CHAPTER 25

25.1 (a) The analytical solution can be derived by separation of variables

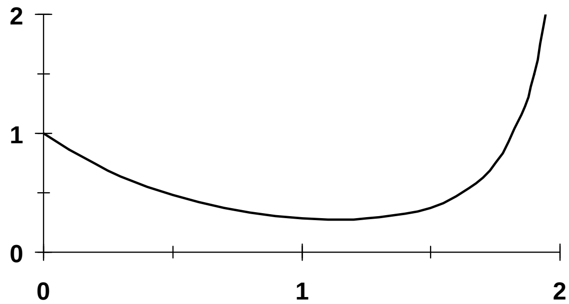
$$\int \frac{dy}{y} = \int t^3 - 1.5 \, dt$$

$$\ln y = \frac{t^4}{4} - 1.5t + C$$

Substituting the initial conditions yields $C = 0$. Taking the exponential gives the final result

$$y = e^{\frac{t^4}{4} - 1.5}$$

The result can be plotted as



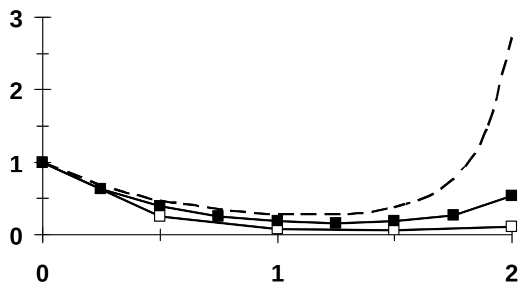
(b) Euler's method with $h = 0.5$

t	*y*	*dy/dt*
0	1	-1.5
0.5	0.25	-0.34375
1	0.078125	-0.03906
1.5	0.058594	0.109863
2	0.113525	0.737915

Euler's method with $h = 0.25$ gives

t	*y*	*dy/dt*
0	1	-1.5
0.25	0.625	-0.92773
0.5	0.393066	-0.54047
0.75	0.25795	-0.2781
1	0.188424	-0.09421
1.25	0.164871	0.074707
1.5	0.183548	0.344153
1.75	0.269586	1.040434
2	0.529695	3.443016

The results can be plotted along with the analytical solution as



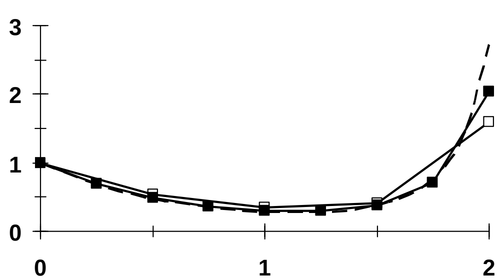
(c) The midpoint method with $h = 0.5$

t	*y*	*dy/dt*	y_m	*dy/dt-mid*
0	1	-1.5	0.25	0.625
0.5	0.536133	-0.73718	0.75	0.351837
1	0.346471	-0.17324	1.25	0.303162
1.5	0.415156	0.778417	1.75	0.60976
2	1.591802	10.34671	2.25	4.17848

with $h = 0.25$ gives

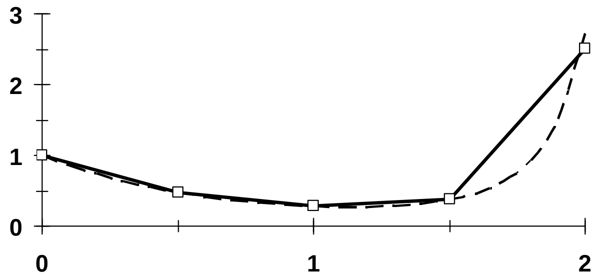
t	*y*	*dy/dt*	y_m	*dy/dt-mid*
0	1	-1.5	0.125	0.8125
0.25	0.695709	-1.03269	0.375	0.566623
0.5	0.490696	-0.67471	0.625	0.406358
0.75	0.363114	-0.39148	0.875	0.314179
1	0.297916	-0.14896	1.125	0.279296
1.25	0.292597	0.132583	1.375	0.30917
1.5	0.377589	0.707979	1.625	0.466086
1.75	0.702802	2.712376	1.875	1.041849
2	2.029023	13.18865	2.125	3.677604

