25.19 The two differential equations to be solved are

$$\frac{dv}{dt} = g - \frac{c_d}{m}v^2$$

$$\frac{dx}{dt} = -v$$

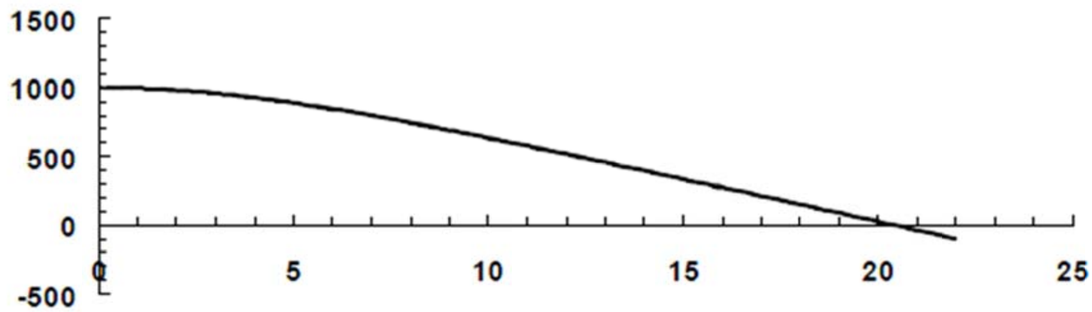
(a) Here are the first few steps of Euler's method with a step size of $h = 0.2$.

t	*x*	*v*	*dx/dt*	*dv/dt*
0	1000	0	0	9.81
0.2	1000	1.962	-1.962	9.800376
0.4	999.6076	3.922075	-3.92208	9.771543
0.6	998.8232	5.876384	-5.87638	9.72367
0.8	997.6479	7.821118	-7.82112	9.657075
1	996.0837	9.752533	-9.75253	9.57222

(b) Here are the results of the first few steps of the 4th-order RK method with a step size of $h = 0.2$.

t	*x*	*v*
0	1000	0
0.2	999.8038	1.961359
0.4	999.2157	3.918875
0.6	998.2368	5.868738
0.8	996.869	7.807195
1	995.1149	9.730582

The results for x of both methods are displayed graphically on the following plots. Because the step size is sufficiently small the results are in close agreement. Both indicate that the parachutist would hit the ground at a little after 20 s. The more accurate 4th-order RK method indicates that the solution reaches the ground between t = 20.2 and 20.4 s.



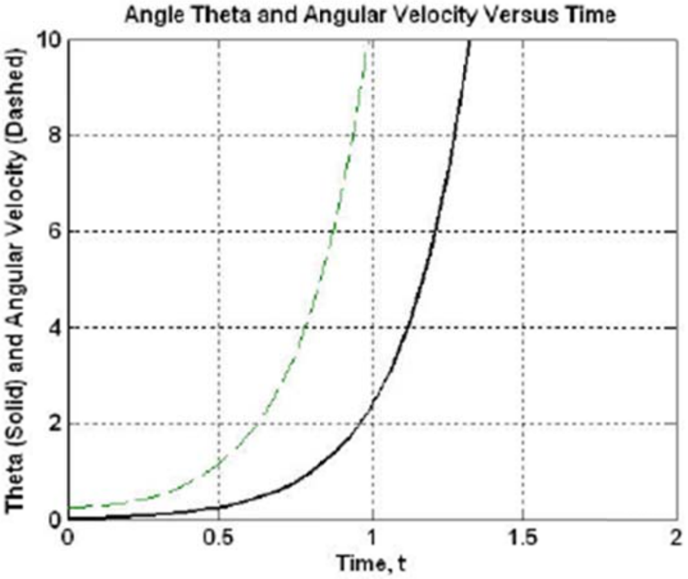
26.13 The second-order equation can be composed into a pair of first-order equations as

$$\frac{d\theta}{dt} = x \qquad \frac{dx}{dt} = \frac{g}{l}\theta$$

We can use MATLAB to solve this system of equations.

```
tspan=[0,5]';
x0=[0,0.25]';
[t,x]=ode45('dxdt',tspan,x0);
plot(t,x(:,1),t,x(:,2),'--')
grid
title('Angle Theta and Angular Velocity Versus Time')
xlabel('Time, t')
ylabel('Theta (Solid) and Angular Velocity (Dashed)')
axis([0 2 0 10])
zoom

function dx=dxdt(t,x)
 dx=[x(2);(9.81/0.5)*x(1)];
```



26.15 (a) Analytic solution:

$$y = \frac{1}{999}\left(1000e^{-x} - e^{-1000x}\right)$$

(b) The second-order differential equation can be expressed as the following pair of first-order ODEs,

$$\frac{dy}{dx} = w$$

$$\frac{dw}{dx} = -1000y - 1001w$$

where $w = y'$. Using the same approach as described in Sec. 26.1, the following simultaneous equations need to be solved to advance each time step,

