Solving ODEs with MATLAB: Instructor's Manual

L.F. Shampine and I. Gladwell
Mathematics Department
Southern Methodist University
Dallas, TX 75275

S. Thompson
Department of Mathematics & Statistics
Radford University
Radford, VA 24142

Contents

Chapter 1

Getting Started

1.1 Introduction

1.2 Existence, Uniqueness, and Well-Posedness

Solution for Exercise 1.1. It is easily verified that both solutions returned by `dsolve` are solutions of the IVP. This fact does not conflict with the basic existence and uniqueness result because that result is for IVPs written in the standard (explicit) form

$$y' = f(t,y), \qquad y\left(t_0\right) = y_0$$

In fact, when we write the given IVP in this form, we obtain two IVPs,

$$y' = f_1(t,y) = \sqrt{1-y^2}, \qquad y(0) = 0$$

and

$$y' = f_2(t,y) = -\sqrt{1-y^2}, \qquad y(0) = 0$$

It is easily verified that both functions, f_1 and f_2, satisfy a Lipschitz condition on a region containing the initial condition, hence both IVPs have a unique solution. For example,

$$\left|\frac{\partial f_1}{\partial y}\right| = \left|\frac{-y}{\sqrt{1-y^2}}\right| \le \frac{0.5}{\sqrt{0.75}}$$

for $-0.5 \le y \le 0.5$ and for all t. In this way we find that the given IVP has exactly two solutions.

Solution for Exercise 1.2. By definition, $f(t,y)$ satisfies a Lipschitz condition with constant L in a region if

$$|f(t,u) - f(t,v)| \le L|u-v|$$

for all (t,u), (t,v) in the region. If this function $f(t,y)$ satisfies a Lipschitz condition on $|t| \le 1, |y| \le 1$, then

$$|\sqrt{|u|} - \sqrt{|0|}\,| = \sqrt{|u|} \le L|u| = L|u-0|$$

This implies that

$$\frac{1}{\sqrt{|u|}} \le L$$

However, if we let $u \to 0$, we see that this is not possible. Accordingly, $f(t, y)$ does not satisfy a Lipschitz condition on this rectangle. Two solutions to the IVP are $y(t) \equiv 0$ and $y(t) = \dfrac{t^2}{4}$.

On the rectangle $|t| \leq 1, 0 < \alpha \leq y \leq 1$,

$$\left|\frac{\partial f}{\partial y}\right| = \left|\frac{1}{2\sqrt{y}}\right| \leq \frac{1}{2\sqrt{\alpha}}$$

This upper bound on the magnitude of the partial derivative serves as a Lipschitz constant for $f(t, y)$ on this rectangle.

Solution for Exercise 1.3. For the first ODE we can use partial fractions to obtain

$$\frac{1}{(t-1)(t-2)} = \frac{1}{t-2} - \frac{1}{t-1}$$

and then integrate to find that

$$y(t) = C \ln\left(\left|\frac{t-2}{t-1}\right|\right)$$

The arbitrary constant C is determined by the initial condition. Depending on where the initial value is specified and on its value, the maximal interval on which the solution is defined is one of $(-\infty, 1)$, $(1, 2)$ and $(2, \infty)$.

For the second ODE, separating variables to get $y' y^{-4/3} = -3\sin(t)$ and integrating leads to

$$y(t) = \frac{1}{(C - \cos(t))^3}$$

If the initial condition is such that the arbitrary constant C satisfies either $C > 1$ or $C < -1$, the maximal interval on which the solution is defined is $(-\infty, \infty)$. However, if $-1 \leq C \leq 1$, the interval extends from the initial point in each direction only until the first point is reached where $\cos(t) = C$.

Solution for Exercise 1.4. The command

```
>> dfs('5*(y - t^2)',[0 5 -2 20]);
```

produces a direction field for the ODE and clicking at a few points in the window shows the instability of the IVP that is studied in the text by means of its analytical solution $(t^2 + 0.4t + 0.08) + Ce^{5t}$.

Solution for Exercise 1.5. The text states that the general solution of the ODE is

$$t^2 + 0.4t + 0.08 + Ce^{5t}$$

for an arbitrary constant C. From this we see that the solution of the IVP is

$$y(t) = t^2 + 0.4t + 0.08$$

and the local solution that goes through (t_n, y_n) is

$$u(t) = y(t) + (y_n - y(t_n))\, e^{5(t-t_n)}$$

The program `ex05ch1.m` does the computations in a straightforward way using these expressions. This IVP is unstable because of the term Ce^{5t} in the general solution. The local errors are small in the beginning and so are the global errors because the instability due to the exponential term is modest. As t increases, the exponential term and the instability of the problem grow rapidly, so we expect the global errors to grow rapidly, and they do.

1.3 Standard Form

Solution for Exercise 1.6. Expanding the derivative in the ODE gives

$$p(x)y''(x) + p'(x)y'(x) + q(x)y(x) = r(x)$$

Letting $y_1 = y$ and $y_2 = y'$, we have $y_1' = y' = y_2$ and

$$y_2' = y'' = \frac{r - p'y' - qy}{p} = \frac{r - p'y_2 - qy_1}{p}$$

The given BVP thus takes the form

$$\begin{aligned} y_1' &= y_2, \quad y_1(0) = 0 \\ y_2' &= \frac{r - p'y_2 - qy_1}{p}, \quad y_2(1) = \frac{2}{p(1)} \end{aligned}$$

Letting $y_1 = y$ and $y_2 = py'$, we have $y_1' = y' = \dfrac{y_2}{p}$ and

$$y_2' = py'' + p'y' = r - qy = r - qy_1$$

With these variables, the given BVP assumes the form

$$\begin{aligned} y_1' &= \frac{y_2}{p}, \quad y_1(0) = 0 \\ y_2' &= (py')' = r - qy = r - qy_1, \quad y_2(1) = 2 \end{aligned}$$

Solution for Exercise 1.7. A little manipulation and taking a square root provides the two equivalent ODEs

$$y'' = \pm\frac{e^x}{\sqrt{y}}$$

which are explicit and in special second order form. If we let $y_1 = y$ and $y_2 = y'$, we obtain the equivalent first order system

$$\begin{aligned} y_1' &= y_2 \\ y_2' &= \pm\frac{e^x}{\sqrt{y_1}} \end{aligned}$$

Solution for Exercise 1.8. Denote the variables $f,\ f',\ f'',\ f'''$ by $y_1,\ y_2,\ y_3,\ y_4$, respectively. A first order system of ODEs for these unknowns and the corresponding boundary conditions are obtained in the usual way. Further define

$$y_5(\eta) = \int_0^\eta [1 - f'(\xi)e^{\Omega\xi}]d\xi$$

and

$$y_6(\eta) = \int_0^\eta f'(\xi)e^{\Omega\xi}[1 - f'(\xi)e^{\Omega\xi}]d\xi$$

With initial values $y_5(0) = 0$ and $y_6(0) = 0$, we have $\Delta^* = y_5(b)$ and $\theta = y_6(b)$. Applying the Second Fundamental Theorem of Calculus to the definitions of $y_5(\eta)$ and $y_6(\eta)$ then leads to ODEs for these unknowns in terms of the other unknowns. Altogether we have the system of first order ODEs

$$\begin{aligned} y_1' &= y_2 \\ y_2' &= y_3 \\ y_3' &= y_4 \\ y_4' &= -\left(\Omega + y_1\right)y_4 - \Omega y_1 y_3 + \left(2\beta - 1\right)y_2 y_3 - \Omega y_2^2 \\ y_5' &= 1 - y_2 e^{\Omega\eta} \\ y_6' &= y_2 e^{\Omega\eta}\left(1 - y_2 e^{\Omega\eta}\right) \end{aligned}$$

and boundary conditions

$$\begin{aligned} y_1(0) &= 0 \\ y_2(0) &= 0 \\ y_2(b) &= e^{-\Omega b} \\ y_3(b) &= -\Omega e^{-\Omega b} \\ y_5(0) &= 0 \\ y_6(0) &= 0 \end{aligned}$$

Solution for Exercise 1.9. Let $y_1(x) = \mu(x)$, $y_2(x) = \mu'(x)$, $y_3(x) = \int_0^x \dfrac{dt}{\sqrt{1+y_1^2(t)}}$ and $y_4(x) = H$. These variables satisfy the first order ODEs

$$\begin{aligned} y_1' &= y_2 \\ y_2' &= -\omega^2 \left(\frac{1-\alpha^2}{y_4} \frac{1}{\sqrt{1+y_1^2}} + \alpha^2 \right) y_1 \\ y_3' &= \frac{1}{\sqrt{1+y_1^2}} \\ y_4' &= 0 \end{aligned}$$

The boundary conditions are

$$\begin{aligned} y_1(0) &= \epsilon \\ y_2(0) &= 0 \\ y_2(1) &= 0 \\ y_4(1) &= \frac{1}{\alpha^2}\left[1-(1-\alpha^2)y_3(1)\right] \end{aligned}$$

Solution for Exercise 1.10. For convenience, throughout we'll not show the independent variable t. What is striking about this problem is that for each of the two given canonical forms, the direct dependence on the derivatives $x^{(i)}$ is suppressed.

Consider the first canonical form. Since $v_1 = y - b_N x$, we see that for $i = 1, 2, \ldots, N-1$,

$$\begin{aligned} v_i' &= (-a_{N-i}v_1 + v_{i+1}) + (b_{N-i} - a_{N-i}b_N)\,x \\ &= -a_{N-i}\,(y - b_N x) + v_{i+1} + b_{N-i}x - a_{N-i}b_N x \\ &= b_{N-i}x - a_{N-i}y + v_{i+1} \end{aligned}$$

In particular $v_1' = b_{N-1}x - a_{N-1}y + v_2$. By differentiating the latter equation repeatedly, substituting the former equations for v_i', and simplifying, we obtain

$$\begin{aligned} v_1'' &= b_{N-1}x' + b_{N-2}x - a_{N-1}y' - a_{N-2}y + v_3 \\ v_1''' &= b_{N-1}x'' + b_{N-2}x' + b_{N-3}x - a_{N-1}y'' - a_{N-2}y' - a_{N-3}y + v_4 \\ &\vdots \\ v_1^{(N-1)} &= b_{N-1}x^{(N-2)} + b_{N-2}x^{(N-3)} + \cdots + b_1 x \\ &\quad -a_{N-1}y^{(N-2)} - a_{N-2}y^{(N-3)} - \cdots - a_1 y + v_N \end{aligned}$$

Differentiating the equation for $v_1^{(N-1)}$ and using the fact that

$$v_N' = -a_0 v_1 + (b_0 - a_0 b_N)\,x$$

gives

$$\begin{aligned} v_1^{(N)} &= b_{N-1}x^{(N-1)} + b_{N-2}x^{(N-2)} + \cdots + b_1 x' \\ &\quad -a_{N-1}y^{(N-1)} - a_{N-2}y^{(N-2)} - \cdots - a_1 y' - a_0 v_1 + (b_0 - a_0 b_N)\,x \end{aligned}$$

Equating this expression to $(y - b_N x)^{(N)} = y^{(N)} - b_N x^{(N)}$ and simplifying gives

$$\sum_{i=0}^{N} b_i x^{(i)} = y^{(N)} + \sum_{i=1}^{N-1} a_i y^{(i)} + a_0 (v_1 + b_N x)$$

or

$$\sum_{i=0}^{N} b_i x^{(i)} = y^{(N)} + \sum_{i=0}^{N-1} a_i y^{(i)}$$

which is the original ODE.