The results can be plotted along with the analytical solution as



(d) The 4th-order RK method with $h = 0.5$ gives

t	*y*	k_1	y_m	k_2	y_m	k_3	y_e	k_4	ϕ
0	1	-1.5000	0.625	-0.9277	0.7681	-1.1401	0.4300	-0.5912	-1.0378
0.5	0.4811	-0.6615	0.3157	-0.3404	0.3960	-0.4269	0.2676	-0.1338	-0.3883
1	0.2869	-0.1435	0.2511	0.1138	0.3154	0.1429	0.3584	0.6720	0.1736
1.5	0.3738	0.7008	0.5489	2.1186	0.9034	3.4866	2.1170	13.7607	4.2786
2	2.5131	16.3350	6.5968	65.2466	18.8247	186.1883	95.6072	1350.4523	311.6095



25.2 (*a*) The analytical solution can be derived by separation of variables

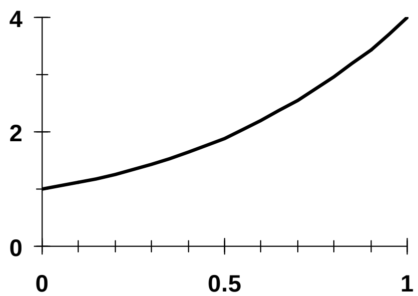
$$\int \frac{dy}{\sqrt{y}} = \int 1 + 2x \; dx$$

$$2\sqrt{y} = x + x^2 + C$$

Substituting the initial conditions yields $C = 2$. Substituting this value and solving for y gives the final result

$$y = \frac{(x^2 + x + 2)^2}{4}$$

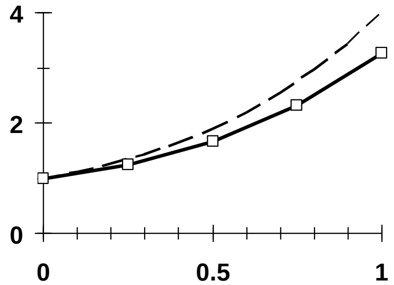
The result can be plotted as



(*b*) Euler's method with $h = 0.25$ gives

x	*y*	*dy/dx*
0	1	1
0.25	1.25	1.677051
0.5	1.669263	2.583999
0.75	2.315263	3.803997
1	3.266262	5.421841

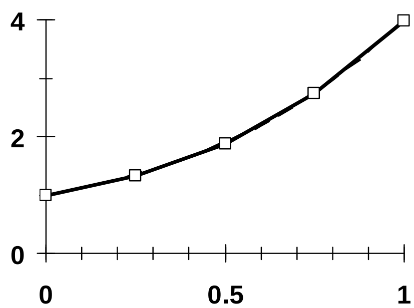
The results can be plotted along with the analytical solution as



(*c*) Heun's method without iteration gives

x	*y*	*dy/dx*	x_{end}	y_{end}	dy_{end}/dx	*dy/dx*-average
0	1	1	0.25	1.25	1.677051	1.338525
0.25	1.334631	1.732894	0.5	1.767855	2.659214	2.196054
0.5	1.883645	2.744919	0.75	2.569875	4.007707	3.376313
0.75	2.727723	4.128955	1	3.759962	5.817186	4.973071
1	3.970991	5.978203	1.25	5.465542	8.182474	7.080339

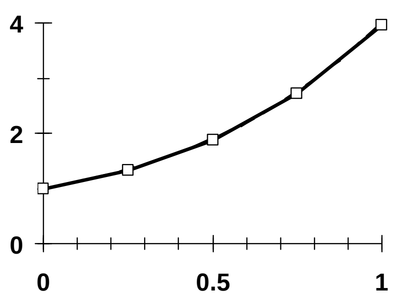
The results along with the analytical solution are displayed below:



(*d*) Ralston method with h = 0.25 gives

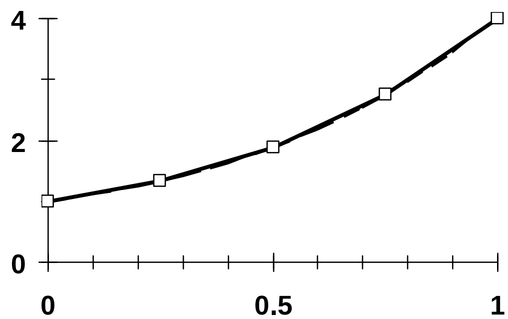
x	*y*	k_1	x+0.75h	$y+0.75k_1h$	k_2	$(1/3)k_1+(2/3)k_2$
0	1	1	0.1875	1.1875	1.498372	1.332248
0.25	1.333062	1.731875	0.4375	1.657788	2.414159	2.186731
0.5	1.879745	2.742076	0.6875	2.393884	3.674643	3.363787
0.75	2.720691	4.12363	0.9375	3.493872	5.373922	4.957158
1	3.959981	5.96991	1.1875	5.079339	7.606369	7.060883

