVP if ~is empty(tleft)
    tleft=[d,sort(tleft,'descend')];
    [tl,wl]=ode45(@rhs,tleft,[alpha,ydi(1)],opts);
    x=[tl(end:-1:2);x]; y=[wl(end:-1:2,1);y];
end
% nested function for Runge-Kutta method
function dy=rhs(x,u) dy=[u(2); f(x,u(1))];
end
end
```

5.2.3 Combined Method: Chebyshev Collocation with Runge-Kutta

=======

```
%the temperature of reaction-diffusion problem, p=3
%u"+u^3=0; u(0)=u(1)=0;
%inner conditions: u(0.2)=u(0.8)=1.929990320692795
%f0=start solution

f=@(x,y) -y.^3;
df=@(x,y) -3*y.^2;
err=1e-8;
NMAX=50;
N=128;
f0=@(x) 12*pi/sqrt(2)*x.*(1-x);
a=0; b=1;
c=0.2;
d=0.8;
```

```
alpha=1.92990320692795; bet=1.92990320692795;
tic;
[x,y,ni]=solvepolylocalceb(N,f,df,a,b,c,d,alpha,bet,f0,err,NMAX);
toc;
ni;
figure(1);
plot(x,y,[c,d],[alpha,bet],'o')
title('The average temp. of reaction-diffusion eq.  C.C+R-K method')
grid 'on'
```

5.2.4 Compare methods

=======

```
%Bratu's nonlinear problem (Lamda=1)
%Compare methods: Global B-splines, B-splines+Runge-Kutta,C.C+Runge-Kutta
% start solution : y(x)=x*(1-x)
%u"+exp(u)=0;
%inner conditions: u(0.2)=u(0.8)=0.08918993462883;
%this values are computed using Maple codes
%err=tolerance
%NMAX=max number of iterations
%N=number of degree freedom
%ni=number of iterations

f=@(x,y) -exp(y);
df=@(x,y) -exp(y);
err=1e-10;
NMAX=50;
N=15;
f0=@(x) x.*(1-x);
a=0; b=1;
c=0.2;
d=0.8;
alpha=0.08918993462883; bet=0.08918993462883;
%1.Pseudospectral method(C.C)+Runge-Kutta
tic;
[x,y,ni]=solvepolylocalceb(N,f,df,a,b,c,d,alpha,bet,f0,err,NMAX);
toc;
ni;
figure(1);
plot(x,y,[c,d],[alpha,bet],'o')
grid on
title('Bratu's problem CC+RK')
%2.Combined B-splines+Runge-Kutta
k=3;d=0.2;
e=0.8;N=15;
```

```
t=linspace(a,b,100);
tic;
[x,y]=polycalnlinRK(f,df,a,b,d,e,alpha,bet,N,k,f0,t,err);
toc;
figure(2);
plot(x,y,[d,e],[alpha,bet],'x')
grid on
title('Bratu's problem BS+RK')
%3.Global B-splines
N=15;
%start solution
startv=@(x) 3.9*x.*(1-x)/7;
[x,y]=polycollocnelin(f,df,a,b,d,e,alpha,bet,N,k,startv,t,err);
toc;
figure (3);
plot(x,y,[d,e],[alpha,bet],'*')
grid on
title(' Bratu' problem- global method B-Splines')
```

=======

```
function [X,y,ni]=solvebilocalceb(N,f,dfdy,alpha,bet,f0,err,NMAX)
%SOLVBILOCALCEB - solution of a BVP with Chebyshev collocation
%call [X,y,ni]=solvebilocalceb(N,f,dfdy,alpha,bet,u0,err,NMAX)
%x - abscissas
%y - solution values on Chebyshev grid
%f0 - starting value (function)
%ni - eff # of iters
%N - #points
%f - rhs function
%dfdy - derivative of f wrt y
%alpha, bet - boundary values
%err - error
%NMAX max # of iter.s.

if nargin<8, NMAX=20; end
if nargin<7, err=1e-8; end
[X,D]=chebdif(N,2);
if nargin<6
    u0=zeros(N,1);
else
    u0=f0(X);
end
i=2:N-1;
I=eye(N);
u=u0;
for k=1:NMAX % Newton iterations
    F=[D(i,:,2)*u-I(i,:)*f(X,u); u(N)-bet; u(1)-alpha];
```

```
    dF=[D(i,:,2)-I(i,:)*diag(dfdy(X,u));I(N,:);I(1,:)];
    un=u-dF\F;
    if norm(u-un,inf)<err
        %y=[alpha;un;bet];
        y=un;
        ni=k;
        return
    end
    u=un;
end
error('too many iterations')
```

=======