Now consider the second canonical form. Define w as the solution of

$$w^{(N)} + \sum_{j=0}^{N-1} a_j w^{(j)} = x$$

and let

$$W = \left(w, w', \ldots, w^{(N)}\right)^T = (w_1, w_2, \ldots, w_N)^T$$

Note that the given system for v' is the same as that for W' when the above equation for $w^{(N)}$ is reduced to a system of first order equations in the usual way. We need to show that $y = \sum_{i=0}^{N} b_i w^{(i)}$ is the same as the given output form

$$y = (b_0 - a_0 b_N) w_1 + (b_1 - a_1 b_N) w_2 + \cdots + (b_{N-1} - a_{N-1} b_N) w_N + b_N x$$

This is true however, since

$$\begin{aligned}
\sum_{i=0}^{N} b_i w^{(i)} &= \sum_{i=0}^{N-1} b_i w^{(i)} + b_N w^{(N)} \\
&= \sum_{i=0}^{N-1} b_i w^{(i)} + b_N \left(x - \sum_{i=0}^{N-1} a_i w_{i+1} \right) \\
&= \sum_{i=0}^{N-1} (b_i - a_i b_N) w_{i+1} + b_N x
\end{aligned}$$

Now we consider initial values for the first canonical form. Since $y = v_1 + b_N x$ we have $v_1(0) = y(0) - b_N x(0)$. We also have

$$\begin{aligned}
v_2 &= v_1' - b_{N-1} x + a_{N-1} y \\
&= y' - b_N x' - b_{N-1} x + a_{N-1} y \\
v_3 &= v_2' - b_{N-2} x + a_{N-2} y \\
&= y'' - b_N x'' - b_{N-1} x' + a_{N-1} y' - b_{N-2} x + a_{N-2} y \\
&\vdots \\
v_{i+1} &= v_i' - b_{N+1-i} x + a_{N+1-i} y \\
&= y^{(i)} - \textstyle\sum_{j=0}^{i} b_{N-i-j} x^{(j)} + \sum_{j=0}^{i-1} a_{N-i-j} y^{(j)}
\end{aligned}$$

Substituting $t = 0$ shows that $v_{i+1}(0)$ may be expressed in terms of the values

$$y(0), y'(0), \ldots, y^{(i)}(0), x(0), x'(0), \ldots, x^{(i)}(0)$$

Consider the second canonical form. Let $c_i = b_i - a_i b_N$. Then

$$y = c_1 v_1 + c_2 v_2 + \cdots + c_N v_N + b_N x$$

Differentiate the previous equation and substitute for v_N' to obtain

$$\begin{aligned}
y' &= c_1 v_1' + c_2 v_2' + \cdots + c_N v_N' + b_N x' \\
&= c_1 v_2 + c_2 v_2' + \cdots + c_{N-1} v_N + c_N (-a_0 v_1 - a_1 v_2 - \cdots - a_{N-1} v_N) + b_N x' + x
\end{aligned}$$

Repeat this to obtain

$$\begin{aligned} y'' &= c_1v_2' + c_2v_3' \cdots + c_{N-2}v_{N-1}' + c_{N-1}\left(-a_0v_1 - a_1v_2 - \cdots - a_{N-1}v_N\right) \\ &\quad + b_Nx'' + x' + c_N\left(-a_0v_1 - a_1v_2 - \cdots - a_{N-1}v_N\right) + x \\ &= c_1v_3 + c_2v_4 + \cdots + c_{N-2}v_N + c_{N-1}\left(-a_0v_1 - a_1v_2 - \cdots - a_{N-1}v_N\right) \\ &\quad + c_N\left(-a_0v_1 - a_1v_2 - \cdots - a_{N-1}v_N\right) + b_Nx'' + x' + x \end{aligned}$$

Continue this process as far as $y^{(i)}$ and substitute $t = 0$ to obtain a system of linear algebraic equations whose coefficients involve $y^{(i)}(0)$ and $x^{(i)}(0)$. The system may be solved to obtain $v_1, v_2, \dots, v_N$.

1.4 Control of the Error

Solution for Exercise 1.11. The control of `bvp4c` is

$$|y_i(t_n) - y_{n,i}| \le re|y_i(t_n)| + ae_i \tag{1.1}$$

The control of `MIRKDC` is

$$|y_i(t_n) - y_{n,i}| \le \tau|y_i(t_n)| + \tau$$

which is identical when $re = \tau$ and when $ae_i = \tau$ for each i. The control of `DVERK` is only roughly equivalent to (1.1). If $|y_i(t_n)| \le 1$, the control is

$$|y_i(t_n) - y_{n,i}| \le \tau$$

which is a pure absolute error control with tolerance τ. If $|y_i(t_n)| > 1$, the control is

$$|y_i(t_n) - y_{n,i}| \le \tau|y_i(t_n)|$$

which is a pure relative error control with tolerance τ. The two cases correspond in a rough way to (1.1) with $re = \tau$ and $ae_i = \tau$ for all i.

Solution for Exercise 1.12. At $t = \frac{\pi}{2}$ the solution has the value 1, so is on the edge of the region where $f(t, y)$ is defined. No matter what kind of error control you use, it will allow a numerical approximation $y_n > 1$ when $y(t)$ is sufficiently close to 1. Accordingly, as the solver approaches $t = \frac{\pi}{2}$, it is very possible that the code will try to form an approximation $y_n > 1$ and the computation may fail in the evaluation of the square root. The other difficulty is suggested by the observation that

$$\frac{\partial f}{\partial y} = \frac{-y}{\sqrt{1 - y^2}}$$

is unbounded as $(t, y) \to \left(\frac{\pi}{2}, 1^-\right)$. This tells us that $f(t, y)$ does not satisfy a Lipschitz condition in a region containing this point and it is possible that any solution is not unique. In fact there are other solutions of the ODE that pass through this point, $y(t) \equiv 1$ being one. When uniqueness breaks down, we don't know which solution a code will compute, if any.

Solution for Exercise 1.13. The solution approaches very quickly the boundary of the region where the function $\ln(y)$ is defined. No matter what kind of error control you use, it will allow a numerical approximation $y_n \le 0$ when $y(t)$ is sufficiently close to 0. Accordingly, for "large" t, it is very possible that the code will produce an approximation $y_n \le 0$ and then the computation may fail in the evaluation of the natural logarithm. Just what happens depends on the computing environment. Often there is an immediate failure, but MATLAB evaluates the logarithm as a complex number or `Inf`. Depending on the solver, the integration might continue on toward the end of the interval specified.

1.5 Qualitative Properties

Solution for Exercise 1.14. Add the differential equations to see that

$$\sum_{i=1}^{10} y_i'(t) = \frac{d}{dt}\left(\sum_{i=1}^{10} y_i(t)\right) = 0$$

hence that the sum of the solution components is constant.

Solution for Exercise 1.15. Differentiate $G(t, x(t), y(t))$ to obtain

$$G' = x^{-c}\left(y^{-a}e^{cx+ay}\left(cx' + ay'\right) - ay^{-a-1}y'e^{cx+ay}\right) - cx^{-c-1}x'y^{-a}e^{cx+ay}$$

and then divide out a common factor to obtain

$$xy\left(cx' + ay'\right) - axy' - cx'y = 0$$

Use the fact that

$$cx'ay' = ac(x - y)$$

to see that the last expression is identically zero, hence $G(t, x(t), y(t))$ is constant.

The program `ex15ch1.m` uses Euler's method to solve the given IVP. It plots the solution in the phase plane and in another figure, the conserved quantity G. For a step size $h = 0.01$, the curve doesn't quite close and $G(t, x(t), y(t))$ varies from approximately 120 to 165. For $h = 0.001$, the curve appears to be closed and $G(t)$ varies from approximately 122 to 125. The program `ex15bch1.m` solves the IVP with `ode45`. With default error tolerances the curve doesn't quite close and $G(t)$ ranges from about 122 to 125. Reducing the relative error tolerance to `1e-6`, the curve appears to be closed and $G(t)$ is approximately 121.85.

Chapter 2

Initial Value Problems

2.1 Introduction

2.2 Numerical Methods for IVPs

2.2.1 One–Step Methods

Local Error Estimation

Runge–Kutta Methods

Explicit Runge–Kutta Formulas

Solution for Exercise 2.1.

- For Simpson's method $A_1 = \frac{1}{6}$, $A_2 = \frac{4}{6}$, $A_3 = \frac{1}{6}$, $\alpha_1 = 0$, $\alpha_2 = \frac{1}{2}$, and $\alpha_3 = 1$. Direct substitution shows that the first four order equations are satisfied but that

$$A_1{\alpha_1}^4 + A_2{\alpha_2}^4 + A_3{\alpha_3}^4 = \frac{5}{24} \neq \frac{1}{5}$$

- For the Gaussian 2-point method

$$A_1 = A_2 = \frac{1}{2}, \alpha_1 = \frac{\sqrt{3}-1}{2\sqrt{3}}, \alpha_2 = \frac{\sqrt{3}+1}{2\sqrt{3}}$$

  Direct substitution shows that the first four order equations are satisfied but that

$$A_1{\alpha_1}^4 + A_2{\alpha_2}^4 + A_3{\alpha_3}^4 = \frac{7}{36} \neq \frac{1}{5}$$

Solution for Exercise 2.2. We use Euler's method to calculate

$$y_{n,1} = y_n + hf\left(t_n, y_n\right)$$

When we use the midpoint rule, we need to apply $y_{n,1}$ with a step of size $\frac{h}{2}$. We then obtain the formula

$$y_{n+1} = y_n + h\left(f\left(t_n + \frac{h}{2}, y_n + \frac{h}{2}f\left(t_n, y_n\right)\right)\right)$$

Solution for Exercise 2.3. For the desired order of accuracy, the coefficients must satisfy

$$\begin{aligned}\gamma_1 + \gamma_2 &= 1\\ \gamma_2 \alpha_1 &= \frac{1}{2}\\ \gamma_2 \beta_{1,0} &= \frac{1}{2}\end{aligned}$$

To obtain these equations, first expand $f_{n,2}$ in a Taylor series:

$$f_{n,2} = f(t_n, y_n) + h\alpha_1 \frac{\partial f}{\partial t}(t_n, y_n) + h\beta_{1,0} f(t_n, y_n) \frac{\partial f}{\partial y}(t_n, y_n) + \dots$$

Then substitute this expansion into the formula to obtain

$$\begin{aligned} y_{n+1} &= y_n + h\gamma_1 f_{n,1} + \\ & \quad h\gamma_2 \left(f(t_n, y_n) + h\alpha_1 \frac{\partial f}{\partial t}(t_n, y_n) + h\beta_{1,0} f(t_n, y_n) \frac{\partial f}{\partial y}(t_n, y_n) \right) + \dots \end{aligned}$$

Now expand the solution, using the fact that $u'(t) = f(t, u(t))$ to see that

$$\begin{aligned} u(t_n + h) &= y_n + hu'(t_n) + \frac{h^2}{2} u''(t_n) + \dots \\ &= y_n + hf(t_n, y_n) + \frac{h^2}{2} \left(\frac{\partial f}{\partial t}(t_n, y_n) + f(t_n, y_n) \frac{\partial f}{\partial y}(t_n, y_n) \right) + \dots \end{aligned}$$

Equating coefficients of powers of h in the expansions of y_{n+1} and $u(t_{n+1})$ gives the stated equations of condition.

