David S. Gilliam
Department of Mathematics
Texas Tech University
Lubbock, TX 79409

806 742-2566
gilliam@texas.math.ttu.edu
http://texas.math.ttu.edu/~gilliam

A Survey of Some Methods in Numerical DEs

1 A Hodge Podge of Methods for ODEs and PDEs

Method: Collocation Method (a Weighted Residual Method)

Given a differential equation $L[u] = 0$ for $u(\xi)$ with $\xi \in Q$ (Q some domain) and boundary conditions $B[u] = 0$, we seek an approximate solution $u(\xi) = w(\xi; \alpha)$ where $\alpha = \{\alpha_j\}_{j=1}^N$ is a set of parameters and $B[w] = 0$ for all choices of α.

We try to determine the α by requiring that the equation be satisfied by w at a set of points $S = \{\xi_j\}_{j=1}^N \subset Q$. Thus we arrive at a system of N equations in the N unknowns α:

$$L[w(\xi_j, \alpha)] = 0, \quad j = 1, \cdots, N.$$

Example
Consider

$$\begin{aligned} L[u] &= u'' + u + x = 0, \\ u(0) &= 0, \quad u(1) = 0 \end{aligned} \tag{1.1}$$

Suppose we choose to approximate the solution by

$$w(x) = \alpha_1 x(1-x) + \alpha_2 x(1-x^2).$$

Note that w satisfies the boundary conditions. A straightforward calculation gives:

$$L[w(x)] = -\alpha_1(2 - x + x^2) - \alpha_2(5x + x^3) + x.$$

Applying our collocation method at $x_1 = 1/3$ and $x_2 = 2/3$, we obtain the 2×2 system

$$\begin{gathered}(48/27)\alpha_1 + (46/27)\alpha_2 = 1/3 \\ (48/27)\alpha_1 + (98/27)\alpha_2 = 2/3\end{gathered}$$

which has solutions $\alpha_1 = 9/416$ and $\alpha_2 = 9/52$. Thus we obtain

$$w(x) = \frac{9}{416}x(1-x) + \frac{9}{52}x(1-x^2).$$

The exact solution to this problem is

$$u(x) = \frac{\sin(x)}{\sin(1)} - x$$

and the maximum difference between y and w on $[0,1]$ occurs at $x = .7916$ with a value of .00081.

Assignment 1:

1. Program this example, plot y and w and check the error given above.
2. Try using two different functions for example. (maple or mathematica can be help in these calculations).
3. Try using different points x_1 and x_2, e.g. $x_1 = 1/4$ and $x_2 = 3/4$.
4. Approximate the solution using three values for α.

References

1. K. Wright, A.H.A. Ahmed and A.H. Selemann, "Mesh selection in collocation Boundary Value Problems," *IMA J. Num. Analysis*, 11, No. 1, Jan 1991, 7-20.
2. U. Ascher, J. Christiansen and R.D. Russell, " Collocation software for boundary value problems ODEs," *ACM Trans. Math. Software* 7, No. 2, June 1981, 209-222.
3. E.N. Houstis, W.F. Mitchell and J.R. Rice, "Collocation software for second order elliptical PDEs," 11, No. 4, Dec. 1985, 379-412.
4. L. Collatz, "The numerical treatment of Differential Operators," Springer-Verlag, NY, 1966.

Method: Least Squares Method (a Weighted Residual Method)

First a variational principle is created for a given differential equation and then an approximation to the solution is sought which contains some free parameters. By use of the *Variational Principle*, the free parameters are determined and approximate solution is obtained.

For example, given a differential equation $L[u] = 0$ for $u(\xi)$ in a region Q with homogeneous boundary conditions $B[u] = 0$ given on some portion of the boundary of Q, we define the functional

$$J[v(\xi)] = \int_Q \Big(L[v(\xi)] \Big)^2 \, d\xi.$$

Note that the actual solution u to $L[u] = 0$ and $B[u] = 0$ satisfies $J[u] = 0$.

Our goal is to choose an approximation $w(\xi; \alpha)$ to u that contains some parameters $\alpha = \{\alpha_j\}_{j=1}^N$ and so that and $B[w] = 0$ for all choices of α.

Then we obtain an approximate minimum by requiring that the following equations be satisfied:

$$\frac{\partial}{\partial \alpha_k} J[w(\xi, \alpha)] = 0, \quad k = 1, \cdots, N.$$

References

1. C.L. Chang and M.D. Gunzburger, "A subdomain-Galerkin/least squares method for first-order elliptic systems in the plane," *SIAM J. Num. Analysis*, 27, No. 5, 1990, 1197-1211.

2. L. Collatz, "The numerical treatment of Differential Operators," Springer-Verlag, NY, 1966, p. 220-221.

Example

Consider once again the example

$$\begin{aligned} L[u] &= u'' + u + x = 0, \\ u(0) &= 0, \quad u(1) = 0 \end{aligned} \tag{1.2}$$

Suppose we again choose to approximate the solution by

$$w(x) = \alpha_1 x(1-x) + \alpha_2 x(1-x^2).$$

Note that w satisfies the boundary conditions. In this case we may define

$$J[v(x)] = \int_0^1 (v'' + v + x)^2 \, dx.$$

Using w in the functional be get

$$J[w] = \frac{1}{210} \Big[707\alpha_1^2 + 2121\alpha_1\alpha_2 + 2200\alpha_2^2 - 385\alpha_1 - 784\alpha_2 + 70 \Big].$$

Applying the partials with respect to α_j we arrive at:

$$\frac{\partial J[w]}{\partial \alpha_1} = \frac{1}{210}\Big[1414\alpha_1 + 2121\alpha_2 - 385\Big] = 0$$
$$\frac{\partial J[w]}{\partial \alpha_2} = \frac{1}{210}\Big[2121\alpha_1 + 4400\alpha_2 - 784\Big] = 0$$

which yields solutions

$$\alpha_1 = 4448/246137 \approx .0181, \quad \alpha_2 = 413/2437 \approx .1694.$$

In this case the greatest difference between u and w occurs at $x = .5215$ where the difference is approximately .0016.

Assignment 2:

1. Program this example, plot y and w and check the error given above.
2. Try using two different functions for example. (maple or mathematica can be help in these calculations).
3. Approximate the solution using three values for α.

Method: Taylor Series

This method yields an approximate solution to a differential equation near a single point a. The method is applicable for differential equations in which all expressions are analytic functions of the variables. The idea is to use the differential equation to obtain the coefficients in a Taylor series about the point a. We note that this is really an application of the famous Cauchy-Kovalevskaya theorem.

References

1. V.P. Igumnov, "Representations of solutions to differential equations by modified Lie series," *Diff. Equations* 20, 1984, 683-688.
2. M. Razzaghi and M. Razzaghi, "Solution of linear two-point boundary value problems via Taylor series," *J. Franklin Inst.*, 326, No. 4, 1989, 511-521.
3. Y.F. Chang and G.F. Corliss, "Solving ode's using Taylor series," *ACM Trans. Math Software*, 8, 1982, 114-144.
4. N. Finizio and G. Ladas, "Ode's with modern applications," Wadsworth Pub. Company, Belmont Calif., 1982, 116-120.

Example

We seek an approximate solution to

$$y'(x) = F(x,y), \quad y(a) = y_0$$

in the form

$$y(x) = \sum_{j=0}^{N} \frac{y^{(j)}(a)}{j!}(x-a)^j.$$

The idea is to use the differential equation and initial condition to find the Taylor coefficients.

$$\begin{aligned} y(a) &= y_0 \\ y' =& F(x,y) \text{ implies} \\ y'(a) &= F(a,y_0) \\ y''(x) =& F_x(x,y) + F_y(x,y)y_x \text{ implies} \\ y''(a) &= F_x(a,y_0) + F_y(a,y_0)F(a,y_0), \\ &\text{etc.} \end{aligned}$$

Consider

$$\begin{aligned} y' &= x^2 - y^2, \\ y(0) &= 1 \end{aligned}$$

A straightforward calculation yields:

$$\begin{aligned} y' &= x^2 - y^2, \\ y'' &= 2x - 2yy' \\ y''' &= 2 - 2(y')^2 - 2yy' \\ y'''' &= -6y'y'' - 2yy''' \end{aligned}$$

from which we immediately obtain for $a = 0$

$$\begin{aligned} y'(0) &= -1, \\ y''(0) &= 2 \\ y'''(0) &= -4 \\ y''''(0) &= 20. \end{aligned}$$

Thus we obtain the approximate solution

$$\begin{aligned} y &= 1 - x + \frac{2}{2!}x^2 - \frac{4}{3!}x^3 + \frac{20}{4!}x^4 \\ &= 1 - x + x^2 - \frac{2}{3}x^3 + \frac{5}{6}x^4 \end{aligned}$$

Assignment 3:

1. Write a Maple or Mathematica program to carry out the Taylor method up to order n for $y' = F(x,y)$ with $y(a) = y_0$. Apply your program with $n = 4, 6, 8, 10$ and plot the results.
2. Apply your program to $y' = y^2,\ y(1) = 1$. How do your answers compare with the exact solution on $[1,2]$?
3. Same question with $y' = 3x^2 y,\ y(0) = 1$ on $[0,1]$.

Method: Variational Principle: Rayleigh-Ritz

The variational expression from which a differential equation is derived can be used to approximate the solution. Note that most equations of mathematical physics and engineering arise from a variational principle.