```
function [X,Y,ni]=solvepolylocalceb(N,f,dfdy,a,b,c,d,alpha,bet,f0,err,NMAX)
%SOLVBILOCALCEB - solution of a BVP with Chebyshev collocation
%call [X,y,ni]=solvepolylocalceb(N,f,dfdy,a,b,alpha,bet,f0,err,NMAX)
%x - abscissas
%y - solution values on Chebyshev grid
%f0 - starting value (function)
%ni - number of iterations
%N - number of mesh points
%f - rhs function
%dfdy - derivative of f wrt y
%a,b - interval endpoints
%c,d - inner point
%alpha, bet - boundary values
%f0 starting function
%err - tolerance
%NMAX max number of iterations.

if nargin<12, NMAX=50; end
if nargin<11, err=1e-3; end
g=@(t,y) (d-c)^2/4*f(((d-c)*t+(c+d))/2,y);
dg=@(t,y) (d-c)^2/4*dfdy(((d-c)*t+(c+d))/2,y);
if (nargin<10)||isempty(f0)
    g0=@(t) zeros(size(t));
else
    g0=@(t) f0(((d-c)*t+(d+c))/2);
end
[x,y,ni]=solvebilocalceb(N,g,dg,alpha,bet,g0,err,NMAX);
xx=linspace(-1,1,10*N)';
Y=chebint(y,xx);
format long
%chebint(y,[-0.6,0.6])
X=((d-c)*xx+(c+d))/2;
D1f=chebdifft(y, 1);
Dd=2/(d-c)*D1f(1); Dc=2/(d-c)*D1f(end);
```