Continuous Extensions

Solution for Exercise 2.4. Table 2.1 displays the Butcher tableau for this pair.

0			
$\frac{1}{2}$	$\frac{1}{2}$		
$\frac{3}{4}$	0	$\frac{3}{4}$	
	0	1	0
	$\frac{2}{9}$	$\frac{3}{9}$	$\frac{4}{9}$

Table 2.1: The (2,3) pair.

The local error estimate is

$$E_{n+1} = y^*_{n+1} - y_{n+1} = h \left(\frac{2}{9} f_{n,1} - \frac{6}{9} f_{n,2} + \frac{4}{9} f_{n,3} \right)$$

We would accept a step if

$$|E_{n+1}| \le \epsilon_a + \epsilon_r |y^*_{n+1}|$$

where ϵ_a and ϵ_r are the absolute and relative error tolerances, respectively. Otherwise, we would take the step again with a smaller step size. To determine the step size for the next step in the case this step was successful or to determine the step size with which to redo a failed step, we would calculate $h_{\text{new}} = \alpha h$ so as to make the quantity $h_{\text{new}}^3 \dfrac{e_{n+1}}{\epsilon_a + \epsilon_r |y_{n+1}^*|} = \alpha^3 h^3 \dfrac{e_{n+1}}{\epsilon_a + \epsilon_r |y_{n+1}^*|}$ approximately equal to 1. Solving gives

$$\alpha = \left(\frac{\epsilon_a + \epsilon_r |y_{n+1}^*|}{|y_{n+1}^* - y_{n+1}|} \right)^{1/3}$$

Since we're dealing with asymptotic estimates, in practice we would aim conservatively for a value of α that is rather less than 1, say 0.5. Our estimate of α would be reduced in this case by a factor of $(0.5)^{1/3}$ or approximately 0.8. In an actual RK code we would limit the amount by which the step size is allowed to change on any step (e.g., some RK codes would require that α be no less than 0.5 and no larger than 2.0 for any given step).

2.2.2 Methods with Memory

Adams Methods

BDF methods

Error Estimation and Change of Order

Continuous Extensions

Solution for Exercise 2.5.

- The polynomial $P(t)$ that interpolates (t_{n-1}, f_{n-1}) and (t_n, f_n) for AB2 is given by

$$P(t) = f_n + \frac{t - t_n}{t_n - t_{n-1}} (f_n - f_{n-1})$$

  Thus

$$\begin{aligned} y_{n+1} &= y_n + \int_{t_n}^{t_n+h} \left(f_n + \frac{t - t_n}{t_n - t_{n-1}} (f_n - f_{n-1}) \right) dt \\ &= y_n + f_n (t - t_n) |_{t_n}^{t_n+h} + \frac{(x - t_n)^2}{2} \left(\frac{f_n - f_{n-1}}{t_n - t_{n-1}} \right) |_{t_n}^{t_n+h} \\ &= y_n + h_n f_n + \frac{h_n^2}{2} \left(\frac{f_n - f_{n-1}}{t_n - t_{n-1}} \right) \\ &= y_n + h_n \left(\left(1 + \frac{h_n}{2h_{n-1}} \right) f_n - \left(\frac{h_n}{2h_{n-1}} \right) f_{n-1} \right) \\ &= y_n + h_n \left(\left(1 + \frac{r}{2} \right) f_n - \left(\frac{r}{2} \right) f_{n-1} \right) \end{aligned}$$

- The polynomial that interpolates (t_{n-1}, y_{n-1}), (t_n, y_n), and (t_{n+1}, y_{n+1}), for BDF2 is given by

$$\begin{aligned} P(t) &= y_{n-1} \frac{(t - t_n)(t - t_{n+1})}{(t_{n-1} - t_n)(t_{n-1} - t_{n+1})} + y_n \frac{(t - t_{n-1})(t - t_{n+1})}{(t_n - t_{n-1})(t_n - t_{n+1})} \\ &\quad + y_{n+1} \frac{(t - t_{n-1})(t - t_n)}{(t_{n+1} - t_{n-1})(t_{n+1} - t_n)} \\ &= y_{n-1} \frac{(t - t_n)(t - t_{n+1})}{h_{n-1}(h_{n-1} + h_n)} + y_n \frac{(t - t_{n-1})(t - t_{n+1})}{h_{n-1} h_n} \\ &\quad + y_{n+1} \frac{(t - t_{n-1})(t - t_n)}{h_n (h_{n-1} + h_n)} \end{aligned}$$

Differentiating this expression gives

$$\begin{aligned}P'(t) &= y_{n-1}\frac{2t-t_n-t_{n+1}}{h_{n-1}(h_{n-1}+h_n)} + y_n\frac{2t-t_{n-1}-t_{n+1}}{h_{n-1}h_n}\\ &+ y_{n+1}\frac{2t-t_{n-1}-t_n}{h_n(h_{n-1}+h_n)}\end{aligned}$$

Substituting $t = t_{n+1}$ gives

$$\begin{aligned}P'(t_{n+1}) &= y_{n-1}\frac{h_n}{h_{n-1}(h_{n-1}+h_n)} - y_n\frac{h_n+h_{n-1}}{h_{n-1}h_n}\\ &+ y_{n+1}\frac{h_{n-1}+2h_n}{h_n(h_{n-1}+h_n)}\end{aligned}$$

When the last expression is multiplied by h_n, the coefficient of y_{n-1} becomes

$$\frac{{h_n}^2}{h_{n-1}(h_{n-1}+h_n)} = \frac{\left(\dfrac{h_n}{h_{n-1}}\right)^2}{1+\dfrac{h_n}{h_{n-1}}} = \frac{r^2}{1+r}$$

The coefficient of y_n becomes

$$h_n\left(\frac{h_{n-1}+h_n}{h_{n-1}h_n}\right) = 1+\frac{h_n}{h_{n-1}} = 1+r$$

The coefficient of y_{n+1} becomes

$$\frac{h_{n-1}+2h_n}{h_{n-1}+h_n} = \frac{1+2\dfrac{h_n}{h_{n-1}}}{1+\dfrac{h_n}{h_{n-1}}} = \frac{1+2r}{1+r}$$

Thus

$$\left(\frac{1+2r}{1+r}\right)y_{n+1} - (1+r)y_n + \left(\frac{r^2}{1+r}\right)y_{n-1} = h_n f(t_{n+1}, y_{n+1})$$

Solution for Exercise 2.6. Using the result cited at the beginning of the chapter regarding the error in the interpolating polynomial we see that, in this case, the error has the form

$$y'(t) - P(t) = \frac{y'''(c)}{2}(t-t_n)(t-t_{n+1})$$

so that the error in y_{n+1} is equal to

$$\begin{aligned}\int_{t_n}^{t_{n+1}} (y(t)-P(t))\,dt &= y(t_{n+1}) - \left(y(t_n) + \frac{h}{2}\left(y'_{n+1}+y'_n\right)\right)\\ &= \frac{1}{2}\int_{t_n}^{t_{n+1}} y'''(c(t))(t-t_n)(t-t_{n+1})dt\end{aligned}$$

Since $(t-t_n)(t-t_{n+1})$ doesn't change sign on the interval (t_n, t_{n+1}), we may use an Integral Mean Value Theorem to express the error as

$$\frac{1}{2}y'''(\psi)\int_{t_n}^{t_{n+1}} (t-t_n)(t-t_{n+1})dt$$

for some point ψ in (t_n, t_{n+1}). The value of the last integral is $-\dfrac{(t_{n+1}-t_n)^3}{6}$. Hence the error is $-\dfrac{1}{12}h^3y'''(\psi)$ or approximately $-\dfrac{1}{12}h^3y'''(t_n)$.

Solution for Exercise 2.7. The local truncation error of AM2 is

$$-\frac{1}{12}h^3y^{(3)}(t_n)+\ldots$$

For this differential equation, $y^{(3)}(t_n)=-y(t_n)$, hence

$$y(t_n+h)-y_{n+1}=\frac{1}{12}h^3y(t_n)+\ldots$$

for AM2. Alternatively, this expression can be derived by direct expansion.

The predictor-corrector scheme applied to this ODE predicts

$$p=y(t_n)+h(-y(t_n))=(1-h)y(t_n)$$

with AB1 and then evaluates and corrects with AM2 to get

$$y_{n+1}=y(t_n)+\frac{h}{2}\left[(-y(t_n))+(-p)\right]=\left[1-h+\frac{h^2}{2}\right]y(t_n)$$

The true solution is

$$y(t_n+h)=e^{-h}y(t_n)=\left[1-h+\frac{h^2}{2!}-\frac{h^3}{3!}+\ldots\right]y(t_n)$$

The two expansions show that the error of AB1-AM2 in PECE form is

$$y(t_n+h)-y_{n+1}=-\frac{h^3}{3!}y(t_n)+\ldots$$

Solution for Exercise 2.8. For this formula

$$\alpha_0=1,\ \alpha_1=\frac{3}{2},\ \alpha_2=-3,\ \alpha_3=\frac{1}{2},\ \beta_0=0,\ \beta_1=3,\ \beta_2=0,\ \beta_3=0$$

With these values it is straightforward to evaluate the expressions given for the coefficients in the expansion

$$lte_n=\sum_{j=0}^{\infty}C_jh^jy^{(j)}(t_{n+1})$$

of the local truncation error. We find that $0=C_0=C_1=C_2=C_3$, which shows that the formula is of order 3.

The program `ex08ch2.m` performs the numerical experiment in a straightforward way and confirms the results displayed in Table 2.2.

Solution for Exercise 2.9.

- The stability region for the backward Euler method is the set of values z for which $\dfrac{1}{|1-z|}<1$ or equivalently for which $|1-z|>1$. It thus consists of those points whose distance from the point $(1,0)$ exceeds 1, that is, the exterior of the disk with radius 1 centered at $(1,0)$.