Typically one has an expression

$$J[u] = \iint\limits_R L(x, \partial_{x_j}) u(x)\, dx$$

from which we form

$$J[u+h] - J[u] = \iint\limits_R \Big\{ L(x, \partial_{x_j})(u(x) + h(x)) - L(x, \partial_{x_j}) u(x) \Big\}\, dx.$$

After some integration by parts, this can often be written as

$$J[u+h] - J[u] = \iint\limits_R N(x, \partial_{x_j}) u(x)\, dx + O(\|h\|^2) + \text{B.T.'s}.$$

The *variational principle* requires

$$\delta J \equiv J[u+h] - J[u]$$

vanish to leading order, i.e.,

$$N(x, \partial_{x_j}) u(x) = 0 \quad \text{Euler-Lagrange Equations.}$$

Examples:

1. The variations expression

$$J[u] = \iint\limits_R \left[a \left(\frac{\partial u}{\partial x} \right)^2 + b \left(\frac{\partial u}{\partial y} \right)^2 + cu^2 + 2fu \right] dx\, dy$$

gives the Euler-Lagrange equation

$$\frac{\partial}{\partial x} \left(a \frac{\partial u}{\partial x} \right) + \frac{\partial}{\partial y} \left(a \frac{\partial u}{\partial y} \right) - cu = f.$$

2. For ordinary differential equations, the variational expression

$$J[u] = \int_a^b \left(\sum_{k=0}^m p_k(x) \left(\frac{d^k}{dx^k} u \right)^2 - 2f(x)u(x) \right) dx$$

gives

$$\sum_{k=0}^m (-1)^k \frac{d^k}{dx^k} \left(p_k(x) \frac{d^k u(x)}{dx^k} \right) = f(x)$$
$$u^{(j)}(a) = 0, \ j = 0, 1, \cdots, (m-1),$$
$$u^{(j)}(b) = 0, \ j = 0, 1, \cdots, (m-1),$$

3. If $A_{ij}(x)$ is a symmetric positive definite matrix for each x in a region and $c(x)$ is positive, then the variational expression

$$J[u] = \int_\Omega \left(\sum_{i,j=1}^m A_{i,j} \frac{\partial u}{\partial x_i} \frac{\partial u}{\partial x_j} + cu^2 - 2uf \right) dx$$

gives the Elliptic dirichlet boundary value problem

$$-\sum_{i,j=1}^m \frac{\partial}{\partial x_i} \left(A_{i,j} \frac{\partial u}{\partial x_j} \right) + cu = f,$$
$$u\Big|_{x \in \partial\Omega} = 0.$$

4. If $A_{ij}(x)$ is a symmetric positive definite matrix for each x in a region and $c(x)$ is positive, then the variational expression

$$J[u] = \int_\Omega \left(\sum_{i,j=1}^m A_{i,j} \frac{\partial u}{\partial x_i} \frac{\partial u}{\partial x_j} + cu^2 - 2uf \right) dx + \int_{\partial\Omega} \sigma u^2 \, dS$$

gives the Elliptic oblique derivative boundary value problem

$$-\sum_{i,j=1}^m \frac{\partial}{\partial x_i} \left(A_{i,j} \frac{\partial u}{\partial x_j} \right) + cu = f,$$
$$\left[\sum_{i,j=1}^m \left(A_{i,j} \frac{\partial u}{\partial x_j} \cos(\nu, x_i) \right) + \sigma u \right] \Big|_{x \in \partial\Omega} = 0.$$

Rayleigh-Ritz Method: The Rayleigh-Ritz Method is to find an approximate minimum of the the form $J[u]$. Typically the procedure goes like this. Choose a sequence of functions $\{\phi_j(x)\}_{j=1}^N$ that satisfy the "natural boundary conditions" and form a linear combination

$$u^N(x) = \sum_{j=1}^N a_j \phi_j(x).$$

Now find the unknowns $\{a_j\}_{j=1}^N$ satisfying

$$\frac{\partial}{\partial a_j} J[u^N] = 0, \quad j = 1, 2, \cdots, N.$$

Example:

Consider the elliptic problem (Poisson problem)

$$\begin{aligned}
&u_{xx} + u_{yy} = \sin(\pi x), \text{ for } 0 < x < 1,\ 0 < y < 1, \\
&u = 0, \text{ on } \ x = 0, x = 1, y = 0, y = 1
\end{aligned}$$

This problem comes from the variational principle $\delta J = 0$ where

$$J[u] = \int_0^1 \int_0^1 \left(u_x^2 + u_y^2 + 2u\sin(\pi x)\right)\, ds\, dy.$$

We choose to approximate $u(x, y)$ by a linear combination of

$$\begin{aligned}
\phi_1(x, y) &= x(1-x)y(1-y) \\
\phi_2(x, y) &= x^2(1-x)y(1-y) \\
\phi_3(x, y) &= x(1-x)y^2(1-y)
\end{aligned}$$

Since all the ϕ_j vanish on the boundary so will $u^3(x, y)$. A straightforward calculation gives the following expression to be minimized.

$$\begin{aligned}
\big[24\pi^3 a_3^2 + (35\pi^3 a_2 + 70\pi^3 a_1 + 2100) + 24\pi^3 a_2^2 \\
+ (70\pi^3 a_1 + 2100)a_2 + 70\pi^3 a_1^2 + 4200 a_1\big]/(3150\pi^3)
\end{aligned}$$

Differentiating this with respect to a_1, a_2 and a_3 and setting equal to zero results in the following system to be solved:

$$\begin{pmatrix} 140\pi^3 & 70\pi^3 & 70\pi^3 \\ 70\pi^3 & 48\pi^3 & 35\pi^3 \\ 70\pi^3 & 35\pi^3 & 48\pi^3 \end{pmatrix} \begin{pmatrix} a_1 \\ a_2 \\ a_3 \end{pmatrix} = \begin{pmatrix} -4200 \\ -2100 \\ -2100 \end{pmatrix}$$

with solution

$$\left\{ a_1 = -\frac{30}{\pi^3},\ a_2 = 0,\ a_3 = 0 \right\}.$$

Thus our approximate solution is

$$u^3(x, y) = -\frac{30}{\pi^3} x(1-x)y(1-y).$$

The exact solution is

$$u(x, y) = \frac{\sin(\pi x)}{\pi^2 \sinh(\pi)} \left[\sinh(\pi y) + \sinh(\pi(1-y)) - \sinh(\pi)\right].$$

References

1. L.V. Kantorovich and V.I. Krylov, "Approximation methods for higher analysis," Interscience, NY, 1958, Chap 4, p 241-357.
2. I. Stakgold, "Greens functions and boundary value problems," John Wiley and Sons, NY, 1979, p 539-544.
3. E. Zauderer, "Partial Differential Equations of Applied Mathematics," John Wiley and Sons, NY, 1983, p 470-483.

Assignment 4:

1. Write a Maple or Mathematica program to check the above calculations, compute the error on a fine grid and plot the results.
2. Consider the boundary value problem

$$\begin{aligned} -u'' + ku &= 1,\ 0 < x < 1 \\ u(0) &= u'(1) = 0. \end{aligned}$$

Use the Rayleigh-Ritz method with $\phi_1(x) = x$ and $\phi_2(x) = x^2$. Carry out the calculations using general k as far as possible for u^1 and u^2. Then for numerical examples consider the two cases $k = 0$ and $k = 1$. For comparison note that the closed form solutions are:

$$\begin{aligned} u(x) &= \frac{1}{k} - \frac{\cosh(\sqrt{k}(x-1))}{k\cosh(\sqrt{k})},\ k > 0, \\ u(x) &= x - \frac{x^2}{2},\ k = 0. \end{aligned}$$

Note that in this case the form J is given by

$$J[u] = \int_0^1 \left(|u'(x)|^2 + k|u(x)|^2 - 2u(x)\right) dx$$

where we have noted that the energy form corresponding to the first to terms contains a factor of $\frac{1}{2}$. Multiplying through by 2 gives the form of J given above.

The corresponding values are

$$\begin{aligned} J[u] &= -\frac{1}{k} + k^{-3/2}\tanh(\sqrt{k}),\ k > 0, \\ J[u] &= -\frac{1}{3}, k = 0. \end{aligned}$$

2 Numerical Methods for ODEs

2.1 Finite Difference Methods for First Order Equations

In this section we will consider the initial value problem for for a first order system in the form:

$$\begin{aligned} &y' = f(t,y),\ t \in [a,b] \\ &y(a) = y_a. \end{aligned} \tag{2.1}$$

Euler's Method:

This method, which can be derived many different ways, is not particularly practical but is by far the simplest method and the starting point for many more complicated methods.

Unlike the methods discussed in the earlier section, this method does not produce a function that can be evaluated at any point but rather is gives approximate values to the solution at points in the form $\{(t_j, y_j)\}_{j=1}^{M+1}$ where

$$y(t_j) \approx y_j, \quad a = t_1 < t_2 < \cdots < t_M < t_{M+1} = b.$$

While any mesh can be used, we will select uniform mesh

$$t_k = a + h(k-1),\ k = 1, \cdots, (M+1), \quad h = \frac{(b-a)}{M}.$$

Assuming that y, y' and y'' are continuous we can apply Taylor's theorem at t_1 to obtain

$$y(t) = y(t_1) + y'(t_1)(t-t_1) + y''(c_1)\frac{(t-t_1)^2}{2},\ c_1 \in [t_1, t].$$

Using $y'(t_1) = f(t_1, y(t_1))$ and $h = (t_2 - t_1)$ we get an approximation for y at t_2

$$y(t_2) = y(t_1) + hf(t_1, y(t_1)) + y''(c_1)\frac{h^2}{2}.$$

For h sufficiently small we write

$$y_2 = y_1 + hf(t_1, y_1)$$

to obtain an approximate value at t_2. Repeating, we have

$$t_{k+1} = t_k + h, \quad y_{k+1} = y_k + hf(t_k, y_k), \quad k = 1, \cdots (M+1) \text{ Euler's Method.}$$

Assuming that at each step we begin at exactly $y(t_k)$ (which won't be quite true), we define the local truncation error (LTE) to be the error obtained at a given step. In this case the LTE is

$$y''(c_k)\frac{h^2}{2}.$$

Summing the LTE's all the way to t_M we get, roughly anyway, the so-called Global Truncation Error (GTE). In this case we have

$$\sum_{k=1}^{M} y^{(2)}(c_k)\frac{h^2}{2} \approx y^{(2)}(c)Mh^2 = \frac{y^{(2)}(c)}{2}M\frac{(b-a)}{M}h = O(h).$$

Thus we see that the GTE, $E(h)$, for Euler's method is

$$E(h) \approx Ch.$$

From this we have

$$E\left(\frac{h}{2}\right) \approx C(h/2) = \frac{1}{2}Ch \approx \frac{1}{2}E(h).$$

So cutting the step size in half roughly reduces the GTE by a factor of 1/2.