The results along with the analytical solution are displayed below:



(***e***) The 4th-order RK method with h = 0.5 gives

x	y	k_1	y_m	k_2	y_m	k_3	y_e	k_4	ϕ
0	1	1	1.25	1.67705	1.41926	1.78699	1.89350	2.75209	1.78003
0.5	1.89001	2.74956	2.57740	4.01357	2.89341	4.25251	4.01627	6.01219	4.21577
1	3.99784	5.99838	5.49743	8.20631	6.04942	8.60845	8.30206	11.5253	8.52554



25.3 The second-order ODE is transformed into a pair of first-order ODEs as in

$$\frac{dy}{dt} = z \qquad\qquad y(0) = 2$$

$$\frac{dz}{dt} = t - y \qquad\qquad z(0) = 0$$

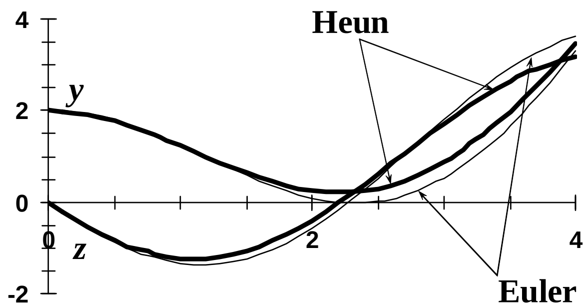
(***a***) The first few steps of Euler's method are

t	*y*	*z*	*dy/dt*	*dz/dt*
0	2	0	0	-2
0.1	2	-0.2	-0.2	-1.9
0.2	1.98	-0.39	-0.39	-1.78
0.3	1.941	-0.568	-0.568	-1.641
0.4	1.8842	-0.7321	-0.7321	-1.4842
0.5	1.81099	-0.88052	-0.88052	-1.31099

(***b***) For Heun (without iterating the corrector) the first few steps are

t	y	z	dy/dt	dz/dt	y_{end}	z_{end}	dy/dt	dz/dt	avg slope
0	2	0	0	-2	2	-0.2	-0.2	-1.9	-0.1
0.1	1.99	-0.195	-0.195	-1.89	1.9705	-0.384	-0.384	-1.7705	-0.2895
0.2	1.96105	-0.37803	-0.37803	-1.76105	1.923248	-0.55413	-0.55413	-1.62325	-0.46608
0.3	1.914442	-0.54724	-0.54724	-1.61444	1.859718	-0.70868	-0.70868	-1.45972	-0.62796
0.4	1.851646	-0.70095	-0.70095	-1.45165	1.781551	-0.84611	-0.84611	-1.28155	-0.77353
0.5	1.774293	-0.83761	-0.83761	-1.27429	1.690532	-0.96504	-0.96504	-1.09053	-0.90132

Both results are plotted below:



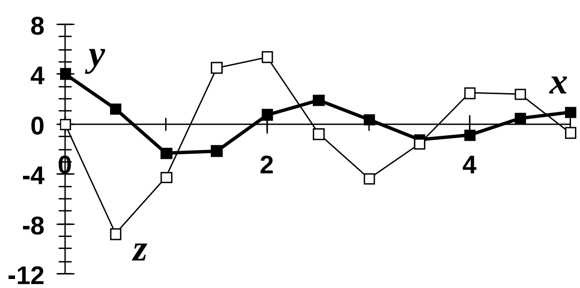
25.4 The second-order ODE is transformed into a pair of first-order ODEs as in

$$\frac{dy}{dx} = z \qquad y(0) = 4$$

$$\frac{dz}{dx} = -0.5z - 7y \qquad z(0) = 0$$

The results for the 4th-order RK method are tabulated and plotted below:

x	y	z	k_{11}	k_{12}	y_{mid}	z_{mid}	k_{21}	k_{22}	y_{mid}	z_{mid}	k_{31}	k_{32}	y_{end}	z_{end}	k_{41}	k_{42}	ϕ_1	ϕ_2
0	4.0000	0.0000	0.00	-28.00	4.00	-7.00	-7.00	-24.50	2.25	-6.13	-6.13	-12.69	0.94	-6.34	-6.34	-3.39	-5.43	-17.63
0.5	1.2839	-8.8138	-8.81	-4.58	-0.92	-9.96	-9.96	11.42	-1.21	-5.96	-5.96	11.42	-1.70	-3.10	-3.10	13.42	-7.29	9.09
1	-2.3623	-4.2706	-4.27	18.67	-3.43	0.40	0.40	23.81	-2.26	1.68	1.68	15.00	-1.52	3.23	3.23	9.03	0.52	17.55
1.5	-2.1025	4.5067	4.51	12.46	-0.98	7.62	7.62	3.02	-0.20	5.26	5.26	-1.25	0.53	3.88	3.88	-5.64	5.69	1.73
2	0.7438	5.3700	5.37	-7.89	2.09	3.40	3.40	-16.30	1.59	1.29	1.29	-11.80	1.39	-0.53	-0.53	-9.47	2.37	-12.26
2.5	1.9291	-0.7605	-0.76	-13.12	1.74	-4.04	-4.04	-10.15	0.92	-3.30	-3.30	-4.78	0.28	-3.15	-3.15	-0.38	-3.10	-7.23
3	0.3798	-4.3750	-4.38	-0.47	-0.71	-4.49	-4.49	7.24	-0.74	-2.56	-2.56	6.49	-0.90	-1.13	-1.13	6.88	-3.27	5.65
3.5	-1.2553	-1.5525	-1.55	9.56	-1.64	0.84	0.84	11.08	-1.05	1.22	1.22	6.71	-0.65	1.80	1.80	3.62	0.73	8.13
4	-0.8916	2.5120	2.51	4.99	-0.26	3.76	3.76	-0.03	0.05	2.50	2.50	-1.59	0.36	1.72	1.72	-3.38	2.79	-0.27
4.5	0.5046	2.3754	2.38	-4.72	1.10	1.20	1.20	-8.29	0.80	0.30	0.30	-5.78	0.66	-0.51	-0.51	-4.34	0.81	-6.20
5	0.9097	-0.7232	-0.72	-6.01	0.73	-2.22	-2.22	-3.99	0.35	-1.72	-1.72	-1.61	0.05	-1.53	-1.53	0.42	-1.69	-2.80



25.5 (*a*) The Heun method without iteration can be implemented as in the following table:

t	y	k_1	y_{end}	k_2	ϕ
0	1	0	1	0.000995	0.000498
0.1	1.00005	0.000995	1.000149	0.007843	0.004419
0.2	1.000492	0.007845	1.001276	0.025841	0.016843
0.3	1.002176	0.025865	1.004762	0.059335	0.0426
0.4	1.006436	0.059434	1.012379	0.11156	0.085497
0.5	1.014986	0.111847	1.02617	0.184731	0.148289
•					•
•					•

•					•
2.9	3.784421	0.051826	3.789604	0.01065	0.031238
3	3.787545	0.010644	3.78861	0.000272	0.005458

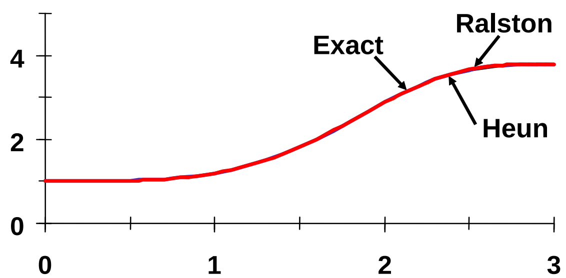
(*b*) The Ralston 2nd order RK method can be implemented as in the following table:

t	*y*	k_1	*yint*	k_2	ϕ
0	1	0	1	0.000421	0.00028
0.1	1.000028	0.000995	1.000103	0.005278	0.003851
0.2	1.000413	0.007845	1.001001	0.020043	0.015977
0.3	1.002011	0.02586	1.00395	0.049332	0.041508
0.4	1.006162	0.059418	1.010618	0.096672	0.084254
0.5	1.014587	0.111803	1.022972	0.164537	0.146959
•					•
•					•
•					•
2.9	3.785066	0.051835	3.788954	0.017276	0.028796
3	3.787946	0.010646	3.788744	0.001116	0.004293

Both methods are displayed on the following plot along with the exact solution,

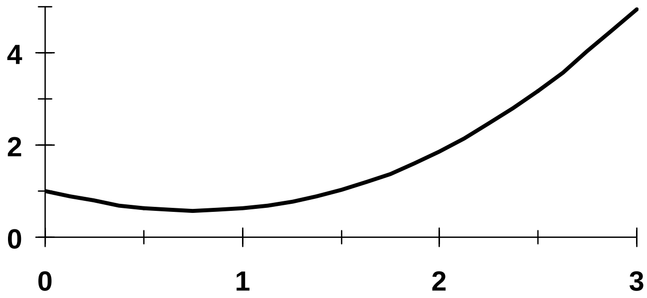
$$y = e^{-\cos t + \frac{\cos^3 t}{3} + \frac{2}{3}}$$

Note that both methods agree closely with the exact solution.