- The stability region for the trapezoidal rule is determined by the condition $\left|\dfrac{2+z}{2-z}\right|<1$. This may be written $\left|\dfrac{z-(-2)}{z-2}\right|<1$. The stability region thus consists of the points that are closer to $(-2,0)$ than to $(2,0)$; these points constitute the left half of the complex plane.

i	AB3	Unstable Formula
2	1.34e−003	9.68e−004
3	2.31e−004	6.16e−003
4	3.15e−005	1.27e+000
5	4.08e−006	6.43e+005
6	5.18e−007	2.27e+018
7	6.53e−008	4.23e+044
8	8.19e−009	2.27e+098
9	1.03e−009	1.03e+207
10	1.28e−010	Inf

Table 2.2: Maximum error when $h = 1/2^i$.

- If we apply Heun's method to the test equation $y' = \lambda y$, we obtain

$$\begin{aligned} y_{n,1} &= y_n + h\lambda y_n = y_n(1+h\lambda) \\ y_{n+1} &= y_n + h\left(\frac{1}{2}\lambda y_n + \frac{1}{2}h\lambda(1+h\lambda)y_n\right) \\ &= y_n\left(1 + h\lambda + \frac{1}{2}(h\lambda)^2\right) \end{aligned}$$

  The stability condition then becomes $\left|1 + z + \frac{z^2}{2}\right| < 1$. However, for large values z the magnitude of this quadratic exceeds 1; hence the stability region is finite. For example, the real portion of the region works out to be the interval $(-2, 0)$.

Solution for Exercise 2.10. The program `ex10ch2.m` is a straightforward implementation of all three methods. It asks the user to designate which method is to be used and then does the computations and plots the results as specified by the exercise.

Solution for Exercise 2.11. The leading term in the local truncation error of AB1 is

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

As stated in the text, the solution of the IVP is

$$y(t) = \frac{1}{10} + \frac{9}{10}e^{-100t}$$

whence

$$y''(t_n) = 9 \times 10^3\, e^{-100t_n}$$

The largest h for which $|lte_n| \le \tau$ is

$$h = e^{50t_n}\sqrt{\frac{2\tau}{9 \times 10^3}}$$

Although a small step size h is needed to satisfy the specified absolute accuracy requirement near $t = 0$, as the integration progresses, the exponential term grows very rapidly and so does the step size.

Solution for Exercise 2.12. For this IVP, based on the ODE $y' = \lambda y$, the backward Euler solution at $t = n$ is given by

$$y_{n+1} = \left(\frac{1}{1-h\lambda}\right)^n = \left(\frac{-1}{9}\right)^n$$

t	y
2.0	0.472
3.0	0.330
4.0	0.248
5.0	0.199
6.0	0.166
7.0	0.142
8.0	0.124
9.0	0.110
10.0	0.099

Table 2.3: Backward Euler solution

since $h = 1$ and $\lambda = 10$. This solution decays to 0 even though $\lambda > 0$. This happens because the value $h\lambda$ lies within the stability region for the backward Euler method. Hence, the numerical solution is damped and does not track the exponentially growing exact solution. This doesn't happen when using an adaptive implementation of backward Euler such as that in `ode15s` which varies the step size in order to maintain a locally accurate solution. When fixed step size solutions are used, there is a very real danger of using a step size that is too large. Some applications codes in fact intentionally use the smallest step size that will generate a numerically stable solution – *hoping* that acceptable accuracy will be achieved. Others intentionally use very large step sizes in order to damp solution components as quickly as possible. For example, one widely used applications code is well–known for its ability to remove completely a square wave from a hyperbolic system of PDEs with just one integration step.

Solution for Exercise 2.13. On the surface, the fixed step size backward Euler solution given in Table 2.3 appears to be adequate; and it is in fact fairly accurate for this problem. This accuracy is coincidental since no attempt was made to maintain local accuracy. The program `ex13ch2.m` solves the problem using each of `ode45` and `ode15s`. With default error tolerances, `ode45` requires 75 integration steps and yields an approximate solution of $-1.8 \cdot 10^{34}$ at $t = 10$. Using error tolerances of 10^{-8}, it requires 909 steps and yields an approximate solution of $1.4 \cdot 10^{30}$! With default error tolerances, `ode15s` requires 234 steps and yields a solution $1.0 \cdot 10^{36}$; and with error tolerances of 10^{-8}, requires 1266 steps and yields a solution of $1.0 \cdot 10^{31}$. What's going on? A careless answer is that the solvers behaved badly—when in fact they have performed well. Each yields a solution for which the weighted local error at each step is smaller than the desired error tolerance. Since neighboring solution curves diverge exponentially fast from the solution of interest, each solver is forced to track these unstable solution curves. What the solvers are telling us is there is an inherent instability in the ODE and that any physical model based on the ODE should be examined critically. While this may be a trivial observation for this particular problem, is may not at all be obvious that a more complicated model contains inherent instabilities, particularly if the instabilities have been hidden previously by *ad hoc* fixed step size solutions.

Solution for Exercise 2.14. The stability of a predictor–corrector pair with only one correction isn't the same as that for the corrector alone; the predictor has an effect on the stability. To see this substitute $y' = \lambda y$ into the predictor formula AB1

$$y_{n+1,p} = y_n + hy'_n$$

to obtain $y_{n+1,p} = (1 + h\lambda)\, y_n$. Next, evaluate $y'_{n+1,p} = \lambda(1 + h\lambda)y_n$. Substitute these values into the corrector formula BDF1 to obtain

$$y_{n+1=}y_{n+1,c} = \left(1 + z + z^2\right) y_n$$

where $z = h\lambda$. Thus, the pair is stable if $|1 + z + z^2| < 1$. The stability region is then certainly finite due to the quadratic term. This situation is somewhat similar to that for Heun's method (which we have seen in Solution 2.9 yields the stability condition $|1 + z + \frac{1}{2}z^2| < 1$).

Solution for Exercise 2.15. The numerical results are obtained easily by modifying the complete program supplied with the exercise. The Jacobian is

$$f_y = 2y - 3y^2$$

For $0 \leq y \leq 1$, the magnitude of the eigenvalue (Jacobian) is bounded by 5, hence a Lipschitz constant for $f(y)$ in this range of y is 5, which is not large. For $0 < y < \frac{2}{3}$, the eigenvalue is positive and for $\frac{2}{3} < y < 1$, it is negative. In the first part of the integration $y(t)$ is small and positive, so the eigenvalue is positive and the IVP is unstable in a linear stability analysis. In the last part of the integration $y(t)$ is close to 1, so the eigenvalue is negative and the IVP is stable.

The first part of the integration is not stiff because the IVP is not stable there. In this interval $y(t)$ is $O(\epsilon)$, so the eigenvalue has magnitude $O(\epsilon)$. On this interval of length $O(\epsilon^{-1})$, the product of the length and the dominant eigenvalue is $O(1)$, hence the IVP is only moderately unstable. In the interval where ignition takes place, the solution changes rapidly and we expect that a relatively small step size will be needed to resolve the change. That alone suggests that stability does not constrain the step size severely and the IVP is not stiff. However, we also observe that in this interval of length $O(1)$, the product of the length and the Lipschitz constant is $O(1)$, hence the IVP is not stiff. In the last part of the integration $y(t)$ is close to 1 and the eigenvalue is about -1. The interval is of length $O(\epsilon^{-1})$, so the product is also $O(\epsilon^{-1})$. In this interval the solution is slowly varying, the IVP is stable, and the product is large, so the IVP is stiff. The numerical results comparing a stiff solver `ode15s` to a non–stiff solver `ode45` confirm these theoretical arguments about the intervals on which this IVP is non–stiff and and those on which it is stiff.

Solution for Exercise 2.16. The program `ex14ch2.m` may be used for this exercise. The program uses values of θ from 0 to π in increments of one degree. It also makes use of the fact that the stability region is symmetric with respect to the real axis. In the program the values `method` $= 1$, 2 and 3 correspond to AB1, BDF1 and BDF2 respectively. For AB1 the stability region is the interior of the region whose circular boundary is generated when `method` $= 1$. For BDF1 and BDF2 the stability regions are the exteriors of the curves generated.

2.3 Solving IVPs in Matlab

Solution for Exercise 2.17. The program `ex17ch2.m` may be used to study this problem. When this program is executed, the fourth plot displays the three solutions obtained.

Solution for Exercise 2.18. The program `ex18ch2.m` is an easy modification of `ch2ex1.m` that plots the solution as it is computed. To solve the IVP with `ode45`, all you have to do is replace the name `ode15s` with `ode45`.

Solution for Exercise 2.19. The IVP is solved in a straightforward way with `ode15s` by the program `ex19ch2.m`. Changing the name of the solver to `ode45` and running the program again shows that it is not expensive to solve the IVP with a code based on explicit Runge–Kutta formulas; hence the IVP is at most modestly stiff.

Solution for Exercise 2.20. It is straightforward to make copies of the specified functions and modify the copy of `ode15s` as suggested. Once that is done, it is easy to modify `ch2ex1.m` to make the program `ex20ch2.m`. In the first part of the integration the solver must use a small step size to resolve the rapid changes in the solution. Because accuracy determines the step size, the problem is not stiff there and the iteration matrix is well-conditioned. As the integration progresses, the solution tends to a constant (vector). Correspondingly the step sizes increase steadily and the problem becomes increasingly stiff. The solver will not step past the end of the interval of integration. This causes the last step to be shortened and the condition of the iteration matrix to be improved correspondingly.

Solution for Exercise 2.21. The IVP is solved and the solution is plotted in a straightforward way by the program `ex21ch2.m`.

Solution for Exercise 2.22. The program `ex22ch2.m` contains the commands necessary to solve the five problems and to generate the corresponding phase plots.

2.3.1 Event Location

Solution for Exercise 2.23. Using `ode45`, the program `ex23ch2.m` solves this problem in a straightforward way. It tests whether the integration terminated on an event by testing whether `ie` is empty. The program reports that

```
Cork left bottle at t = 1.24577 with speed v(t) = 16.8994.
```

Solution for Exercise 2.24. Using `ode45`, the program `ex24ch2.m` solves this problem in a straightforward way. With the value of $\phi(0)$ coded, the program plots the trajectory and reports that

```
With phi(0) = 0.3782, the range is 4.99957.
```

Editing the program to use the other value of $\phi(0)$, it then reports that

```
With phi(0) = 9.7456, the range is 4.99959.
```

The program uses the fact that a terminal event at the initial point is ignored.

Solution for Exercise 2.25. The program `ex25ch2.m` may be used to solve this problem. The event occurs when $t = 5.1755$.

Solution for Exercise 2.26. The IVP to be solved is

$$\begin{aligned} \frac{dx}{dt} &= y, \quad x(x_0) = x_0 \\ \frac{dy}{dt} &= \frac{1}{2}f'(x), \quad y(x_0) = \sqrt{f\left(x_0\right)} \end{aligned}$$

In addition, we wish to use event location to determine the times t_i for which $x(t_i) = x_i$ for given values x_i.

Consider the integral $\int_0^x \frac{1}{\sqrt{s}} ds$. We solve the IVP

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

for $0 \le t \le 2$. We locate the times t_i for which $x(t_i) = x_i = i/10$ for $i = 1, 2, \ldots, 9$. Note that for purposes of checking the solution, elementary calculus may be used to obtain the exact relationship $t = 2\sqrt{x}$. The program `ex26ch2.m` solves this problem.