Note that if the equation is $y' = f(t)$ with $y(a) = 0$ then this method give

$$y(b) \approx \sum_{k=1}^{M} f(t_k)h$$

which is a Riemann sum approximation to the integral $\displaystyle\int_a^b f(t)\,dt$.

Here is an Euler matlab code taken from John Mathews book. In Chapter 9 see the code euler.m. Also see the code `a9_1.m` which is a big start on one way to do Problem 1 in Assignment 5.

Modified Euler's Method:

Once again we consider (2.1) and apply the fundamental theorem of calculus to get

$$\int_{t_1}^{t_2} f(t, y(t))\,dt = \int_{t_1}^{t_2} y'(t)\,dt = y(t_2) - y(t_1).$$

This implies that

$$y(t_2) = y(t_1) + \int_{t_1}^{t_2} f(t, y(t))\,dt.$$

Now use the trapezoid rule to evaluate the integral with step size $h = t_2 - t_1$ to obtain

$$y(t_2) \approx y(t_1) + \frac{h}{2}[f(t_1, y(t_1)) + f(t_2, y(t_2))].$$

Unfortunately, we do not know $y(t_2)$ on the right side, so we use, for example, Euler's method to approximate it: $y(t_2) \approx y(t_1) + hf(t_1, y(t_1))$. We thus obtain a numerical procedure by denoting $y(t_k)$ by y_k for each k we set

$$\begin{aligned} p_k &= y_k + hf(t_k, y_k),\ t_{k+1} = t_k + h, \\ y_k + 1 &= y_k + \frac{h}{2}\left[f(t_k, y_k) + f(t_{k+1}, p_k)\right]. \end{aligned}$$

Note that if the equation were $y' = f(t)$ with $y(0) = 0$ then we would have

$$y(b) \approx \frac{h}{2}\sum_{k=1}^{M}[f(t_k) + f(t_{k+1})]$$

which is exactly the trapezoid rule for numerical quadrature.

In this case the LTE for the trap rule is

$$-y''(c_k)\frac{h^3}{12}$$

and the GTE is

$$-\sum_{k=1}^{M} y^{(2)}(c_k)\frac{h^3}{12} \approx -\frac{y^{(2)}(c)(b-a)}{12}h^2 = O(h^2).$$

Thus we see that the GTE $E(h)$ for Modified Euler Method is

$$E(h) \approx Ch^2.$$

From this we have

$$E\left(\frac{h}{2}\right) \approx C(h/2)^2 = \frac{1}{4}Ch^2 \approx \frac{1}{2}E(h).$$

So cutting the step size in half roughly reduces the GTE by a factor of 1/4.

Taylor's Method:

Assuming that y has derivatives of all order we could, again by Taylor's theorem, for any N write

$$y(t+h) = y(t) + y'(t)h + \frac{y^{(2)}(t)h^2}{2} + \cdots + \frac{y^{(N)}(t)h^N}{N!} \tag{2.2}$$

with a LTE of

$$\frac{y^{(N+1)}(c)h^{N+1}}{(N+1)!}.$$

We can adapt this result to each interval $[t_k, t_{k+1}]$ to obtain a numerical procedure

$$y_{k+1} = y_k + d_1 h + \frac{d_2 h^2}{2} + \frac{d_3 h^3}{6} + \cdots + \frac{d_N h^N}{N!}$$

where $d_j = y^{(j)}(t_k)$ for $j = 1, 2, \cdots, N$. In this case the LTE is $O(h^{N+1})$ and the GTE is $E_N(h) \approx Ch^N = O(h^N)$. Thus we have

$$E_N\left(\frac{h}{2}\right) = \frac{1}{2^N}E_N(h).$$

Computing the values d_k for a specific example at not to bad. Consider, for example, (see the exercises)

$$\begin{aligned} y' &= \frac{t-y}{2} \\ y^{(2)} &= \frac{2-t+y}{4} \\ y^{(3)} &= \frac{-2+t-y}{8} \\ y^{(4)} &= \frac{2-t+y}{16} \end{aligned}$$

from which we readily compute the values d_j.

There is a Taylor's matlab code named taylor.m taken from John Mathews book Chapter 9 .

Runge-Kutta Methods:

The dissadvantage of Taylor's method is obvious. We have to do some analytic calculations on every problem. To eliminate this difficulty and still have a higher order method we consider the one-step Runge-Kutta methods which are related to Taylor's method but do not require separate calculations.

We give a brief discussion of this method in the second order case. The fourth order case is more common but the algebra is very messy.

In the second order case the idea is to look for a formula

$$y_{k+1} = y_k + hF(t_k, y_k, h, f)$$

with F in the form

$$F(t,y,h,f) = \gamma_1 f(t,y) + \gamma_2 f(t+\alpha h, y+\beta h f(t,y).$$

where $\gamma_1, \gamma_2, \alpha, \beta$ are to be determined.

We apply Taylor's theorem in two variable (t, y) applied to the second term (the term containing γ_2) to obtain an approximation through the second derivative terms:

$$F(t,y,h,f) = \gamma_1 f(t,y) + \gamma_2 \Big\{ f(t,y) + h\big[\alpha f_t + \beta f f_y\big] + h^2\big[\frac{1}{2}\alpha^2 f_{tt} + \alpha\beta f_{ty} f + \frac{1}{2}\beta^2 f^2 f_{yy}\big] \Big\} + O(h^3)$$

where

$$f_t = \frac{\partial f}{\partial t},\ f_y = \frac{\partial f}{\partial y}, \text{etc.}$$

Similarly, we have

$$\begin{aligned} y' &= f(t,y) \\ y'' &= f_t + f_y y' = f_t + f_y f \\ y''' &= f_{tt} + 2f_{ty} f + f_{yy} f^2 + f_t f_y + f_y^2 f. \end{aligned}$$

Now if we expand $y(t)$ in a Taylor polynomial expansion about t_k we have

$$y(t_n+h) = y(t_n) + \frac{y'(t_n)}{1}h + \frac{y''(t_n)}{2}h^2 + \frac{y'''(t_n)}{6}h^3 + O(h^4).$$

Let us denote $y^{(k)}(t_n) = y_n^{(k)}$, then the LTE is

$$\begin{aligned} LTE =& y(t_{n+1}) - y(t_n) - hF(t_n, y(t_n), h; f) \\ =& hy_n^{(1)} + \frac{h^2}{2}y_n^{(2)} + \frac{h^3}{6}y_n^{(3)} + O(h^4) - hF(t_n, y(t_n), h; f) \\ =& h[1-\gamma_1-\gamma_2]f + h^2[(1/2-\gamma_2\alpha)f_t + (1/2-\gamma_2\beta)f_y f] \\ &+ h^3[(1/6 - 1/2\gamma_2\alpha^2)f_t t + (1/3-\gamma_2\alpha\beta)f_{ty}f + (1/6-1/2\gamma_2\beta^2)f_{yy}f^2 \\ &+ 1/6 f_y f_t + f_y^2 f] + O(h^4) \end{aligned}$$

For general f we cannot eliminate the third term but by choosing

$$\begin{aligned} \gamma_1 + \gamma_2 &= 1 \\ \gamma_2\alpha &= 1/2 \\ \gamma_2\beta &= 1/2 \end{aligned}$$

we can eliminate the first two terms. This implies that the LTE is $O(h^3)$. The system of equations is underdetermined but we can write

$$\gamma_1 = 1-\gamma_2, \quad \alpha = \beta = \frac{1}{2\gamma_2}$$

for arbitrary γ_2.