```
Dd1=double(Dd);
Dc1=double(Dc);
%solver options
opts=odeset('AbsTol',err,'Reltol',1e-7);
%solve right IVP
if d~=b
     tright=[d,b];
     [tr,wr]=ode23tb(@rhs,tright,[bet,Dd1],opts);
     X=[X;tr(2:end)];
     Y=[Y;wr(2:end,1)];
end
% solve left IVP
if c~=a
    tleft=[c,a];
    [tl,wl]=ode23tb(@rhs,tleft,[alpha,Dc1],opts);
    X=[tl(end:-1:2);X];
    Y=[wl(end:-1:2,1);Y];
end
% nested function for RK method
    function dy=rhs(x,u)
        dy=[u(2); f(x,u(1))];
    end
end
```

Bibliography

[1] O.Agratini, I.Chiorean, G.Coman, R.Trîmbiţaş, *Analiză numerică şi Teoria Aproximării,* Presa Universitară Clujeană, Cluj-Napoca, 2002 (in romanian).

[2] V.Anisiu, *Calcul simbolic cu Maple,* Presa Universitară Clujeană, 2006 (in romanian).

[3] U.Ascher, J. Christiansen, and R.D. Russel, *A Collocation Solver for Mixed Order Systems of Boundary Value Problem*, Mathematics of Computation, vol. 33(146), pp: 659-679, 1979.

[4] U.Ascher, S.Pruess and R.D. Russel, *On spline basis selection for solving differential equations,* SIAM J. Numer. Anal. 20 nr:1, 1983.

[5] U. Ascher, R.M. Mattheij, and R.D. Russel, *Numerical Solution of Boundary Value Problems for Ordinary Differential Equations*, SIAM, 1997.

[6] U. Ascher, *Solving Boundary Value Problems with a Spline-Collocation Code,* Journal of Comput. Physics, vol **34**(3), pp: 401-413, 1980.

[7] U. Ascher, *Discrete least squares approximations for ordinary differential equations,* SIAM J.Numer.Anal.,**15**, pp: 478-496, 1978.

[8] C. de Boor, *A practical guide to splines,* Springer-Verlag, Berlin, Heidelberg, New York, 1978.

[9] C. deBoor, *Package for calculating with B-spline,* SIAM J. Numer. Anal., 14, No.3, pp: 441-472, 1977.

[10] C. deBoor, *Spline Toolbox 3,* MathWorks Inc, Nattick, MA, 2008.

[11] C. deBoor, *Good approximation by splines with variable knots II,* Lecture Notes in Mathematics Series 363. New York, Springer Verlag, 1973.

[12] C. deBoor, B. Swartz, *Collocation at Gaussian points,* SIAM J. Numer. Anal., Vol10, pp: 582-606, 1973.

[13] C. deBoor, R. Weiss, *SOLVEBLOCK: A Package for Solving Almost Block Diagonal Linear Systems, with Application to Spline Approximations and the Numerical Solution of Ordinary Differential Equations,* MRC TSR #1625, Madison, Wisconsin, 1976.

[14] L.R. Burden, J.D. Faires, *Numerical Analysis*, PWS-Kent Publishing Company, New York, 1985.

[15] C. Canuto, *Boundary Conditions in Chebishev and Legendre Methods,* SIAM J.Numer.Anal.23, pp: 815-831, 1986.

[16] F.A Costabile, F.Dell' Accio, *On the nonsingularity of a special class of centrosymetric matrices arising in spectral methods in BVPs,* Applied Mathematics and Computation 206, pp: 991-993, 2008.

[17] F.A Costabile, A. Napoli, *A collocation method for global approximation of general second order BVPs,* Computing Letters 3, pp: 23-24, 2007.

[18] J. Christiansen, R.D. Russel, U. Ascher, *A Collocation Solver for Mixed Order Systems of Boundary Value Problems,* Math.Comp., **13,** no. 146, pp 659-679, 1979.

[19] J. Christiansen, R.D. Russel, *Error analysis for spline collocation methods with application to knot selection,* Math.Comp-v.32, pp: 415-419, 1978.

[20] J.W. Demmel, *Applied Numerical Linear Algebra,* SIAM, 1997.

[21] **E. Drăghici, Daniel N.Pop,** *Solution of a polylocal problems using Chebishev polynomials,* General Mathematics, Vol.16, Nr.4, pp: 47-59, Sibiu, 2008.

[22] L. Fox, I.B Parker, *Chebishev Polynomials in Numerical Analysis,* Oxford Mathematical HandBooks, O.U.P, 1968.

[23] W. Gautschi, *Numerical Analysis, An Introduction,* Birkhauser Verlag, Basel, 1997.

[24] C.I. Gheorghiu, *Spectral methods for differential problems,* Casa Cărţii de Ştiinţă, Cluj Napoca, 2007.

[25] C.I Gheorghiu, D. Trif, *The numerical approximation to positive solution of some reaction -diffusion problems,* P.U.M.A 11, pp: 243-253, 2000.

[26] G. Goldner, Radu T.Trîmbiţaş, *A combined method for two point boundary value problem,* P.U.M.A, vol 11, No2, pp: 255-264, 2000.

[27] L. Greengard, V. Rokhlin, *On the Numerical Solution of Two-Point Boundary Value Problems,* Comm.Pure Appl.Math.XLIV, pp: 419-452, 1991.

[28] E. Hairer, S.P Norset, G. Wanner, *Solving Ordinary Differential Equations I, Nonstiff Problems, Second Revised Editions,* Springer Verlag Berlin, New-York, Heidelberg, 2000.

[29] A.Heck, *Introduction to Maple,* Springer-Verlag, Berlin, Heidelberg, 1997.

[30] P. Henrici, *Discrete Variable Methods in Ordinary Differential Equations*, John Willey and Sons, New York, 1962.

[31] W.Heinrichs, *Strong convergence Estimates for Pseudospectral Methods,* Appl. Math. pp:401-417, 1992.

[32] D. Hinton, *Sturm's 1836 Oscillation Results -Evolution of the Sturm-Liouville Theory, Past &Present,* W.O. Amrein, A.M. Hinz, D.B. Pearson(editors) Birkhauser Verlag,Basel-Boston-Berlin, pp: 1-29, 2005.

[33] E.Houstis, *A collocation method for systems of nonlinear ordinary differential eqution,* J. Math.Anal. Appl., **62** pp: 24-37, 1978.

[34] C.Lanczos, *Applied Analysis,* Prentice Hall Inc., Englewood Cliffs, N.J, 1956.