The values of t calculated by the solver `ode45` and the corresponding exact values of t produced by the program agree to four significant figures. The program `ex26ch2.m` may be easily modified to solve each of the other two problems.

Solution for Exercise 2.27. To see the effect of clustering, experiment with values between $k = 0.30$ and $k = 0.35$. For $k = 0.32$ the ball makes it off the ramp while for $k = 0.31$ the bounce times cluster before the ball makes it off the ramp. Use the zoom feature from the tools menu to to see the bounces more clearly.

Solution for Exercise 2.28. The program `ex28ch2.m` includes the wall at the end of the ramp. The program generates figures which show the path of the ball for $k = 0.70$ and the path for $k = 0.35$.

Solution for Exercise 2.29. The program `ex29ch2.m` is a straightforward modification of `dfs.m`. When invoked with

```
 >> ex29ch2('cos(t)*y',[0 12 -6 6])
```

the program behaves just like `dfs` when it is invoked with

```
 >> dfs('cos(t)*y',[0 12 -6 6])
```

2.3.2 ODEs Involving a Mass Matrix

Solution for Exercise 2.30. The IVP is solved in a straightforward way with `ode15s` by the program `ex30ch2.m`. Changing the name of the solver to `ode45` and running the program again shows that it is not expensive to solve the IVP with a code based on explicit Runge–Kutta formulas; hence the IVP is at most modestly stiff.

Solution for Exercise 2.31. The following edited M–file shows only the necessary modifications for this problem. Tracing the displacement of the heavier end of the baton as displayed in the figure generated by the program shows that the baton behaves as one would expect.

```
function ex31ch2
%m1 = 0.1;
%m2 = 0.1;
%L = 1;
m1 = 0.1;
m2 = 0.5;
L = 2;
%title('Original baton problem');
title('Modified baton problem');
%axis([0 22 0 25])
axis([0 30 0 25])
```

Solution for Exercise 2.32. The program `ex32ch2.m` is a straightforward modification of the program `ch2ex4.m` that solves this problem.

Solution for Exercise 2.33. The program `ex33ch2.m` does the computations for this exercise in a straightforward way. The numerical solution of the linear model agrees to graphical accuracy with the analytical solution. The numerical solutions of the linear and nonlinear models are quite close. In particular, the times at which the terminal events occur are quite close:

```
 >> ex33ch2
 Linear model:    lower pallet lost at t = 0.0242.
 Nonlinear model: lower pallet lost at t = 0.0240.
```

2.3.3 Large Systems and the Method of Lines

Solution for Exercise 2.34. When this was done in the most straightforward way by measuring the time with

```
 >> tic, ch2ex6, toc
```

and changing only `ode15s` in the program to `ode23`, the run time on one computer was actually reduced from 5.72s to 4.67s, showing this evidently is not a stiff IVP.

Solution for Exercise 2.35. All that is necessary is to follow the definition of S in `ch2ex6.m` with

```
 S(1,1) = 1;
 S(1,N) = 1;
```

The periodic boundary condition is taken into account by following the definition of `dvdt` with

```
dvdt(1) = c_h(1)*(v(1) - v(end));
```

Solution for Exercise 2.36. The program `ex36ch2.m` implements the requested algorithm. The execution time with the stiff solver is longer by a factor of about 5; and that the number of failed steps for the stiff solver is greater by a factor of about 4.

2.3.4 Singularities

Solution for Exercise 2.37. The program `ex37ach2.m` may be used to verify the coefficients in the expansion.

To see how the coefficients arise, we can proceed as follows. We have

$$\begin{aligned} y &= 1 + \alpha x^2 + O\left(x^4\right) \\ y^n &= 1 + n\alpha x^2 + O\left(x^4\right) \\ y' &= 2\alpha x + O\left(x^3\right) \\ y'' &= 2\alpha + O\left(x^2\right) \end{aligned}$$

where we have used a binomial expansion for y^n. Substituting these into the ODE $y'' + \frac{2}{x} y' + y^n = 0$ and simplifying yields the equation $0 = (6\alpha + 1) + O\left(x^2\right)$ so that $\alpha = -\frac{1}{6}$. Note that this shows also that $y''(0) = -\frac{1}{3}$. Moving on to the next higher order term and again using a binomial expansion for y^n, we see that

$$\begin{aligned} y &= 1 - \frac{1}{6}x^2 + \beta x^4 + O\left(x^6\right) \\ y^n &= 1 + n\left(-\frac{1}{6}x^2 + \beta x^4\right) + O\left(x^4\right) \\ y' &= -\frac{1}{3}x + 4\beta x^3 + O\left(x^5\right) \\ y'' &= -\frac{1}{3} + 12\beta x^2 + O\left(x^4\right) \end{aligned}$$

Substituting these into the ODE and simplifying as before yields the equation

$$0 = \left(20\beta - \frac{n}{6}\right) x^2 + O\left(x^4\right)$$

so that $\beta = \frac{n}{120}$. To compute the next term we have

$$\begin{aligned} y &= 1 - \frac{1}{6}x^2 + \frac{n}{120}x^4 + \theta x^6 + O\left(x^8\right) \\ y^n &= 1 + n\left(-\frac{1}{6}x^2 + \frac{n}{120}x^4 + \theta x^6\right) + \frac{n(n-1)}{2}\left(-\frac{1}{6}x^2 + \frac{n}{120}x^4 + \theta x^6\right)^2 + O\left(x^8\right) \\ y' &= -\frac{1}{3}x + \frac{n}{30}x^3 + 6\theta x^5 + O\left(x^7\right), \\ y'' &= -\frac{1}{3} + \frac{n}{10}x^2 + 30\theta x^4 + O\left(x^6\right) \end{aligned}$$

Substituting these into the ODE and simplifying leads to the equation

$$0 = \left(42\theta + \frac{n^2}{120} + \frac{n^2 - n}{72}\right) x^4 + O\left(x^6\right)$$

which shows that

$$\theta = \frac{\left(-\frac{n^2}{120} - \frac{n^2 - n}{72}\right)}{42} = \cdots = \frac{5n - 8n^2}{3 \cdot 7!}$$

Hence as claimed,

$$y = 1 - \frac{1}{3!}x^2 + \frac{n}{5!}x^4 + \frac{5n - 8n^2}{3 \cdot 7!}x^6 + O\left(x^8\right)$$

and

$$y' = -\frac{1}{3}x + \frac{4n}{5!}x^3 + \frac{6\left(5n - 8n^2\right)}{3 \cdot 7!}x^5 + O\left(x^7\right)$$

In system form our ODE becomes

$$\begin{aligned} y_1' &= y_2 \\ y_2' &= -\frac{2}{x}y_2 - y_1^n \end{aligned}$$

If we use the first three terms of the series expansion for y to approximate $y_1(0.1) = y(0.1)$ and use the first two terms of the series expansion for y' to approximate $y_2(0.1) = y'(0.1)$, and then estimate the errors using the first neglected terms of each expansion, we see that the magnitude of the errors in y_1 and y_2 are approximately $3.8 \cdot 10^{-9}$ and $2.3 \cdot 10^{-7}$, respectively. If we start the integration in a similar fashion at $x = 0.05$, the approximate errors are $5.98 \cdot 10^{-11}$ and $7.1 \cdot 10^{-9}$, respectively. The program `ex37ach2.m` starts the integration in this fashion and continues the integration until $y = 0$. The program uses small error tolerances to demonstrate that the error in the computed solution is due to the errors in the initial approximations. It solves the problem twice, beginning with $x = 0.1$ in the first case and with $x = 0.05$ in the second case. In both cases, the first value of x for which $y(x) = 0$ is found to be $x = 6.89685$. The program generates a figure which shows the computed solution and its derivative for the choice $x = 0.5$; for the choice $x = 0.1$ the graph is virtually identical.

Sometimes it's possible to get lucky for a problem with a singularity especially one that is removable as in this case. We can try to start the problem at $x = 0$, being careful to define $y''(0) = -\frac{1}{3}$ to avoid the indeterminate form at $x = 0$. The program `ex37bch2.m` gives the correct solution for the problem using this strategy.

Solution for Exercise 2.38. If we assume that $y(x) \sim ax^p$ as $x \to 0$, then $y' \sim pax^{p-1}$ and $y'' \sim p(p-1)ax^{p-2}$. Substituting these expressions into the ODE along with $e^{2x} \sim 1 + 2x \sim 1x^0$ says that to leading order,

$$ax^p \left(p(p-1)ax^{p-2}\right)^2 \sim 1x^0$$

Equating the exponents we find that $p = 4/3$. Equating the coefficients and using this value for p, we find that

$$a = \left(\frac{81}{16}\right)^{1/3}$$

To leading order we see that there is a unique solution $y(x)$ that has the behavior that we assume. The second derivative of this solution is not bounded, which should not be a surprise because $y(0) = 0$ and according to the ODE, $y(x)(y''(x))^2 \to 1$ as $x \to 0$.

With the usual unknowns $(y_1(x), y_2(x)) = (y(x), y'(x))$, the ODE is written as the system

$$\begin{aligned} y_1' &= y_2 \\ y_2' &= \frac{e^x}{\sqrt{y_1}} \end{aligned}$$

After moving away from the singular point with the series, the IVP is solved in a straightforward way by the program `ex38ch2.m`. It is easy to vectorize the evaluation of the series for $y(x)$ and $y'(x)$, so this is done to facilitate computing values for comparison with the numerical solution.

Chapter 3

Boundary Value Problems

3.1 Introduction

3.2 Boundary Value Problems

3.3 Boundary Conditions

3.3.1 Boundary Conditions at Singular Points

Solution for Exercise 3.1. If $y(x) \sim 1 + c(x-1)$ as $x \to 1$, then $y'(x) \sim c$. The result $(1-x^7) \sim 7(1-x)$ follows easily from l'Hospital's rule applied to

$$\lim_{x\to 1} \frac{1-x^7}{1-x}$$

or by passing to the limit in the identity

$$\frac{1-x^7}{1-x} = 1 + x + x^2 + x^3 + x^4 + x^5 + x^6$$

With this,

$$(1-x^7)y' \sim 7(1-x)c$$

and

$$((1-x^7)y')' \sim -7c \sim -\lambda x^7 y \sim -\lambda$$

hence there is a solution of this form with $c = \lambda/7$.

Coding the evaluation of the ODEs as explained in the statement of the problem, the BVP is solved numerically in a straightforward way with the program `ex01ch3.m`. This program accepts guesses for λ from the command line. Some results are

```
>> ex01ch3
Guess for eigenvalue: 10
Computed the eigenvalue 8.72727.
>> ex52ch3
Guess for eigenvalue: 100
Computed the eigenvalue 152.39.
>> ex52ch3
Guess for eigenvalue: 500
Computed the eigenvalue 435.022.
```

```
>> ex52ch3
Guess for eigenvalue: 1000
Computed the eigenvalue 855.492.
>> ex52ch3
Guess for eigenvalue: 5000
Computed the eigenvalue 2110.51.
>> ex52ch3
Guess for eigenvalue: 10000
Computed the eigenvalue 5025.87.
```

The first three values are in good agreement with the results reported by Scott.