Special Cases:

For $\gamma_2 = 1/2$ we get the modified Euler's Method

$$y_{n+1} = y_n + \frac{h}{2}[f(t_n, y_n) + f(t_{n+1}, y_n + hf(t_n, y_n))].$$

For $\gamma_2 = 1$ we get the so-called Midpoint Method

$$y_{n+1} = y_n + \frac{h}{2}[f(t_n + h/2, y_n) + f(t_{n+1}, y_n + (h/2)f(t_n, y_n))].$$

We now consider choosing γ_2 in order to minimize the LTE at the $O(h^3)$ level. Note that

$$LTE = c(f, \gamma_2)h^3 + O(h^4)$$

where

$$c(f,\gamma_2) = \left[\left(\frac{1}{6} - \frac{1}{8\gamma_2}\right)f_{tt} + \left(\frac{1}{3} - \frac{1}{4\gamma_2}\right)f_{ty}f + \left(\frac{1}{6} - \frac{1}{8\gamma_2}\right)f_{yy}f^2 + \frac{1}{6}f_y f_t + \frac{1}{6}f_y^2 f \right].$$

Now applying the Cauchy-Schwartz inequality we get

$$|c(f,\gamma_2)| \le c_1(f)c_2(\gamma_2),$$

$$c_1(f) = \left[f_{tt}^2 + f_{ty}^2 f^2 + f_{yy}^2 f^4 + f_y^2 f_t^2 + f_y^4 f^2\right]^{1/2},$$

$$c_2(\gamma_2) = \left[\left(\frac{1}{6} - \frac{1}{8\gamma_2}\right)^2 + \left(\frac{1}{3} - \frac{1}{4\gamma_2}\right)^2 + \left(\frac{1}{6} - \frac{1}{8\gamma_2}\right)^2 + \frac{1}{18}\right]^{1/2}.$$

We can compute the minimum of $c_2(\cdot)$ to obtain

$$\gamma_2 = 3/4, \quad \text{with} \quad c_2(3/4) = 1/\sqrt{18}.$$

The resulting method is usually refered to as *Huen's Method* which takes the form

$$y_{k+1} = y_k + \frac{h}{4}\big[f(t_k,y_k) + 3f(t_k + \frac{2}{3}h, y_n + \frac{2}{3}hf(t_k,y_k))\big].$$

There is a Huen's matlab code named huen.m in Chapter 9 taken from John Mathews book.

This same analysis can be carried out for "higher order" methods but the algebra becomes very messy. One method that was very popular until about 1970 was the 4th order Runge-Kutta (RK4) method with a LTE of $O(h^5)$. If f does not depend on y this method reduces to simpson' rule for quadrature. The RK4 method goes like this

$$\begin{aligned}
f_1 &= f(t_k, y_k)\\
f_2 &= f(t_k + h/2, y_k + h/2 f_1)\\
f_3 &= f(t_k + h/2, y_k + h/2 f_2)\\
f_4 &= f(t_k + h, y_k + hf_3)\\
y_{k+1} &= y_k + \frac{h}{6}(f_1 + 2f_2 + 2f_3 + f_4).
\end{aligned}$$

There is a RK4 matlab code named rk4.m in Chapter 9 taken from John Mathews book.

Assignment 5:

1. Consider the problem $y' = (t-y)/2$, $y(0) = 1$ on $[0,3]$.

   a) Write a code using Euler's method to approximate the solution. Run it for $h = 1, 1/2, 1/4, \cdots, 1/64$ and test the formula for the effect on the GTE which should be $y_{M+1} - y(b)$ where you can compute the exact solution to find $y(3)$ (using $h = 1$ you should get a $C = .256$).

   b) Redo part a) using the modified euler's method, the midpoint method, Huen's method. Does Huen's method give the smallest GTE? Use the 4th order Runge-Kutta method and Taylor's method of orders 2,4 and 6. Compare all these with what you get using Matlab's builtin ode45 and ode23. You might also check "flops" and use "tic" and "toc" to time the methods.

2. Here are some other equations to try on these methods:

   a) $y' = t^2 - y$ on $[0,2]$ with $y(0) = 1$. The exact answer is $y = -e^{-t} + t^2 - 2t + 2$. Use $h = .2, .1, .05$.

   b) $y' = \dfrac{1}{1-t^2}$ on $[0,1]$ with $y(0) = 1$. Note that the exact answer doesn't exist on this interval. Try the methods anyway with $h = .1, .05, .025$.

   c) $y' = e^{-2t} - 2y$ on $[0,2]$ with $y(0) = 1/10$ and $h = .2, .1, .05$. Here $y = e^{-2t}/10 + te^{-2t}$.

3. Apply the midpoint method, the modified euler method and Huen's method for the following problem with the same h in each case.

$$y' = -y + t + 1, \quad 0 \le t \le 1, \quad y(0) = 1.$$

   What do you observe, i.e., how do they compare?

4. Solve the equation $y' = 1 + y^2$ using any method above and Taylor's method of order 2, 4 and 6 on $[0,5]$ with $y(0) = 1$ and compare your answers to $y = \tan(t - \pi/4)$.

5. Solve $y' = 2y - y^2$ on $[0,5]$ with $y(0) = 1$ and compare your answers to $y = 1 + \tanh(t)$.

6. Solve $y' = (1 - y^2)^{1/2}$ on $[0, 1.55]$ with $y(0) = 0$ and compare your answers to $y = \sin(t)$.

7. Solve $y' = y^2 \sin(t)$ on $[0, 1.55]$ with $y(0) = 1$ and compare your answers to $y = \sec(t)$.

References

1. Richard L. Burden, J. Douglas Faires and Albert C. Reynolds, "Numerical Analysis," Prindle, Weber and Schmidt, Boston, 1978.

2. G. Dahlquist and A. Bjorck, "Numerical Methods," Prentice-Hall Series in Automatic Computation, Englewood, NJ, 1974.

3. Kendall E. Atkinson, "An Introduction to Numerical Analysis," John Wiley and Sons, 1978.

4. John H. Mathews, "Numerical Methods: for computer science, engineering amd mathematics," 2nd Ed, 1992 Prentice Hall, Englewood Cliffs, New Jersey, 07632, U.S.A.

2.2 Multi-Step Methods for First Order Initial Value Problems

The methods described in the previous sections are called **one-step methods** since approximation at a point t_{k+1} only involves information from the one previous mesh point t_k. Since we actually have available to us the information at all the mesh points $\{t_j\}_{j=1}^{k}$ and because the error $|y(t_k) - y_k|$ tends to increase with k, it is reasonable to develop methods that take advantage of the extra known information. These methods, that we briefly discuss below are refered to as **multistep methods**.

Definition: *An m-multistep method for solving the initial value problem*

$$y' = f(t,y), \quad a \le x \le b, \quad y(a) = \alpha$$

is given by a formula of the following type:

$$\begin{aligned} w_{k+1} =& a_{m-1}w_k + a_{m-2}w_{k-1} + \cdots + a_0 w_{k+1-m} \\ &+ h[b_m f(t_{k+1}, w_{k+1}) + b_{m-1} f(t_k, w_k) \\ &+ \cdots + b_0 f(t_{k+1-m}, w_{k+1-m})] \end{aligned}$$

for $k = (m-1), m, \cdots, (N-1)$ where the starting values

$$w_0 = \alpha_0, \; w_1 = \alpha_1, w_2 = \alpha_2, \cdots, w_{m-1} = \alpha_{m-1},$$

are specified and $h = (b-a)/N$.

When $b_m = 0$ the method is called an ***explicit*** *method which simply means that w_{k+1} is given explicitly interm of previously defined terms. When $b_m \neq 0$, this method is called* ***implicit*** *since w_{k+1} occurs on both sides of the equation and is thus only determined implicitly.*

These methods typically require that $\alpha_0 = \alpha$ and then the remaining values of α_j are determined using a Runge-Kutta (or some other one-step method).

One procedure often used to derive multi-step methods is to first notice that

$$y(t_{k+1}) = y(t_k) + \int_{t_k}^{t_{k+1}} f(t, y(t))\, dt.$$

Since we cannot integrate $f(t, y(t))$ without knowing $y(t)$ we instead integrate an interpolating polynomial, P, which is determined by the previous data $(t_0, y_0), \cdots, (t_k, y_k)$. When we assume in addition that $y(t_j) = y_j$ to get

$$y(t_{k+1}) \approx y_k + \int_{t_k}^{t_{k+1}} P(t)\, dt.$$

The polynomial most often used is called the Newton-backward-interpolation polynomial. Another polynomial which is easy to describe is the Lagrange interpolation polynomial. The first degree Lagrange interpolation polynomial is

$$\begin{aligned} P(t) &= \frac{(t-t_k)}{(t_{k-1}-t_k)} f(t_{k-1}, y(t_{k-1})) + \frac{(t-t_{k-1})}{(t_k - t_{k-1})} f(t_k, y(t_k)) \\ &= \frac{f(t_{k-1}, y(t_{k-1}))}{-h}(t-t_k) + \frac{f(t_k, y(t_k))}{h}(t - t_{k-1}) \end{aligned}$$

Some common explicit multi-step methods:

Adams-Bashforth Three-Step Method:

$$
\begin{aligned}
&y_0 = \alpha,\ y_1 = \alpha_1,\ y_2 = \alpha_2, \\
&y_{k+1} = y_k + \frac{h}{12}\big[23f(t_k, y_k) - 16f(t_{k-1}, y_{k-1}) + 5f(t_{k-2}, y_{k-2})\big]
\end{aligned}
$$

for $k = 2, 3, \cdots, (N-1)$ with a LTE of $3y^{(4)}(c_k)h^3/8$.

Adams-Bashforth Four-Step Method:

$$
\begin{aligned}
&y_0 = \alpha,\ y_1 = \alpha_1,\ y_2 = \alpha_2, y_3 = \alpha_3 \\
&y_{k+1} = y_k + \frac{h}{24}\big[55f(t_k, y_k) - 59f(t_{k-1}, y_{k-1}) + 37f(t_{k-2}, y_{k-2} - 9f(t_{k-3}, y_{k-3})\big]
\end{aligned}
$$

for $k = 2, 3, \cdots, (N-1)$ with a LTE of $251y^{(6)}(c_k)h^4/720$.

There is a sample matlab program for the Adams-Bashforth-Moulton method named abm.m in Chapter 9 taken from John Mathews book.

Milne's Method:

This explicit method comes from integrating the Newton backward polynomial over $[t_{k-3}, t_{k+1}]$. Once again there is a sample matlab program for the Milne-Simpson method named milne.m in Chapter 9 taken from John Mathews book.

In practice, implicit multi-step methods are not usually used alone; rather, they are used to improve approximations obtained by explicit methods. The usual procedure, involving a combination of an explicit and implicit method, is called a **Predictor-corrector method**. We comment that it is often more efficient to simply reduce the step size if more accuracy is required.