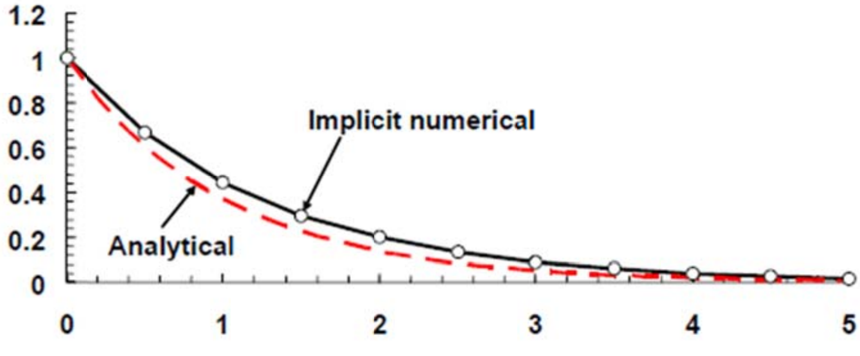
$$y_{i+1} - hw_{i+1} = y_i$$

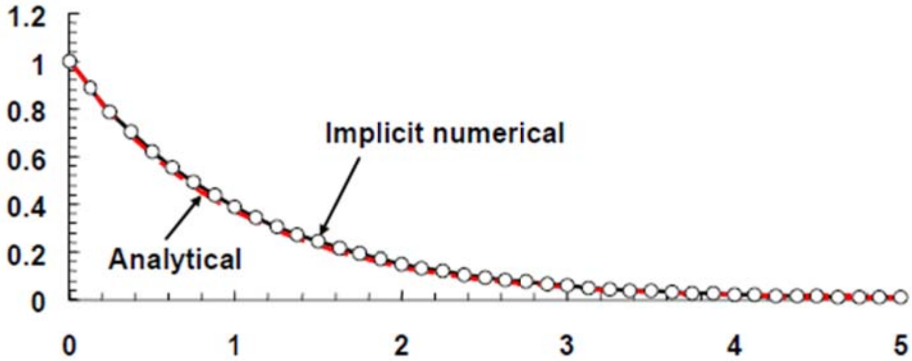
$$1000hy_{i+1} + 1001hw_{i+1} = w_i$$

If these are implemented with a step size of 0.5, the following values are simulated

x	y	w
0	1	0
0.5	0.667332	-0.66534
1	0.444889	-0.44489
1.5	0.296593	-0.29659
2	0.197729	-0.19773
2.5	0.131819	-0.13182
3	0.087879	-0.08788
3.5	0.058586	-0.05859
4	0.039057	-0.03906
4.5	0.026038	-0.02604
5	0.017359	-0.01736

The results for y along with the analytical solution are displayed below:



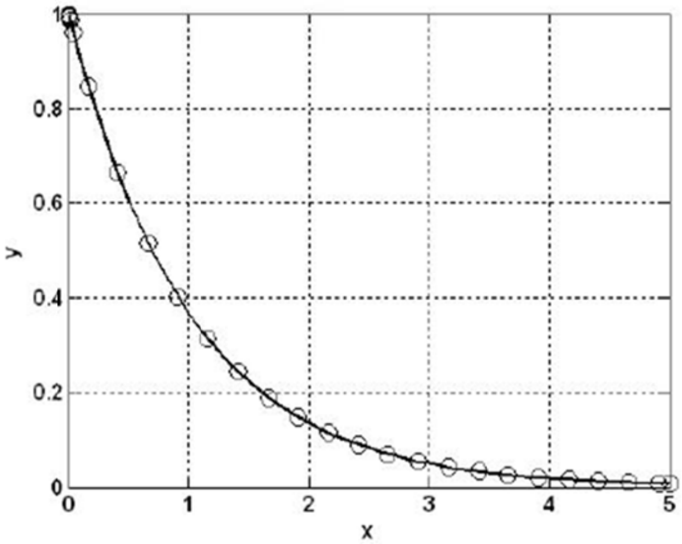


Finally, we can also solve this problem using one of the MATLAB routines expressly designed for stiff systems. To do this, we first develop a function to hold the pair of ODEs,

```
function dy = dydx(x, y)
dy = [y(2);-1000*y(1)-1001*y(2)];
```

Then the following session generates a plot of both the analytical and numerical solutions. As can be seen, the results are indistinguishable.

```
x=[0:.1:5];
y=1/999*(1000*exp(-x)-exp(-1000*x));
xspan=[0 5];
x0=[1 0];
[xx,yy]=ode23s(@dydx,xspan,x0);
plot(x,y,xx,yy(:,1),'o')
grid
xlabel('x')
ylabel('y')
```



27.4 The second-order ODE can be expressed as the following pair of first-order ODEs,

$$\frac{dy}{dx} = z$$
$$\frac{dz}{dx} = \frac{2z + y - x}{7}$$

These can be solved for two guesses for the initial condition of z. For our cases we used −1 and −0.5. We solved the ODEs with the Heun method without iteration using a step size of 0.125. The results are