Solution for Exercise 3.2. The boundary condition at $x = 0$ tells us that $y(0) = \alpha = 10^{-1}$. As $x \to 0$, $y^3(x) \sim \alpha^3 = 10^{-3}, y'(x) \sim \beta p\, x^{\beta-1} + \gamma, y'' \sim \beta(\beta-1)p\, x^{\beta-2}$. Substituting into the ODE, we want

$$2\beta(\beta-1)p\, x^{\beta-1} + \beta p\, x^{\beta-1} + \gamma \sim 10^{-3}$$

With our assumption that $\beta < 1$, we need a value of β for which the coefficient of $x^{\beta-1}$ vanishes. This determines β to be 0.5. Equating the constant terms, we then find that $\gamma = 10^{-3}$.

Solution for Exercise 3.3. This problem is solved in a straightforward way with the program `ex03ch3.m`. It passes the parameters through the call list of `bvp4c`.

Solution for Exercise 3.4. This problem may be solved using the program `ex04ch3.m`.

3.3.2 Boundary Conditions at Infinity

Solution for Exercise 3.5. It is easily shown that the general solution has the form stated and that for the boundary conditions

$$y(0) = 1,\ y(+\infty) = 0,\ y'(+\infty) = 0$$

there is a unique solution given by $A = 0$, $B = 0$ and $C = 1$.

The last of the boundary conditions

$$y(0) = 1,\ y'(0) = 1,\ y(+\infty) = 0$$

requires that $A = B = 0$. The other two boundary conditions then require that $C = 1$ and $-C = 1$, which is impossible.

The third set of boundary conditions approximate the set

$$y(0) = 1,\ y'(0) = 1,\ y(+\infty) = 0$$

for which there is no solution. So, the problem being solved numerically is close to being not well–posed. Specifically, the system of linear equations that is to be solved for A, B and C is

$$\begin{aligned} A + B + C &= 1 \\ A + 2B - C &= 1 \\ Ae^{20} + 2Be^{40} - Ce^{-20} &= 0 \end{aligned}$$

Attempting to solve this linear system in MATLAB results in a warning that the matrix is close to singular and provides an estimate of $5 \cdot 10^{17} =$ `1/RCOND` for its condition number.

Solution for Exercise 3.6. Substituting the assumed form of the solution into the ODE and dividing out the common factor $e^{-\lambda x}$ results in

$$\rho\lambda^2(\lambda^2 - \lambda\Omega - \lambda\delta - \rho\lambda e^{-\lambda x} + \Omega\delta + \Omega\rho e^{-\lambda x}) \sim \gamma\rho^2\lambda^2(\Omega - \lambda)e^{-\lambda x}$$

If we now let $x \to \infty$, this reduces to

$$\rho\lambda^2(\lambda^2 - \lambda\Omega - \lambda\delta + \Omega\delta) = \rho\lambda^2(\lambda - \Omega)(\lambda - \delta) = 0$$

Evidently the ODE is satisfied to terms of leading order if $\lambda = \Omega$. When $\lambda = \Omega$, the assumed form has

$$f'(x) \sim -\Omega\rho e^{-\Omega x}$$

With $\rho = -\Omega^{-1}$, we have solutions that satisfy one of the boundary conditions at infinity. The derivative of such a solution also satisfies the other boundary condition at infinity. We see from this that the boundary conditions specified by Murphy are consistent with the behavior of solutions of the ODE.

It is remarkable that substitution into the ODE of the assumed form for $f(x)$ with $\lambda = \Omega$ shows that it is an exact solution for any δ and ρ. We have already seen how to choose ρ so as to satisfy the boundary conditions at infinity. If we now choose $\delta = \Omega^{-1}$ so that

$$f(x) = \Omega^{-1}(1 - e^{-\Omega x})$$

we satisfy $f(0) = 0$, too.

Solution for Exercise 3.7. Because $h(\infty) = 0$, the equation

$$f'' = \beta\left(E - \frac{h}{\lambda}\right)f$$

can be approximated for large z by $f'' = \beta E f$. Solving this approximating ODE for $\beta = 10, E = 1$, we find that

$$f(z) = Ae^{\sqrt{10}z} + Be^{-\sqrt{10}z}$$

for constants A, B. The solution is to have $f(\infty) = 0$, so $A = 0$. From this we see that the solution of the given equation is approximately a multiple of $e^{-\sqrt{10}z}$ for large z. The solution component $h(z)$ is analyzed similarly.

The BVP is solved numerically in a straightforward way with the program `ex07ch3.m` using the suggested values of Z. A constant guess is used for the initial interval and thereafter the solution for one interval is extended to form the guess for the next. From the consistency of the various solutions $f(z)$ that are plotted, it appears that these values of Z are large enough to give graphical accuracy. The computed solutions $f(z)$ are consistent at $z = 0.1$ with the bounds of Ames and Lohner.

Solution for Exercise 3.8. This problem is solved in a straightforward way using the program `ex08ch3.m`. A constant guess of `[0.5; -0.5]` is used for $(w(z), w'(z))$. The simple boundary condition $w(Z) = 0$ is used for $Z = 2, 3, 4$ and the three solutions are plotted as they are computed. It appears that these values of Z are big enough for graphical accuracy.

Solution for Exercise 3.9. Using the MATLAB Symbolic Toolbox, we can substitute the assumed form of the solution at the origin into the differential equation and collect terms with

```
syms x y p b c eqn
y = 1 + p*x + (4/3)*x^(3/2) + b*x^2 + c*x^(5/2);
eqn = collect(x*diff(y,2,'x')^2 - y^3)
```

The output, edited to show only the lowest order terms, is

```
eqn = ... +(15*b*c-4)*x^(3/2)+(15/2*c+4*b^2-3*p)*x+4*b*x^(1/2)
```

From this we see that we must take $b = 0$ to eliminate the lowest order term and then $c = \dfrac{2p}{5}$ to eliminate the next. In this the parameter p remains free. The other analytical calculations of this exercise are straightforward and simple enough to do by hand.

The BVP is solved in a straightforward way using the program `ex09ch3.m`. The solutions on the various intervals are in sufficiently close agreement that we can be confident that the last interval used is long enough to yield a solution to graphical accuracy. The program confirms the value for p reported in the literature.

3.4 Numerical Methods for BVPs

Solution for Exercise 3.10. The general solution of the ODE is

$$y(x) = c_1 \sin(10x) + c_2 \cos(10x)$$

Applying the boundary conditions we find that $c_2 = 1$ and $c_1 = \frac{B-\cos(100)}{\sin(100)}$. Hence $\frac{\partial y}{\partial B} = \frac{\sin(10x)}{\sin(100)}$ and $\left|\frac{\partial y}{\partial B}\right| = \left|\frac{\sin(10x)}{\sin(100)}\right| < 2$. So small changes in B are reflected in changes of no more than about twice the size in the values of the solution $y(x)$.

Solution for Exercise 3.11. By definition

$$\begin{aligned}
\sigma_i &= S_i e_i + R_i e_{i+1} \\
&= S_i[y(x_i) - y_i] + R_i[y(x_{i+1}) - y_{i+1}] \\
&= \left[-\tfrac{2}{h_i} I - J(x_i)\right] y(x_i) + \left[\tfrac{2}{h_i} I - J(x_{i+1})\right] y(x_{i+1}) - [q(x_i) + q(x_{i+1})] \\
&= \tfrac{2}{h_i}[y(x_{i+1}) - y(x_i)] - [J(x_i)y(x_i) + J(x_{i+1})y(x_{i+1})] - [q(x_i) + q(x_{i+1})]
\end{aligned}$$

Now, using Taylor series we have

$$\begin{aligned}
y(x_{i+1}) - y(x_i) &= 2\left[\tfrac{h_i}{2}\frac{y'(x_{i+\frac{1}{2}})}{1!} + \left(\tfrac{h_i}{2}\right)^3 \frac{y'''(x_{i+\frac{1}{2}})}{3!} + \cdots\right] \\
&= h_i y'(x_{i+\frac{1}{2}}) + \tfrac{h_i^3}{24} y'''(x_{i+\frac{1}{2}}) + \cdots
\end{aligned}$$

$$\begin{aligned}
J(x_i)y(x_i) + J(x_{i+1})y(x_{i+1}) &= 2\left[J(x_{i+\frac{1}{2}})y(x_{i+\frac{1}{2}}) + \left(\tfrac{h_i}{2}\right)^2 \frac{[Jy]''(x_{i+\frac{1}{2}})}{2!} + \cdots\right] \\
&= 2J(x_{i+\frac{1}{2}})y(x_{i+\frac{1}{2}}) + \tfrac{h_i^2}{4}[Jy]''(x_{i+\frac{1}{2}}) + \cdots
\end{aligned}$$

and

$$\begin{aligned}
q(x_{i+1}) + q(x_i) &= 2\left[q(x_{i+\frac{1}{2}}) + \left(\tfrac{h_i}{2}\right)^2 \frac{q''(x_{i+\frac{1}{2}})}{2!} + \cdots\right] \\
&= 2q(x_{i+\frac{1}{2}}) + \tfrac{h_i^2}{4} q''(x_{i+\frac{1}{2}}) + \cdots
\end{aligned}$$

So, neglecting higher order terms,

$$\begin{aligned}
\sigma_i = {} & 2[y'(x_{i+\frac{1}{2}}) - J(x_{i+\frac{1}{2}})y(x_{i+\frac{1}{2}}) - q(x_{i+\frac{1}{2}})] \\
& + h_i^2\left[\tfrac{1}{12} y'''(x_{i+\frac{1}{2}}) - \tfrac{1}{4}\left([Jy]''(x_{i+\frac{1}{2}}) + q''(x_{i+\frac{1}{2}})\right)\right]
\end{aligned}$$

Note that the first term is zero from the ODE. Now differentiating the ODE we have

$$y'''(x) = [Jy]''(x) + q''(x)$$

so

$$\sigma_i = h_i^2\left[\frac{1}{12} y'''(x_{i+\frac{1}{2}}) - \frac{1}{4} y'''(x_{i+\frac{1}{2}})\right] = -\frac{1}{6} h_i^2 y'''(x_{i+\frac{1}{2}})$$

Solution for Exercise 3.12. Consider the given formula

$$y_{i+\frac{1}{2}} = y_i + h_i\left[\frac{5}{24} f(x_i, y_i) + \frac{1}{3} f(x_{i+\frac{1}{2}}, y_{i+\frac{1}{2}}) - \frac{1}{24} f(x_{i+1}, y_{i+1})\right]$$

and its counterpart for the negative direction

$$y_{i+\frac{1}{2}} = y_{i+1} - h_i\left[\frac{5}{24} f(x_{i+1}, y_{i+1}) + \frac{1}{3} f(x_{i+\frac{1}{2}}, y_{i+\frac{1}{2}}) - \frac{1}{24} f(x_i, y_i)\right]$$