2.3 Higher Order Equations and First Order Systems

An m-th order system of first order initial value problems can be written as:

$$
\begin{aligned}
&\frac{du_1}{dt} = f_1(t, u_1, u_2, \cdots, u_m), \\
&\frac{du_2}{dt} = f_2(t, u_1, u_2, \cdots, u_m), \\
&\qquad\qquad \vdots \\
&\frac{du_m}{dt} = f_m(t, u_1, u_2, \cdots, u_m), \\
&\text{for}\ \ a \le t \le b,\ \text{with inital conditions} \\
&u_1(a) = \alpha_1 \\
&u_2(a) = \alpha_2 \\
&\quad \vdots \\
&u_m(a) = \alpha_m
\end{aligned}
$$

Also given a higher initial value problem

$$y^{(m)}(t) = f(t, y, y', y'', \cdots, y^{(m-1)}),\ a \le t \le b,$$

with

$$y(a) = \alpha_1,\ y'(a) = \alpha_2,\ \cdots, y^{(m-1)}(a) = \alpha_m$$

we can let

$$u_1 = y,\ u_2 = y',\ \cdots, u_m = y^{(m-1)},$$

and write as a first order system

$$\begin{aligned}
&\frac{du_1}{dt} = \frac{dy}{dt} = u_2,\\
&\frac{du_2}{dt} = \frac{dy'}{dt} = u_3,\\
&\qquad\quad\vdots\\
&\frac{du_m}{dt} = \frac{dy^{(m-1)}}{dt} = y^{(m)} = f(t, y, y', \cdots, y^{(m-1)})\\
&\qquad\qquad\qquad\qquad\quad = f(t, u_1, u_2, \cdots, u_m)\\
&\text{for } \ a \le t \le b, \text{ with inital conditions}\\
&u_1(a) = y(a) = \alpha_1\\
&u_2(a) = y'(a) = \alpha_2\\
&\qquad\vdots\\
&u_m(a) = y^{(m-1)}(a) = \alpha_m
\end{aligned}$$

Methods to solve systems of first order initial value problems are simply generalizations of the methods used for first-order equations. For example the Runge-Kutta method can be described as follows:

Partition the interval $[a, b]$ into N subitervals with mesh points

$$t_j = a + jh,\ j = 0, 1, \cdots, N.$$

Now use w_{ij} to denote the approximation to $u_i(t_j)$ for each $j = 0, 1, \cdots, N$ and $i = 1, 2, \cdots, m$.

For the initial conditions we set

$$\begin{aligned}
w_{1,0} &= \alpha_1\\
w_{2,0} &= \alpha_2\\
&\vdots\\
w_{m,0} &= \alpha_m
\end{aligned}$$

If we assume the values $w_{1,j},\ w_{2,j},\ \cdots,\ w_{m,j}$ have been computed then we compute $w_{1,j+1}$, $w_{2,j+1}, \cdots, w_{m,j+1}$ by first calculating

$$k_{1,i} = hf_i(t_j, w_{1,j}, w_{2,j}, \cdots, w_{m,j}) \text{ for } i = 1, \cdots, m$$
$$k_{2,i} = hf_i(t_j + \frac{h}{2}, w_{1,j} + \frac{1}{2}k_{1,1}, w_{2,j}\frac{1}{2}k_{1,2}, \cdots, w_{m,j}\frac{1}{2}k_{1,m}) \text{ for } i = 1, \cdots, m$$
$$k_{3,i} = hf_i(t_j + \frac{h}{2}, w_{1,j} + \frac{1}{2}k_{2,1}, w_{2,j}\frac{1}{2}k_{2,2}, \cdots, w_{m,j}\frac{1}{2}k_{2,m}) \text{ for } i = 1, \cdots, m$$
$$k_{4,i} = hf_i(t_j + h, w_{1,j} + k_{3,1}, w_{2,j} + k_{3,2}, \cdots, w_{m,j} + k_{3,m}) \text{ for } i = 1, \cdots, m$$

and then

$$w_{i,j+1} = w_{i,j} + \frac{1}{6}\big[k_{1,i} + 2k_{2,i} + 2k_{3,i} + k_{4,i}\big], \text{ for } i = 1, \cdots, m$$

We note that multistep and predictor-corrector methods can also be extended to first order systems. At this stage of the game I would assume the powers that be have done a good job and simple use a Runge-Kutta solver that has variable mesh sizes and the whole works builtin as in matlab ode45.

There is a new ode solver in Matlab 5 which is a much better code then the older version with the same name. The following is the help file for this code. One major advantage to this code is that you can input a vector "tspan" whose first coordinate is the initial time, whose final coordinate is the final time and the solution is given at exactly the values in "tspan."

Assignment 6: Use the matlab ode45 or the ode45 in the ode suite to solve the systems and compare answers with exact solutions:

1. For $0 \le t \le 1$, $\begin{array}{ll} u_1' = 3u_1 + 2u_2, & u_1(0) = 0 \\ u_2' = 4u_1 + u_2, & u_2(0) = 1 \end{array}$.
   Exact answer $u_1 = (1/3)(\exp(5t) - \exp(-t)),\ u_2 = (1/3)(\exp(5t) + 2\exp(-t))$.
2. For $1 \le t \le 2$, $t^2y'' - 2ty' + 2y = t^3\ln(t),\ \ y(1) = 1,\ y'(1) = 0$.
   Exact answer $y = (7/4)t + (t^3/2)\ln(t) - (3/4)t^3$.
3. For $0 \le t \le 5$, $\begin{array}{rl} u_1' + 4u_1 + 2u_2 = \cos(t) + 4\sin(t), & u_1(0) = 0 \\ u_2' - 3u_1 - u_2 = -3\sin(t), & u_2(0) = -1 \end{array}$.
   Exact answer $u_1 = 2\exp(-t) - 2\exp(-2t) + \sin(t),\ u_2 = -3\exp(-t) + 2\exp(-2t)$.
4. For $0 \le t \le 1$, $\begin{array}{rl} u_1' = u_2, & u_1(0) = 3 \\ u_2' = -u_1 + 2e^{-t} + 1, & u_2(0) = 0 \end{array}$
   Exact answer $u_1 = \cos(t) + \sin(t) + e^{-t} + 1,\ u_2 = \cos(t) - \sin(t) - e^{-t}$.
5. For $1 \le t \le 1.9$ solve $y'' = 2y^3,\ y(1) = -1,\ y'(1) = -1$ and compare with $y = (t-2)^{-1}$.

The following program is an m-file that runs a presentation for solving higher order equations using Runge-Kutta. The program calls John Mathews Runge-Kutta "rks4.m" code for solving higher order equations using Runge-Kutta. It is taken from John Mathews book. Just run the code `a9_s.m` which is an m-file that contains a presentation for solving systems using Runge-Kutta.

2.4 Methods for Second Order Boundary Value Problems

In this section we are primarily concerned with second order boundary value problems (BVPs) of the form:

$$\begin{aligned}
&y'' = f(x, y, y'),\ a \le x \le b \\
&a_0 y(a) - a_1 y'(a) = 0, \\
&b_0 y(b) + b_1 y'(b) = 0
\end{aligned}$$

Problems of this type are very different from IVPs and as you will see require very different methods for solution. On the other hand, the first method we consider actually uses the solutions of IVPs to obtain the solution of the boundary value problem. We note that the methods described below are applicable to higher order BVPs.

Linear Shooting Method for Higher Order Systems:

Considering the discussion concerning reduction of higher order equations to first order systems given in the last subsection, I have decided to present a method for first order systems that will include mth order BVPs as a special case.

Thus, as a special case, we might have a problem like:

$$y^{(m)}(t) = f(t, y, y', y'', \cdots, y^{(m-1)}),\ a \le t \le b,$$

with

$$\sum_{j=1}^{m} a_{i,j} y^{(j-1)}(a) + \sum_{j=1}^{m} b_{i,j} y^{(j-1)}(b) = \beta_i, \quad i = 1, 2, \cdots, m.$$

The Important Linear Case:

$$f(t, y, y', y'', \cdots, y^{(m-1)}) = \sum_{j=0}^{m-1} a_j y^{(j)} + g(t).$$

We can let

$$u_1 = y,\ u_2 = y',\ \cdots, u_m = y^{(m-1)},$$

$$\mathcal{U} = \begin{bmatrix} u_1 \\ u_2 \\ \vdots \\ u_m \end{bmatrix},$$

$$A = \begin{bmatrix} 0 & 1 & 0 & 0 & \cdots & 0 \\ 0 & 0 & 1 & 0 & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & 0 & 1 \\ a_0 & a_1 & a_2 & a_3 & \cdots & a_{m-1} \end{bmatrix}, \quad \mathcal{G} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ \vdots \\ 0 \\ g(t) \end{bmatrix}$$

and we obtain the first order system

$$\begin{aligned}&\frac{d\mathcal{U}}{dt} = A\mathcal{U} + \mathcal{G}\\ &\text{for } \ a \le t \le b, \ \text{with boundary conditions}\\ &B_a\mathcal{U}(a) + B_b\mathcal{U}(b) = \mathcal{B}.\end{aligned}$$

Here B_a and B_b are the $m \times m$ matrices

$$B_a = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1m} \\ a_{21} & a_{22} & \cdots & a_{2m} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m1} & a_{m2} & \cdots & a_{mm} \end{bmatrix}, \ B_b = \begin{bmatrix} b_{11} & b_{12} & \cdots & b_{1m} \\ b_{21} & b_{22} & \cdots & b_{2m} \\ \vdots & \vdots & \ddots & \vdots \\ b_{m1} & b_{m2} & \cdots & b_{mm} \end{bmatrix} \text{ and } \mathcal{B} = \begin{bmatrix} \beta_1 \\ \beta_2 \\ \vdots \\ \beta_m \end{bmatrix}.$$

Rather than restrict ourselves to this special case we will consider a general $m \times m$ matrix valued function $A(t)$ and general forcing term $\mathcal{G}(t)$.