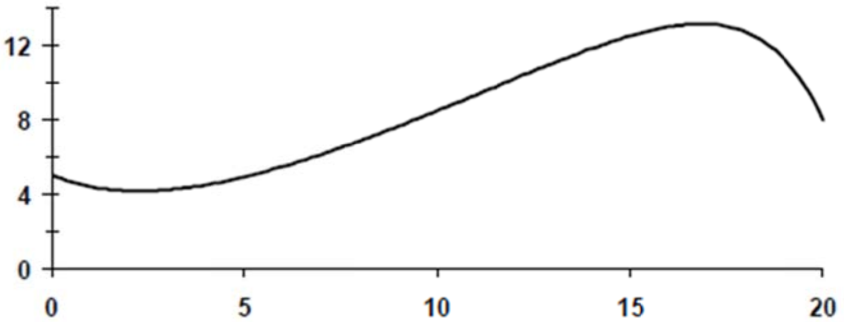
$z(0)$	−1	−0.5
$y(20)$	−11,837.64486	22,712.34615

Clearly, the solution is quite sensitive to the initial conditions. These values can then be used to derive the correct initial condition,

$$z(0) = -1 + \frac{-0.5+1}{22712.34615-(-11837.64486)}(8-(-11837.64486)) = -0.82857239$$

The resulting fit is displayed below:

x	*y*
0	5
2	4.151601
4	4.461229
6	5.456047
8	6.852243
10	8.471474
12	10.17813
14	11.80277
16	12.97942
18	12.69896
20	8



27.5 Centered finite differences can be substituted for the second and first derivatives to give,

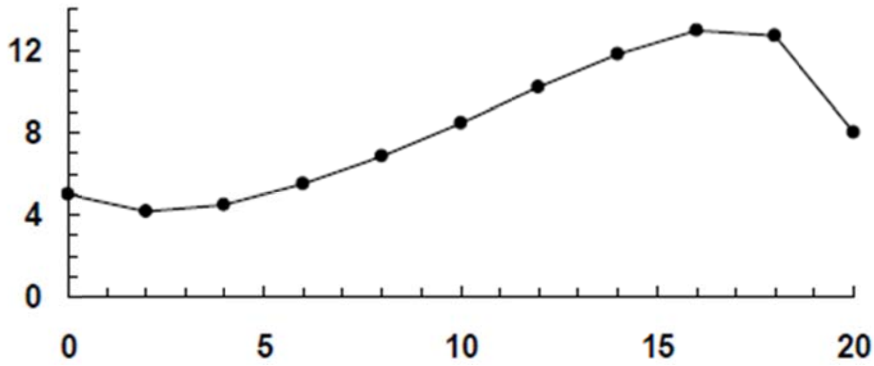
$$7\frac{y_{i+1}-2y_i+y_{i-1}}{\Delta x^2}-2\frac{y_{i+1}-y_{i-1}}{2\Delta x}-y_i+x_i=0$$

or substituting $\Delta x = 2$ and collecting terms yields

$$-2.25y_{i-1}+4.5y_i-1.25y_{i+1}=x_i$$

This equation can be written for each node and solved with methods such as the Tridiagonal solver, the Gauss-Seidel method or LU Decomposition. The following solution was computed using Excel's Minverse and Mmult functions:

x	*y*
0	5
2	4.199592
4	4.518531
6	5.507445
8	6.893447
10	8.503007
12	10.20262
14	11.82402
16	13.00176
18	12.7231
20	8



27.27 (a) The exact solution is

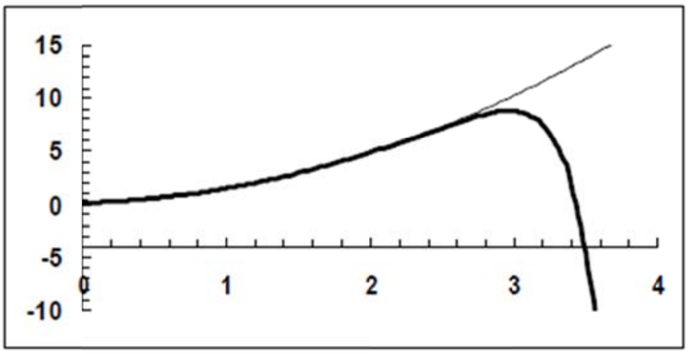
$$y = Ae^{5t} + t^2 + 0.4t + 0.08$$

If the initial condition at $t = 0$ is 0.8, $A = 0$,

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

Note that even though the choice of the initial condition removes the positive exponential terms, it still lurks in the background. Very tiny round off errors in the numerical solutions bring it to the fore. Hence all of the following solutions eventually diverge from the analytical solution.

(b) 4[th] order RK. The plot shows the numerical solution (bold line) along with the exact solution (fine line).



(c)
```
function yp=dy(t,y)
yp=5*(y-t^2);
>> tspan=[0,5];
>> y0=0.08;
>> [t,y]=ode45('dy1',tspan,y0);
```

(d)
```
>> [t,y]=ode23S('dy1',tspan,y0);
```

(e)
```
>> [t,y]=ode23TB('dy1',tspan,y0);
```