Averaging these formulas we obtain

$$y_{i+\frac{1}{2}} = \frac{y_i + y_{i+1}}{2} + \frac{h_i}{8}\left[f(x_i, y_i) - f(x_{i+1}, y_{i+1})\right]$$

Substituting for $y_{i+\frac{1}{2}}$ in the formula

$$y_{i+1} = y_i + h_i \left[\frac{1}{6} f(x_i, y_i) + \frac{2}{3} f(x_{i+\frac{1}{2}}, y_{i+\frac{1}{2}}) + \frac{1}{6} f(x_{i+1}, y_{i+1}) \right]$$

gives the desired "condensed" formula

$$y_{i+1} = y_i + \frac{h_i}{6} \left[f(x_i, y_i) + f(x_{i+1}, y_{i+1}) + 4f\left(x_{i+\frac{1}{2}}, \frac{y_i + y_{i+1}}{2} + \frac{h_i}{8} \left[f(x_i, y_i) - f(x_{i+1}, y_{i+1}) \right] \right) \right]$$

Solution for Exercise 3.13. We are to write out the linear system of equations that result when we use the formula

$$y_{i+1} = y_i + \frac{h_i}{6} \left[f(x_i, y_i) + f(x_{i+1}, y_{i+1}) + 4f\left(x_{i+\frac{1}{2}}, \frac{y_i + y_{i+1}}{2} + \frac{h_i}{8} \left[f(x_i, y_i) - f(x_{i+1}, y_{i+1}) \right] \right) \right]$$

for the linear ODE

$$y'(x) = J(x)y(x) + q(x)$$

with linear boundary conditions. The latter are no different than previously considered so we concentrate on the former. Substituting $f(x, y(x)) = J(x)y(x) + q(x)$ throughout the formula we obtain

$$\begin{aligned} y_{i+1} &= y_i + \tfrac{h_i}{6} [J_i y_i + q_i + J_{i+1} y_{i+1} + q_{i+1} \\ &\quad +4 \left(J_{i+\frac{1}{2}} \left[\tfrac{y_i + y_{i+1}}{2} + \tfrac{h_i}{8} \left(J_i y_i - J_{i+1} y_{i+1} + q_i - q_{i+1} \right) \right] + q_{i+\frac{1}{2}} \right)] \end{aligned}$$

where $q_i = q(x_i)$ etc. Reorganizing this equation we get

$$\begin{aligned} y_{i+1} &= y_i + \tfrac{h_i}{6} \left[J_i y_i + 4 J_{i+\frac{1}{2}} \left(\tfrac{y_i + y_{i+1}}{2} \right) + J_{i+1} y_{i+1} + q_i + 4 q_{i+\frac{1}{2}} + q_{i+1} \right] \\ &\quad + \tfrac{h_i^2}{12} J_{i+\frac{1}{2}} \left[J_i y_i - J_{i+1} y_{i+1} + q_i - q_{i+1} \right] \end{aligned}$$

which be written in the form $S_i y_i + R_i y_{i+1} = v_i$ if we set

$$S_i = I + \frac{h_i}{6} \left[J_i + 2 J_{i+\frac{1}{2}} + \frac{h_i}{2} J_{i+\frac{1}{2}} J_i \right]$$

$$R_i = -I + \frac{h_i}{6} \left[J_{i+1} + 2 J_{i+\frac{1}{2}} - \frac{h_i}{2} J_{i+\frac{1}{2}} J_{i+1} \right]$$

and

$$v_i = -\frac{h_i}{6} \left[q_i + 4 q_{i+\frac{1}{2}} + q_{i+1} + \frac{h_i}{2} J_{i+\frac{1}{2}} (q_i - q_{i+1}) \right]$$

Solution for Exercise 3.14. Since $S(x) \in C[a, b]$ and $S(x)$ is a piecewise cubic polynomial, all we need to show is that $S'(x)$ is continuous at the interior nodes $\{x_i\}$. But the collocation conditions $S'(x_i) = f(x_i, S(x_i)) = f(x_i, y_i)$ at each interior node guarantee that $S'(x)$ is continuous at the nodes. So, $S(x) \in C^1[a, b]$.

3.5 Solving BVPs in Matlab

Solution for Exercise 3.15. After multiplying the ODE by $2y'(x)$, we obtain an expression that can be integrated immediately:

$$0 = 2y'(x)[y''(x) + \lambda e^{y(x)}] = \frac{d}{dx} [(y'(x))^2 + 2\lambda (e^{y(x)} - 1)]$$

Evaluating the constant of integration by setting $x = 0$ results in the conservation law.

Adding the following lines to the program `ch3ex1.m`

```
y = Sxint(1,:);
yp = Sxint(2,:);
residual = yp .^2 + 2*(exp(y) - 1) - yp(1)^2;
plot(xint,residual)
```

evaluates the residual of the solution in the conservation law at the 100 equally spaced points of `xint` and plots it. The residual ranges from 0 to $-4 \cdot 10^{-5}$, which is about what we would expect for the given tolerances.

Solution for Exercise 3.16. The problem is solved in a straightforward way using the program `ex16ch3.m`.

Solution for Exercise 3.17. This BVP is solved in a straightforward way using the program `ex17ch3.m`. It plots the solution and displays on the screen

```
>> sol = ex17ch3;
Other authors report y(0) = 0.63678, y(1) = 0.45759.
Values computed are  y(0) = 0.63679, y(1) = 0.45759.
```

Solution for Exercise 3.18. This BVP is solved in a straightforward way with the program `ex18ch3.m`. It uses the constant guess $(1,1,1,1)$ for the four solution components $(\phi(s), \phi'(s), x(s), y(s))$.

Solution for Exercise 3.19. Let $y_1(x) = y(x)$, $y_2(x) = y'(x)$, $y_3(x) = \int_0^x \dfrac{dt}{\sqrt{1+\epsilon^2 y_1^2(t)}}$ and $y_4(x) = H$. These variables satisfy the first order system of ODEs

$$\begin{aligned} y_1' &= y_2 \\ y_2' &= -\omega^2 \left(\frac{1-\alpha^2}{y_4} \frac{1}{\sqrt{1+\epsilon^2 y_1^2}} + \alpha^2 \right) y_1 \\ y_3' &= \frac{1}{\sqrt{1+\epsilon^2 y_1^2}} \\ y_4' &= 0 \end{aligned}$$

The corresponding boundary conditions are

$$\begin{aligned} y_1(0) &= 1 \\ y_2(0) &= 0 \\ y_2(1) &= 0 \\ y_4(1) &= \frac{1}{\alpha^2}\left[1 - (1-\alpha^2) y_3(1)\right] \end{aligned}$$

When $\epsilon = 0$, it is easily seen first that $H = 1$ and then that the ODE reduces to

$$y'' + \omega^2 y = 0$$

It is straightforward to verify that $y(x) = \cos(\pi x)$ satisfies this ODE when $\omega = \pi$ and that it satisfies the three boundary conditions.

This BVP is solved in a straightforward way using the program `ex19ch3.m`. It plots the solution and displays on the screen

```
>> sol = ex19ch3;
Computed omega = 3.95645.
Computed y(0.5) is 5.65373e-012.
Computed y(1.0) is -1.
```

Solution for Exercise 3.20. The program `ex20ch3.m` solves the problem with a single solution and confirms the initial values reported by Kubiček *et alia.* It may be modified in a straightforward way to obtain the three solutions for the other choice of parameters.

Solution for Exercise 3.21. With the variables specified, the BVP is immediately written as the first order system of ODEs

$$\begin{aligned}\frac{dy_1}{dt} &= y_2 \\ \frac{dy_2}{dt} &= \frac{y_2(1+y_2^2)}{t(3y_2^2-1)}\end{aligned}$$

with boundary conditions

$$y_1(t_0) = 0, \qquad y_2(t_0) = 1, \qquad y_1(1) = 1$$

It is natural to introduce

$$y_3(t) = t^2 + \int_t^1 \frac{2\tau}{(y'(\tau))^2 + 1}\, d\tau$$

for the computation of k. With it, we obviously have $y_3(1) = 1$ and $y_3(t_0) = k$. Differentiating this expression, we obtain the ODE

$$\frac{dy_3}{dt} = 2t - \frac{2t}{y_2^2 + 1}$$

The new independent variable

$$x = \frac{t - t_0}{1 - t_0}$$

obviously ranges from 0 to 1 as t ranges from t_0 to 1. It is then easy to change variables with

$$\frac{d}{dt} = \frac{d}{dx}\frac{dx}{dt} = \frac{1}{1-t_0}\frac{d}{dx}$$

and so arrive at a BVP on the interval $[0, 1]$ involving the unknown parameter t_0.

The BVP is solved in a straightforward way by the program `ex21ch3.m` using the suggested guesses. It reports that

```
 t0 is about  0.3510.
 The reduced drag coefficient is about 0.3748.
```

in good agreement with the values reported by Edwards. After solving in terms of the independent variable x, the corresponding values in t are obtained with

```
t = t0 + (1 - t0)*sol.x
```

In addition to plotting $y(t)$, the program forms and plots the nose cone as a surface.

Solution for Exercise 3.22. This problem is solved in a straightforward way using the program `ex22ch3.m` which reports that

```
 Initial slope of 2 is approximated by 1.99998.
```

Solution for Exercise 3.23. For $\beta = 0.5$, the BVP is solved in a straightforward way by the program `ex23ch3.m`. The program reports to the screen that

```
 Reference value: f''(0) = 0.92768
 Computed value:  f''(0) = 0.92768
```

Solution for Exercise 3.24. This problem is solved in a straightforward way using the program `ex24ch3.m`. A constant guess of `[0; 1; 0]` is used for $(G(x), F(x), F'(x))$. The simple boundary condition $F(X) = 0$ is applied at each of $X = 2$, $X = 3$ and $X = 4$ and the three solutions are plotted as they are computed. It appears that these values of X are large enough to deliver graphical accuracy. The program also reports that

```
  F'(0)   infinity
 1.74376      2
 1.74313      3
 1.74313      4
```

Apparently these values of X are large enough to determine $F'(0)$ to at least the accuracy we expect of the default tolerances that we used in solving the BVPs.

Solution for Exercise 3.25. This is just a matter of removing comments from the program `ch3ex5.m` as discussed in the text and running the various versions with

```
>> tic, ch3ex5, toc
```

Solution for Exercise 3.26. With the hints, the analytical part of this exercise is straightforward. Replacing the guess function of program `ch3ex5.m` with

```
function v = guess(z)
global c alpha beta infty
v = [ 1 ./ (1 + exp(z/c))
     - exp(z/c) ./ (c*(1 + exp(z/c)) .^ 2) ];
```

reduced the run time of the *second* invocation of

```
>> tic, ch3ex5, toc
```

from 2.31s to 2.04s.