The Linear shooting method consists of finding a solution in the form

$$\begin{aligned}\mathcal{U}(t) &= \mathcal{U}_0(t) + \sum_{j=1}^{m} \xi_j \mathcal{U}_j(t)\\ &= \mathcal{U}_0(t) + \Phi(t)\xi\end{aligned}$$

where $\Phi(t) = \begin{bmatrix}\mathcal{U}_1(t) & \mathcal{U}_2(t) & \cdots & \mathcal{U}_m(t)\end{bmatrix}$ and $\xi = \begin{bmatrix} \xi_1 \\ \vdots \\ \xi_m \end{bmatrix}$.

The functions $\{\mathcal{U}_j(t)\}_{j=0}^m$ are computed as the solutions to the following $(m+1)$ IVPs:

$$\begin{aligned}&\frac{d\mathcal{U}_0}{dt} = A\mathcal{U}_0 + \mathcal{G}, \quad \mathcal{U}_0(0) = \mathcal{U}_0 \ \text{ any initial vector}\\ &\frac{d\mathcal{U}_j}{dt} = A\mathcal{U}_j, \quad \mathcal{U}_j(0) = e_j, \quad j = 1, 2, \cdots, m\end{aligned}$$

where $[e_1, e_2, \cdots, e_m]$ is the $m \times m$ identity matrix. We compute the vector ξ to satisfy the boundary condition from

$$Q\xi = \mathcal{B} - B_a\mathcal{U}_0(a) - B_b\mathcal{U}_0(b), \ \text{where } \ Q = B_a + B_b\Phi(b).$$

For these initial value problems I would certainly use the ode45 from the matlab suite so that all the solutions $\mathcal{U}_j$ will be defined on the same grid points. Otherwise you would need to use a spline to make this so.

Second Order Linear Case:

Consider the linear second order case with separated boundary conditions:

$$\begin{aligned}&y'' = py' + qy + r, \quad a \le x \le b\\ &a_0 y(a) - a_1 y'(a) = 0,\\ &b_0 y(b) + b_1 y'(b) = 0.\end{aligned}$$

In this case we have

$$A = \begin{bmatrix} 0 & 1 \\ q & p \end{bmatrix}, \quad \mathcal{G} = \begin{bmatrix} 0 \\ r \end{bmatrix}, \quad \mathcal{U} = \begin{bmatrix} u_1 \\ u_2 \\ \vdots \\ u_m \end{bmatrix}, \quad B_a = \begin{bmatrix} a_0 & -a_1 \\ 0 & 0 \end{bmatrix}, \quad B_b = \begin{bmatrix} 0 & 0 \\ b_0 & b_1 \end{bmatrix}.$$

In this case we have three IVPs

$$\begin{aligned} \frac{d\mathcal{U}_0}{dt} &= A\mathcal{U}_0 + \mathcal{G}, \quad \mathcal{U}_0(0) = \begin{bmatrix} \eta_1 \\ \eta_2 \end{bmatrix} \\ \frac{d\mathcal{U}_1}{dt} &= A\mathcal{U}_1, \;\; \mathcal{U}_1(0) = \begin{bmatrix} 1 \\ 0 \end{bmatrix} \\ \frac{d\mathcal{U}_2}{dt} &= A\mathcal{U}_2, \;\; \mathcal{U}_2(0) = \begin{bmatrix} 0 \\ 1 \end{bmatrix} \end{aligned}$$

We solve these IVPs , then define

$$\Phi(t) = \begin{bmatrix} \mathcal{U}_1(t) & \mathcal{U}_2(t) \end{bmatrix}, \quad Q = B_a + B_b\Phi(b), \quad \xi = Q^{-1}\big[\mathcal{B} - B_a\mathcal{U}_0(a) - B_b\mathcal{U}_0(b)\big]$$

and we finally obtain

$$\mathcal{U}(t) = \mathcal{U}_0(t) + \Phi(t)\xi.$$

Nonlinear Shooting Method for Higher Order Systems:

We now turn to the nonlinear case retaining some of notation from the linear case. We consider systems

$$\begin{aligned} Y'(t) &= F(t, Y), \;\; a \le t \le b, \\ \mathcal{B}Y &\equiv B_aY(a) + B_bY(b) = 0 \end{aligned}$$

where Y is an m vector and the boundary conditions are the same as above.

In this case we again consider an initial value problem

$$\begin{aligned} U' &= F(t, U), \\ U(a) &= S. \end{aligned}$$

Then, denoting the solution to this initial value problem by $U \equiv U(t; S)$ we seek S so that

$$\Psi(S) \equiv B_aS + B_bU(b, S) = 0.$$

The problem is then reduced to finding S^* such that $\Psi(S^*) = 0$.

This is usually done numerically using a Newton method which we now describe.

For a vector Z, let

$$A(t, Z) \equiv \frac{\partial F}{\partial Z}(t, Z(t)).$$

Then consider the m linearized initial value problems

$$\Phi' = A(t,U)\Phi, \quad \Phi(a;S) = I.$$

So starting with an S^0 we solve the $(m+1)$ coupled IVPs and then get a sequence of new problems from a Newton scheme where S^{j+1} is obtained from

$$Q(S^j)[S^{j+1} - S^j] = -\Psi(S^j), \quad \Psi(S) \equiv B_a S^j + B_b U(b, S^j) = 0, \quad Q(S^j) = B_a + B_b \Phi(b, S^j).$$

In slightly different terminology lets consider a special case:

$$\begin{aligned} &y'' = f(x,y,y'). \quad a \le x \le b \\ &a_0 y(a) - a_1 y'(a) = \gamma_1 \\ &b_0 y(b) + b_1 y'(b) = \gamma_2 \end{aligned}$$

Now consider the associated IVP

$$\begin{aligned} &z'' = f(x,z,z'). \quad a \le x \le b \\ &z(a,s) = a_1 s - c_1 a_0 \gamma_1 \\ &z'(a,s) = a_0 s - c_0 \gamma_1 \end{aligned}$$

where we choose c_0 and c_1 so that $c_0 a_1 - c_1 a_0 = 1$ which implies

$$a_0 z(a,s) - a_1 z'(a,s) = \gamma_1.$$

Now we need to choose s so that

$$\phi(s) \equiv b_0 z(b,s) + b_1 z'(b,s) = \gamma_2.$$

Since $\phi(s) = 0$ is a nonlinear equation we proceed using a Newton method. In particular, we define interates

$$s_{m+1} = s_m + \phi(s_m) \left(\left. \frac{d\phi}{ds} \right|_{s=s_j} \right)^{-1}, \quad m = 0, 1, \cdots .$$

The problem is that we really don't have ϕ directly so it is not easy to compute its derivative.

We proceed as follows: Let $\xi_s(x) = \dfrac{\partial z(x,s)}{\partial s}$. Then by the definition of ϕ we see that

$$\phi'(s) = b_0 \xi_s(b) + b_1 \xi_s'(b).$$

Now differentiate the equation for z, i.e.,

$$z''(x,s) = f(x, z(x,s), z'(x,s))$$

with respect to s to get

$$\xi_s(x)'' = f_2(x, z(x,s), z'(x,s)))\xi_s(x) + f_3(x, z(x,s), z'(x,s)))\xi_s'(x),$$

also differential the boundary conditions

$$z(a,s) = a_1 s - c_1 a_0 \gamma_1, \quad z'(a,s) = a_0 s - c_0 \gamma_1$$

with respect to s to get

$$\xi_s(a) = a_1, \ xi_s'(a) = a_0.$$

Thus we have the system

$$\begin{aligned}
&\xi_s(x)'' = f_2(x, z(x,s), z'(x,s))\xi_s(x) + f_3(x, z(x,s), z'(x,s))\xi_s'(x),\\
&z'' = f(x, z, z').\\
&\xi_s(a) = a_1,\\
&\xi_s'(a) = a_0\\
&z(a,s) = a_1 s - c_1 a_0 \gamma_1\\
&z'(a,s) = a_0 s - c_0 \gamma_1.
\end{aligned}$$

Now convert these second order equations to a first order system for the variables z, z', ξ_s, ξ_s' to get the necessary value of ϕ and ϕ' for updating s in the Newton's method.

An Example:

$$\begin{aligned}
&y'' = -y + \frac{2(y')^2}{y}, \quad -1 \le x \le 1\\
&y(-1) = y(1) = \left(e + e^{-1}\right)^{-1}
\end{aligned}$$

The exact solution is $y = \left(e^x + e^{-x}\right)^{-1}$.

The initial value problem for $z(x,s)$ is

$$\begin{aligned}
&z'' = -z + \frac{2(z')^2}{z}, \quad -1 \le x \le 1\\
&z(-1,s) = \left(e + e^{-1}\right)^{-1},\\
&z'(-1,s) = s
\end{aligned}$$

and the equation for ξ_s is

$$\begin{aligned}
&\xi_s'' = \left[-1 - 2\left(\frac{z'}{z}\right)^2 \right]\xi_s + 4\frac{z'}{z}\xi_s',\\
&\xi_s(-1) = 0\\
&\xi_s'(-1) = 1
\end{aligned}$$

The function $\phi(s)$ is
$$\phi(s) = z(1,s) - \left(e + e^{-1}\right)^{-1}$$
and
$$\phi'(s) = \xi_s(1).$$

In this case the zero of ϕ is $s^* = .245777174$.