[35] A.Lupaş, *Metode numerice,* Editura Constant, Sibiu, 2001 (in romanian).

[36] Patrick A.Domenico, Franklin W.Schwartz., *Physical and Chemical Hydrology (Second edition),* John Willey and Sons, Inc. 1998.

[37] I. Păvaloiu, N. Pop, *Interpolare şi Aplicaţii,* Risoprint, Cluj-Napoca, 2005 (in romanian).

[38] S.M Pizer, W.L Wallance, *To Compute Numerically. Concepts and Strategies,* Little, Brown and Company, 1983.

[39] Mathlab 2010.a, *Set of manuals,* MathWorks, Inc-Natick-MA, 2010.

[40] S.Micula, G.Micula, *Handbook of Splines,* Kluwer Academic Publishers, Netherlands, 1999.

[41] M.A. Diminescu, V. Nistorean, *Gospodărirea durabilă a resurselor de apă,* Conf. Internaţ. Energie-Mediu, CIEM Bucureşti, 2005 (in romanian).

[42] **Daniel N.Pop,** *A Collocation Method using Cubic B-Splines Functions for solving second order linear value problems with condition inside the interval* $[0,1]$*,* Studia Univ."Babeş-Bolyai", Mathematica, Volume LV, Number 2, pp: 177-187, June, Cluj-Napoca, 2010.

[43] **Daniel N.Pop, Radu T.Trâmbitas**, *A comparison between two collocation methods for linear polylocal problems-a Computer Algebra based approach,* International Conferences in Modelling and Development of Intelligent Systems-MDIS'09, pp: 164-174, Sibiu, 2009.

[44] **Daniel N. Pop, Radu T.Trâmbitas,** {{*New Trends in Approximation, Optimization and Classification*}}*, ch.*Solution of a polylocal problem - a Computer Algebra based approach*,* Lucian Blaga University Press, Proceedings of International Workshop New Trends in Approximation, Optimization and Classification, Sibiu, pp: 53-63, Sibiu, 2008.

[45] **Daniel N.Pop**, *Error bound for the solution of a polylocal problem with combined methods,* Studia Univ."Babeş-Bolyai" Mathematica, vol LIV, Number 4, pp: 115-123, Cluj-Napoca, 2009.

[46] **Ion Păvăloiu, Daniel N.Pop, Radu T.Trâmbitas,** *Solution of polylocal problem with a pseodospectral method,* Annals "Tiberiu Popovici", Seminar of functional equations, approximaion and convexity, vol 8, pp: 53-65, Cluj-Napoca, 2010.

[47] **Daniel N.Pop**, **Radu T.Trâmbitas,** *An approximation methods for second order nonlinear value polylocal problems using B-splines and Runge-Kutta metods,* N.A.T.2010, Studia Univ."Babeş-Bolyai" Mathematica, vol LVI, Number 2, pp: 515-527, Cluj-Napoca, 2009. .

[48] J.J Riessler, *Méthodes mathématiques pour la CAO,* Editure Macon, Paris, Milan, Barcelone, Bonn, 1991.

[49] I.A Rus, Paraschiva Pavel, *Ecuaţii Diferenţiale,* Editura Didactică şi Pedagogică, Bucureşti, 1982 (in romanian).

[50] R.D.Russel, L.F.Shampine, *A coolocation Method for Boundary Value Problems,* Numer.Math.19, Springer-Verlag, pp: 1-28, 1972.

[51] R.D Russel, *Efficients of B-splines methods for solving differential eqution, Proc Fifth Conference on Numerical Mathematics,* Utilitas Math.Winipeg, Manitoba, pp: 599-617, 1975.

[52] R.D Russel, *A comparation of collocation and finite differences for two point boundary value problems,* SIAM, J.Numer., Anal vol 14, pp: 19-39, 1977.

[53] R.D Russel, J. Christiansen, *Adaptive mesh selection strategies for solving boundary value problems,* SIAM, J.Numer.Anal v.15, pp: 59-80, 1978.

[54] R.D Russel, *Mesh selection methods, in Codes for Boundary Value Problems,* Lecture Notes in Computer Science 74 Childs et al.eds, Springer Verlag, Berlin, 1979.

[55] L.F Shampine, *Design of software for ODE,* J. Comput. Appl. Math, 205, 2007.

[56] L.F. Shampine, *Conservation laws and the numerical solution of ODE's ,* Comput. Math. Appl. 12B, pp:1287-1396, 1986.

[57] L.F Shampine, I. Gladwell, S. Thompson, *Solving ODE's with MATLAB,* Cambridge University Press, the Edinburgh Building Cambridge, United Kingdom, 2003.

[58] L.F Shampine, J. Kierzenka, M.W. Reichelt, *Solving Boundary Value Problems for Ordinary Differential Equations in MATLAB with* `bvp4c`*,* J. Comput. Appl. Math, 26, 2000.

[59] L.F Shampine, J. Gladwell, S. Thomson, *Solving Ode's with MATLAB,* Cambridge University Press, 2003.

[60] M.H Schultz, *Spline Analysis,* Prentice Hall, Inc. Englewood Cliffs, New Jersey, 1972.

[61] A. Solomonoff, E. Turkel, *Global Properties of Pseudospectral Methods,* J. Comp. Phys.81, pp: 239-276, 1989.

[62] A. Solomonoff, *A Fast Algorithm for Spectral Differentiation,* J. Comput. Phys.81, pp. 239-276, 1989.

[63] B. Swartz, *Conditioning Collocation,* SIAM J. Numer. Anal, Vol. **25**(1), pp: 124-146, 1988.

[64] L.N Trefeten, *Spectral methods in MATLAB,* SIAM, Philadelphia, PA, 2000.

[65] Radu T.Trîmbiţaş, *Numerical Analysis,* Cluj Universitary Press, 2006.

[66] Radu T.Trîmbiţaş**,** *Numerical Analysis in MATLAB,* Presa Universitară Clujeană, 2009.

[67] G.M. Vainniko, *On the stability and convergence of collocation method,* Differentiasial'nye Uravneniya vol. 1, pp: 244-254, 1965.

[68] J.A.C Weideman, S.C Reddy, *A MATLAB Differentiation Matrix Suite,* A.C.M Trans. on Math. Software, 26, pp: 465-519, 2000.

[69] Z.Y Liu, *Some properties of centrosymmetric matrices,* ELSEVIER, Applied Mathematics and Computation 141, pp: 297-306, 2003.

Index