Vectorizing the guess function has no effect in this computation, but it is useful when comparing the approximation to the computed solution. To see this, for $c = 5$ follow the computations of `ch3ex5.m` with

```
c = 5;
alpha = (-c + sqrt(c^2 + 4))/2;
beta  = (-c + sqrt(c^2 - 4))/2;
infty = 10*c;
solinit = bvpinit(linspace(-infty,infty,20),@guess);
sol = bvp4c(@ode,@bc,solinit,options);
figure
g = guess(sol.x);
plot(sol.x,sol.y(1,:),sol.x,g(1,:))
legend('Computed','Approximate')
title('Comparison for c = 5.')
```

Even for c this small, the approximation is quite close to the computed solution over the whole interval.

Solution for Exercise 3.27. The program `ex27ch3.m` that is provided runs on one computer in 5.60s when vectorization is not used and 4.39s when it is.

Solution for Exercise 3.28. This is just a matter of seeing for yourself what happens when the experiments of the example are done.

Solution for Exercise 3.29. This problem is solved in a straightforward way using the program `ex29ch3.m`. Providing analytical partial derivatives and vectorizing the evaluation of the ODE reduces the run time on one computer from 10.66s to 2.41s, a considerable improvement. The program also reports that

```
Lambda = 1.000199e+000.
```

Solution for Exercise 3.30. The program `ex30ch3.m` solves this BVP by continuation in the manner outlined. It displays the solution at each step of the continuation. The solution for $\beta_0 = 10$ is a poor approximation to the solution for $\beta_0 = 1575$, but it is good enough to start off a successful continuation.

Chapter 4

Delay Differential Equations

4.1 Introduction

4.2 Delay Differential Equations

Solution for Exercise 4.1.

- Denote the solution for $1 \le t \le 2$ by $y_1(t)$ and the solution for $2 \le t \le 3$ by $y_2(t)$. Since $y(t) = 1$ for $t \le 1$, we have
$$y_1'(t) = 1 + y_1(t)$$
the solution of which is
$$y_1(t) = -1 + Ce^t$$
Continuity of the solution at $t = 1$ requires $y_1(1) = 1$ so that $C = 2e^{-1}$. Thus
$$y_1(t) = -1 + 2e^{t-1}$$
The first derivative is discontinuous since $y_1'(1) = 2 \ne 0 = y'(1)$. For $2 \le t \le 3$ we have
$$y_2'(t) = [1 + y_2(t)]\, y_1(t-1) = [1 + y_2(t)] \left[-1 + 2e^{t-2}\right]$$
Upon dividing by $1 + y_2(t)$ and integrating both sides of this equation, we obtain
$$y_2(t) = -1 + Ce^{-t+2e^{t-2}}$$
Continuity of the solution requires $y_2(2) = y_1(2)$ so that $C = 2e$. Thus
$$y_2(t) = -1 + 2e^{1-t+2e^{t-2}}$$
The first derivative is continuous since $y_1'(2) = y_2'(2) = 2e$. The second derivative is not continuous since $y_1''(2) = 2e \ne 6e = y_2''(2)$.

- Denote the solution for $1 \le t \le 2$ by $y_1(t)$. Since $y(t) = 1$ for $t \le 1$, we have
$$y_1'(t) = 1 + y_1(t)$$
the solution of which is $y_1(t) = -1 + Ce^t$. Continuity of the solution requires $y_1(1) = 1$ so that $C = 2e^{-1}$. Thus
$$y_1(t) = -1 + 2e^{t-1}$$
The first derivative is discontinuous since $y_1'(1) = 2 \ne 0 = y'(1)$. Denote the solution for $2 \le t \le 4$ by $y_2(t)$. We have
$$\frac{y_2'(t)}{1 + y_2(t)} = y(t/2) = -1 + 2e^{t/2-1}$$

the solution of which is

$$y_2(t) = -1 + Ce^{-t+4e^{t/2-1}}$$

Continuity of the solution requires $y_2(2) = y_1(2)$ so that $C = 2e^{-1}$. Thus

$$y_2(t) = -1 + 2e^{-1-t+4e^{t/2-1}}$$

The first derivative is continuous since $y_1'(2) = y_2'(2) = 2e$. The second derivative is discontinuous since $y_2''(2) = 4e \neq 2e = y_1''(2)$. In general, $y^{(k)}$ will be discontinuous at $t = 2^{k-1}$.

- For $1 \le t \le 2$ we have

  $$y'(t) = 1 + y^2(t)$$

  hence

  $$y(t) = \tan(t + C)$$

  Continuity of the solution requires $y(1) = 1$ so that $C = \frac{\pi}{4} - 1$. The solution is infinite when $t + \frac{\pi}{4} - 1 = \frac{\pi}{2}$, that is, when $t = 1 + \frac{\pi}{4} < 2$.

- The solution of the IVP for the ODE is found to be

  $$y(t) = \frac{1}{1-t}$$

  which extends only as far as $t = 1$. Let $y_k(t)$ be the solution of the DDE on $[k\tau, (k+1)\tau]$. For $k = 0$,

  $$y_0'(t) = y_0(t)$$

  with initial value $y_0(0) = 1$, so $y_0(t) = e^t$, which exists and is continuous on all of $[0, \tau]$. Suppose now that $y_k(t)$ exists and is continuous on all of $[k\tau, (k+1)\tau]$. Then $y_{k+1}(t)$ is the solution of the ODE

  $$y_{k+1}'(t) = y_{k+1}(t)y_k(t-\tau)$$

  with initial value $y_{k+1}((k+1)\tau) = y_k((k+1)\tau)$. It is easy to see that

  $$y_{k+1}(t) = y_k((k+1)\tau)e^{\int_{(k+1)\tau}^{t} y_k(s-\tau)ds}$$

  With our assumptions about $y_k(t)$, this solution exists and is continuous on all of $[(k+1)\tau, (k+2)\tau]$. By induction, the solution of the DDE exists for all $t \ge 0$, no matter what the value of $\tau > 0$.

- It is easy to verify that $y = A\sin(t) + B\cos(t)$ is a solution of $y'(t) = -y\left(t - \frac{\pi}{2}\right)$ for all values of A and B using the facts that $\sin\left(t - \frac{\pi}{2}\right) = -\cos(t)$ and $\cos\left(t - \frac{\pi}{2}\right) = \sin(t)$.

4.3 Numerical Methods for DDEs

4.4 Solving DDEs in Matlab

Solution for Exercise 4.2. The program `ex02ch4.m` solves the DDE in a straightforward way as a function with α, b, c as input arguments. It plots the solution and the non–trivial equilibrium solution. From a command line prompt, the following command will execute the program for the given parameters.

```
>> ex02ch4(0.8,2,1)
```

Solution for Exercise 4.3. The programs `ex03ach4.m`, `ex03bch4.m`, and `ex03cch4.m` solve this problem for the three types of models.

Solution for Exercise 4.4. The program `ex04ch4.m` solves the DDE in a straightforward way and plots all of the solution components.

Solution for Exercise 4.5. The program `ex05ch4.m` solves the DDE in a straightforward way and plots the solution in the phase plane.

Solution for Exercise 4.6. Setting the derivatives to zero in the two differential equations leads to

$$\begin{aligned} 0 &= -x_s\lambda_s + Gy_s \\ n &= x_s + y_s \end{aligned}$$

from which the stated expressions for x_s and y_s are immediate. For "large" t, the functions $x(t), y(t), \lambda(t)$ have their steady state values in the integrals defining $\lambda(t)$. Using this and the fact that $x_s\lambda_s = Gy_s$, we have

$$\begin{aligned} \lambda_s &= \frac{c}{n}\left\{x_s\lambda_s \int_{t-D}^{t}\int_{s}^{t} e^{-G(t-w)}dwds + y_s\int_{t-D}^{t} e^{-G(t-s)}ds\right\} \\ &= \frac{c}{n}\left\{Gy_s\int_{t-D}^{t}\frac{1}{G}\left[1-e^{-G(t-s)}\right]ds + y_s\int_{t-D}^{t} e^{-G(t-s)}ds\right\} \\ &= \frac{c}{n}y_s D \end{aligned}$$

Clearly we can obtain a steady state solution with $y_s = 0$, $\lambda_s = 0$ and $x_s = n$. Expressing y_s here in terms of n and x_s and employing a little manipulation results in the stated expression for the non–zero value of λ_s. The program `ex06ch4.m` solves this DDE in a straightforward way and displays to the screen

```
>> ex06ch4;

For G =  0.1,
Analytical x_s =  4.0, y_s = 96.0, lambda_s =  2.4.
Computed   x_s =  4.0, y_s = 96.0, lambda_s =  2.4.

For G =  1.0,
Analytical x_s = 40.0, y_s = 60.0, lambda_s =  1.5.
Computed   x_s = 39.9, y_s = 60.1, lambda_s =  1.5.

For G =  2.0,
Analytical x_s = 80.0, y_s = 20.0, lambda_s =  0.5.
Computed   x_s = 80.1, y_s = 19.9, lambda_s =  0.5.
```

Solution for Exercise 4.7. The program `ex07ch4.m` solves the DDE in a straightforward way and plots both components of the solution.

Solution for Exercise 4.8. The program `ex08ch4.m` solves this DDE in a straightforward way, plots $y(t)$ and $I(t)$, and displays to the screen

```
>> ex08ch4
DDE23 computed      y(10) =  0.06301983001.
Reference solution y(10) =  0.06302089869.
```

Solution for Exercise 4.9. The program `ex09ch4.m` solves this DDE in a straightforward way and plots the four scaled solution components.

Solution for Exercise 4.10. The program `ex10ach4.m` solves both DDEs in a straightforward way and plots the solutions in different figures. The program `ex10bch4.m` is a straightforward modification of this program along the lines indicated that solves the DDE and plots its solution.

Solution for Exercise 4.11. In steady state, $y_2'(t) = 0$, which leads immediately to $y_{2,0} = \dfrac{y_{1,0} + a}{b}$. Using this in the other equation for the steady state, $y_1'(t) = 0$, we find that $y_{1,0}$ must satisfy the algebraic equation

$$-\frac{y^3}{3} + \left(1 - \frac{1}{b}\right) y - \frac{a}{b} = 0$$

After computing all the roots with the MATLAB function `roots`, we find that the only real root is the stated value of $y_{1,0}$ and that it satisfies $y_{1,0}^2 > 1 - rb$. The program `ex11ch4.m` solves the DDEs in both cases and plots the solutions in separate figures.

Solution for Exercise 4.12. The program `ex12ch4.m` solves this problem for all four data sets in the manner suggested in the exercise. Also as suggested there, it uses indexed solution structures for the three delays.

Solution for Exercise 4.13. The programs `ex13ach4.m`, `ex13bch4.m`, and `ex13cch4.m` solve this problem for the three types of models suggested in the exercise.

Solution for Exercise 4.14. The program `ex14ch4.m` solves the DDE in a straightforward way. It plots the solution and displays to the screen

```
>> ex14ch4;
Restart at  14.0.
Restart at  24.9.
Restart at  68.1.
Restart at  75.7.
Restart at 118.9.
```

4.5 Other Kinds of DDEs and Software