Assignment 7: Use the matlab ode45 solver to solve the BVPs and compare answers with exact solutions:

1. $y'' = -4y'/x + 2y/x^2 - 2\ln(x)/x^2,\ \ 1 \le x \le 2,\ \ y(1) = -1/2,\ \ y(2) = \ln(2).$
   Can you find the exact answer using Maple or mathematica or some other way?

2. $y'' = -2y'/x + 2y/x^2 + \sin(\ln(x))/x^2,\ \ 1 \le x \le 2,\ \ y(1) = 1,\ \ y(2) = 2.$
   Can you find the exact answer using Maple or mathematica or some other way?

3. $y'' = -3y' + 2y + 2x + 3,\ \ 0 \le x \le 1,\ \ y(0) = 2,\ \ y(1) = 1.$
   Can you find the exact answer using Maple or mathematica?

4. $y'' = y^3/2,\ \ 1 \le x \le 2,\ \ y(1) = -2/3,\ \ y(2) = -1.$
   Exact answer $y = 2/(x-4)$.

5. $y'' = y^3 - yy',\ \ 1 \le x \le 2,\ \ y(1) = 1/2,\ \ y(2) = 1/3.$
   Exact answer $y = 1/(x+1)$.

6. $y'' = -2yy'/x,\ \ 1 \le x \le 2,\ \ y(1) = 1/2,\ \ y(2) = 2/3.$
   Exact answer $y = x/(x+1)$.

7. $y'' = y^3 - yy',\ \ 1 \le x \le 2,\ \ y(1) = 1/2,\ \ y(2) = 1/3.$
   Exact answer $y = 1/(x+1)$.

Once again there is an m-file presentation named `a9_9.m` for the linear shooting method taken from Chpater 9 of John Mathews book.

Finite Difference Method for BVPs:
First in this section we will restrict to the following special case for a linear second order boundary value problem.

$$\begin{aligned} &y'' = py' + qy + r, \quad a \le x \le b \\ &y(a) = \alpha, \\ &y(b) = \beta. \end{aligned}$$

Later we will discuss more general Sturm-Louiville BVPs.

The basis for the finite difference method goes back to our section on Euler's method. Namely we use difference quotients to approximate the derivatives. Again the particular difference quotient is chosen so that a certain order truncation error is achieved. Notice also in this case we have to approximate a second order derivative. The starting point, in this case, is to divide the interval into $(N+1)$ equal subintervals, whose endpoints are the mesh points $x_i = a+ih$, for $i = 0, 1, 2, \cdots, (N+1)$ where $h = (b-a)/(N+1)$. At the *interior* meshpoints x_i, $i = 1, 2, \cdots, N$ the differential equation to be approximated is

$$y''(x_i) = p(x_i)y'(x_i) + q(x_i)y(x_i) + r(x_i).$$

Expanding y in a third-degree Taylor polynomial about the point x_i evaluated at x_{i-1} and x_{i+1} (here we are assuming $y \in C^4[x_{i-1}, x_{i+1}]$, we have (note that $x_{i+1} = x_i + h$ and $x_{i-1} = x_i - h$)

$$y(x_{i+1}) = y(x_i) + hy'(x_i) + \frac{h^2}{2}y''(x_i) + \frac{h^3}{6}y'''(x_i) + \frac{h^4}{24}y^{(4)}(c_i^+)$$

$$y(x_{i-1}) = y(x_i) - hy'(x_i) + \frac{h^2}{2}y''(x_i) - \frac{h^3}{6}y'''(x_i) + \frac{h^4}{24}y^{(4)}(c_i^-)$$

where $c_i^+ \in [x_i, x_{i+1}]$ and $c_i^- \in [x_{i-1}, x_i]$. Adding these equations together, the terms involving $y'(x_i)$ and $y'''(x_i)$ cancel and after solving for $y''(x_i)$, we arrive at

$$y''(x_i) = \frac{1}{h^2}\big[y(x_{i+1}) - 2y(x_i) + y(x_{i-1})\big] - \frac{h^2}{24}\big[y^{(4)}(c_i^+) + y^{(4)}(c_i^-)\big].$$

This can be slightly more simplified by noting that under our assumptions $y^{(4)}$ is continuous and the point $(y^{(4)}(c_i^+) + y^{(4)}(c_i^-))/2$ lies between $y^{(4)}(c_i^+)$ and $y^{(4)}(c_i^-)$ so by the intermediate value theorem, there is a point

$$c_i \in [c_i^-, c_i^+] \subset [x_{i-1}, x_{i+1}]$$

such that

$$\frac{(y^{(4)}(c_i^+) + y^{(4)}(c_i^-))}{2} = y^{(4)}(c_i).$$

With this we arrive at the desired **centered difference formula**

$$y''(x_i) = \frac{1}{h^2}\big[y(x_{i+1}) - 2y(x_i) + y(x_{i-1})\big] - \frac{h^2}{12}\big[y^{(4)}(c_i)\big].$$

A centered-difference formula for $y'(x_i)$ is also obtained in a similar manner. Namely we n start with a quadratic Taylor polynomial expansion to get

$$y(x_{i+1}) = y(x_i) + hy'(x_i) + \frac{h^2}{2}y''(x_i) + \frac{h^3}{6}y'''(d_i^+)$$

and

$$y(x_{i-1}) = y(x_i) - hy'(x_i) + \frac{h^2}{2}y''(x_i) - \frac{h^3}{6}y'''(d_i^-)$$

and this time we subtract the equations and solve for $y'(x_i)$.

$$y'(x_i) = \frac{1}{2h}\big[y(x_{i+1}) - y(x_{i-1})\big] - \frac{h^2}{6}y'''(d_i)$$

where again we have employed the intermediate value theorem applied to y''' to get a point

$$d_i \in [d_i^-, d_i^+] \subset [x_{i-1}, x_{i+1}].$$

Using these centered difference formulas we have

$$\begin{aligned}\frac{y(x_{i+1}) - 2y(x_i) + y(x_{i-1})}{h^2} &= p(x_i)\left[\frac{y(x_{i+1}) - y(x_{i-1})}{2h}\right] + q(x_i)y(x_i) + r(x_i) \\ &\quad - \frac{h^2}{12}\left[2p(x_i)y'''(d_i) - y^{(4)}(c_i)\right].\end{aligned}$$

A finite-difference method with LTE of order $O(h^2)$ results by omitting the last term and using the boundary conditions $y(a) = \alpha$, $y(b) = \beta$. Namely, we define approximates $w_i \approx y(x_i)$ as follows:

$$\begin{aligned}&w_0 = \alpha, \quad w_{N+1} = \beta \\ &\left[\frac{-w_{i+1} + 2w_i - w_{i-1}}{h^2}\right] + p(x_i)\left[\frac{w_{i+1} - w_{i-1}}{2h}\right] + q(x_i)w_i = -r(x_i), \\ &\text{for} \quad i = 1, 2, \cdots, N\end{aligned}$$

This system of equations can be written in a *tridiagonal form* as

$$-\big(1 + \frac{h}{2}p(x_i)\big)w_{i-1} + (2 + h^2q(x_i))w_i - \big(1 - \frac{h}{2}p(x_i)\big)w_{i+1} = -h^2r(x_i).$$

To make this even more clear we write the system in matrix form: Let

$$A = \begin{bmatrix} 2 + h^2q(x_1) & -1 + \frac{h}{2}p(x_1) & 0 & 0 & \cdots & 0 \\ -1 - \frac{h}{2}p(x_2) & 2 + h^2q(x_2) & -1 + \frac{h}{2}p(x_2) & 0 & \cdots & 0 \\ 0 & -1 - \frac{h}{2}p(x_3) & 2 + h^2q(x_3) & -1 + \frac{h}{2}p(x_3) & & 0 \\ \vdots & \ddots & \ddots & \ddots & & \vdots \\ 0 & 0 & \cdots & 0 & -1 - \frac{h}{2}p(x_N) & 2 + h^2q(x_N) \end{bmatrix}$$

$$W = \begin{bmatrix} w_1 \\ w_2 \\ \vdots \\ w_N \end{bmatrix}, \quad B = \begin{bmatrix} -h^2r(x_1) + \big(1 + \frac{h}{2}p(x_1)\big)\alpha \\ -h^2r(x_2) \\ \vdots \\ -h^2r(x_{N-1}) \\ -h^2r(x_N) + \big(1 - \frac{h}{2}p(x_N)\big)\beta \end{bmatrix}.$$

Thus we obtain the system

$$AW = B.$$

Since the necessity to solve tri-diagonal systems occurs many times (even in the later finite element method using linear splines) it is probably worthwhile to spend a little time talking about solving such systems. Below I have included a matlab code from John Mathews book for carrying out one such procedure. The method I will now present is slightly different in presentation but amounts to the same thing.

Suppose we have and $n \times n$ matrix

$$A = \begin{bmatrix} a_1 & c_1 & 0 & 0 & \cdots & 0 \\ b_2 & a_2 & c_2 & 0 & \cdots & 0 \\ 0 & b_3 & a_3 & c_3 & & 0 \\ & & & & & \\ \vdots & & \ddots & \ddots & & \vdots \\ 0 & 0 & \cdots & 0 & b_n & a_n \end{bmatrix}.$$

Then the matrices L and U in the LU decomposition $A = LU$ can be obtained in the form

$$L = \begin{bmatrix} \alpha_1 & 0 & 0 & 0 & \cdots & 0 \\ b_2 & \alpha_2 & 0 & 0 & \cdots & 0 \\ 0 & b_3 & \alpha_3 & 0 & & 0 \\ \vdots & & \ddots & \ddots & \ddots & \vdots \\ & & & & & \\ 0 & 0 & \cdots & 0 & b_n & \alpha_n \end{bmatrix}.$$

$$U = \begin{bmatrix} 1 & \gamma_1 & 0 & 0 & \cdots & 0 \\ 0 & 1 & \gamma_2 & 0 & \cdots & 0 \\ & & & & & \\ \vdots & & \ddots & \ddots & \ddots & \vdots \\ 0 & 0 & \cdots & 0 & 1 & \gamma_{n-1} \\ 0 & 0 & \cdots & 0 & 0 & 1 \end{bmatrix}.$$

Multiply L time U and equating to A we can recursively find $\{\alpha_j\}$ and $\{\beta_j\}$. We obtain

$$\begin{aligned} &a_1 = \alpha_1, \quad \alpha_1\gamma_1 = c_1 \\ &a_i = \alpha_i + b_i\gamma_{i-1}, \quad i = 2, \cdots, n \\ &\alpha_i\gamma_i = c_i, \quad i = 2, \cdots, (n-1). \end{aligned}$$

From this we immediately obtain

$$\begin{aligned} &\alpha_1 = a_1, \quad \gamma_1 = \frac{c_1}{\alpha_1} \\ &\alpha_i = a_i - b_i\gamma_{i-1}, \quad \gamma_i = \frac{c_i}{\alpha_i}, \ i = 2, \cdots, (n-1), \\ &\alpha_n = a_n - b_n\gamma_{n-1}. \end{aligned}$$

Now to solve $LUx = f$, let $Ux = z$ and $Lz = f$. Then

$$\begin{aligned}
&z_1 = \frac{f_1}{a_1}, \quad z_i = \frac{f_i - b_i z_{i-1}}{\alpha_i}, \quad i = 2, \cdots, n \\
&x_n = z_n, \quad x_i = z_i - \gamma_i x_{i+1}, \quad i = (n-1), (n-2), \cdots, 1.
\end{aligned}$$

Assignment 8: Use the finite difference method to solve the BVPs and compare answers with exact solutions (see exercises in assignment 7):

1. $y'' = -4y'/x + 2y/x^2 - 2\ln(x)/x^2, \quad 1 \le x \le 2, \quad y(1) = -1/2, \quad y(2) = \ln(2)$.
2. $y'' = -2y'/x + 2y/x^2 + \sin(\ln(x))/x^2, \quad 1 \le x \le 2, \quad y(1) = 1, \quad y(2) = 2$.
3. $y'' = -3y' + 2y + 2x + 3, \quad 0 \le x \le 1, \quad y(0) = 2, \quad y(1) = 1$.
4. $y'' = 100y, \quad 0 \le x \le 1, \quad y(0) = 1, \quad y(1) = e^{-10}$. Note the exact answer is $y = e^{-10x}$.

Tere is another presentation named `a9_10.m` from John Mathews Book for solving a second order BVP using the finite difference method.

There is file named findiff.m is a John Mathews code for the finite difference method which is used in the previous file `a9_10.m`.

The is also a code named trisys.m from John Mathews for the solving tri-diagonal systems used in these files.

More General Boundary Conditions:

Finally in this section we will consider the more general Sturm-Louiville boundary value problem

$$\begin{aligned}
&y'' = py' + qy + r, \quad a \le x \le b \\
&a_0 y(a) - a_1 y'(a) = \alpha, \\
&b_0 y(b) + b_1 y'(b) = \beta.
\end{aligned}$$

Now we recall that the finite difference method developed above is an $O(h^2)$ method. We want to maintain this order of approximation. But since the boundary condition contains a derivative this complicates things a bit. We proceed as follws.

In this case we seek approximations to our solution at the $(N+2)$ points $x_0, \cdots, x_{N+1}$ from our finite difference scheme given above.

$$-\big(1 + \frac{h}{2} p(x_i)\big) w_{i-1} + (2 + h^2 q(x_i)) w_i - \big(1 - \frac{h}{2} p(x_i)\big) w_{i+1} = -h^2 r(x_i)$$

for $i = 0, 1, \cdots, (N+1)$.

The problem is that the first equation contains w_{-1} which is to approximate $y(x_{-1})$ and the last equation contains w_{N+2} which is to approximate $y(x_{N+2})$. The points $x_{-1} = a - h$ and $x_{N+2} = b + h$ lie outside the interval.

To take care of this problem and maintain order $O(h^2)$ accuracy we use a centered approximation for y' in the boundary condition to solve for $y(x_{-1})$ and $y(x_{N+2})$. Recall, using a center difference we have

$$\frac{y(x_{N+2}) - y(x_N)}{2h} = y'(x_{N+1}) + O(h), \quad \frac{y(x_1) - y(x_{-1})}{2h} = y'(x_0) + O(h).$$

Using these in the boundary conditions

$$b_0 y(x_{N+1}) + b_1 y'(x_{N+1}) = \beta, \quad a_0 y(x_0) - a_1 y'(x_0) = \alpha$$

we get

$$b_0 y(x_{N+1}) + b_1 \left(\frac{y(x_{N+2}) - y(x_N)}{2h} + O(h) \right) = \beta,$$

and

$$a_0 y(x_0) - a_1 \left(\frac{y(x_1) - y(x_{-1})}{2h} + O(h) \right) = \alpha.$$

Solving for $y(x_{-1})$ and $y(x_{N+2})$ we get

$$y(x_{N+2}) = y(x_N) + 2h \left[\frac{\beta}{b_1} - \frac{b_0}{b_1} y(x_{N+1}) \right] + hO(h),$$

and

$$y(x_{-1}) = y(x_1) + 2h \left[\frac{\alpha}{a_1} - \frac{a_0}{a_1} y(x_0) \right] + hO(h).$$

Thus we obtain order $O(h^2)$ approximations $w_{-1} \approx y(x_{-1})$ and $w_{N+2} \approx y(x_{N+2})$ given by

$$w_{-1} = w_1 + \left[\frac{\alpha}{a_1} - \frac{a_0}{a_1} w_0 \right],$$

and

$$w_{N+2} = w_N + 2h \left[\frac{\beta}{b_1} - \frac{b_0}{b_1} w_{N+1} \right].$$

We now plug these values into the first and last equations to eliminate the terms that correspond to points outside the interval. In particular, the equation for $i = 0$ becomes

$$\begin{aligned} -\left(1 + \frac{h}{2} p(x_0)\right) \left[w_1 + 2h \left(\frac{\alpha}{a_1} - \frac{a_0}{a_1} w_0 \right) \right] + (2 + h^2 q(x_0)) w_0 \\ -\left(1 - \frac{h}{2} p(x_0)\right) w_1 = -h^2 r(x_0) \end{aligned}$$

and for $i = N+1$

$$-\left(1+\frac{h}{2}p(x_{N+1})\right)w_N + (2+h^2q(x_{N+1}))w_{N+1}$$
$$-\left(1-\frac{h}{2}p(x_{N+1})\right)\left[w_N + 2h\left(\frac{\beta}{b_1}-\frac{b_0}{b_1}w_{N+1}\right)\right] = -h^2r(x_{N+1})$$

Collecting like terms, these equations can be written for $i = 0$ as

$$\left[(2+h^2q(x_0)) + 2h\frac{a_0}{a_1}\left(1+\frac{h}{2}p(x_0)\right)\right]w_0 - 2w_1 = -h^2r(x_0) + 2h\frac{\alpha}{a_1}\left(1+\frac{h}{2}p(x_0)\right)$$

$$-2w_N + \left[(2+h^2q(x_{N+1})) + 2h\frac{b_0}{b_1}\left(1-\frac{h}{2}p(x_{N+1})\right)\right]w_{N+1} = -h^2r(x_{N+1}) + 2h\frac{\beta}{b_1}\left(1-\frac{h}{2}p(x_{N+1})\right)$$

Defining

$$c_0 = \left[(2+h^2q(x_0)) + 2h\frac{a_0}{a_1}\left(1+\frac{h}{2}p(x_0)\right)\right],$$

and

$$c_{N+1} = \left[(2+h^2q(x_{N+1})) + 2h\frac{b_0}{b_1}\left(1-\frac{h}{2}p(x_{N+1})\right)\right]$$

the system can be written in amtrix form as

$$AW = B$$

where

$$A = \begin{bmatrix} c_0 & -2 & 0 & 0 & \cdots & 0 \\ -1-\frac{h}{2}p(x_1) & 2+h^2q(x_1) & -1+\frac{h}{2}p(x_1) & 0 & \cdots & 0 \\ 0 & -1-\frac{h}{2}p(x_2) & 2+h^2q(x_2) & -1+\frac{h}{2}p(x_2) & & 0 \\ \vdots & \ddots & \ddots & \ddots & & \vdots \\ 0 & \cdots & 0 & -1-\frac{h}{2}p(x_N) & 2+h^2q(x_N) & -1+\frac{h}{2}p(x_N) \\ 0 & 0 & \cdots & 0 & -2 & c_{N+1} \end{bmatrix}$$

$$W = \begin{bmatrix} w_0 \\ w_1 \\ \vdots \\ w_{N+1} \end{bmatrix}, \quad B = \begin{bmatrix} -h^2r(x_0) + 2h\frac{\alpha}{a_1}\big(1+\frac{h}{2}p(x_0)\big) \\ -h^2r(x_1) \\ \vdots \\ -h^2r(x_N) \\ -h^2r(x_{N+1}) + 2h\frac{\beta}{b_1}\big(1-\frac{h}{2}p(x_{N+1})\big) \end{bmatrix}.$$