25.6 The solution results are as in the following table and plot:

t	*y*	k_1	k_2	k_3	ϕ
0	1	-1	-0.6875	-0.5625	-0.71875
0.5	0.640625	-0.39063	0.019531	0.144531	-0.02799
1	0.626628	0.373372	0.842529	0.967529	0.78517
1.5	1.019213	1.230787	1.735591	1.860591	1.67229
2	1.855358	2.144642	2.670982	2.795982	2.604092
2.5	3.157404	3.092596	3.631947	3.756947	3.562889
3	4.938848	4.061152	4.608364	4.733364	4.537995



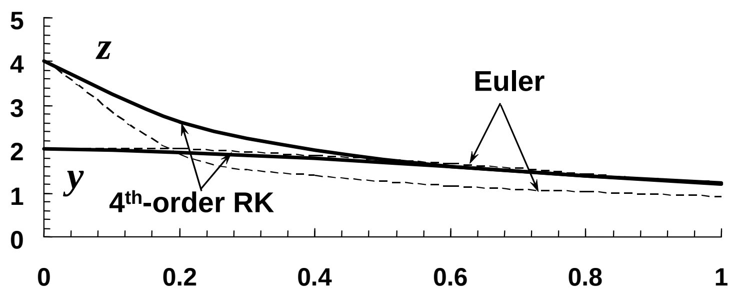
25.7 (*a*) Euler

x	y	z	dy/dx	dz/dx
0	2.0000	4.0000	0.00	-10.67
0.2	2.0000	1.8667	-0.73	-2.32
0.4	1.8550	1.4021	-1.03	-1.22
0.6	1.6492	1.1590	-1.10	-0.74
0.8	1.4286	1.0113	-1.06	-0.49
1	1.2166	0.9139	-0.96	-0.34

(*b*) 4th-order RK

x	y	z	k_{11}	k_{12}	k_{21}	k_{22}	k_{31}	k_{32}	k_{41}	k_{42}	ϕ_1	ϕ_2
0	2.000	4.000	0.000	-10.667	-0.381	-5.736	-0.305	-7.678	-0.603	-3.926	-0.329	-6.903
0.2	1.934	2.619	-0.594	-4.423	-0.786	-2.962	-0.748	-3.338	-0.888	-2.266	-0.758	-3.215
0.4	1.783	1.976	-0.884	-2.321	-0.962	-1.718	-0.947	-1.830	-0.991	-1.377	-0.949	-1.799
0.6	1.593	1.617	-0.990	-1.387	-1.001	-1.087	-0.999	-1.131	-0.989	-0.897	-0.997	-1.120
0.8	1.393	1.392	-0.990	-0.901	-0.963	-0.732	-0.968	-0.753	-0.928	-0.617	-0.963	-0.748
1	1.201	1.243	-0.930	-0.618	-0.884	-0.515	-0.893	-0.526	-0.840	-0.441	-0.887	-0.524

Both methods are plotted on the same graph below. Notice how Euler's method (particularly for z) is not very accurate for this step size. The 4$^{\text{th}}$-order RK is much closer to the exact solution.



25.8 $\dfrac{dy}{dx} = 10e^{-\frac{(x-2)^2}{2(0.075)^2}} - 0.6y$

<u>4$^{\text{th}}$-order RK method:</u>

One step (h = 0.5): $y_1 = 0.3704188$
Two steps (h = 0.25): $y_2 = 0.3704096$
$\Delta_{\text{present}} = -9.119\times10^{-6}$

$$\text{correction} = \frac{\Delta}{15} = -6.08\times10^{-7}$$

$y_2 = 0.370409$

$$\frac{dy}{dx} = -0.3$$

$$y_{\text{scale}} = 0.5 + |0.5(-0.3)| = 0.65$$

$\Delta_{\text{new}} = 0.001(0.65) = 0.00065$

Since $\Delta_{\text{present}} < \Delta_{\text{new}}$, therefore, increase step.

$$h_{\text{new}} = 0.5\left|\frac{0.00065}{9.119\times10^{-6}}\right|^{0.2} = 1.1737$$

25.9 We will look at the first step only

$\Delta_{\text{present}} = y_2 - y_1 = -0.24335$

$$\frac{dy}{dx} = 4e^0 - 0.5(2) = 3$$

$y_{\text{scale}} = 2 + (2(3)) = 8$

$\Delta_{\text{new}} = 0.001(8) = 0.008$

Because $\Delta_{\text{present}} > \Delta_{\text{new}}$, decrease step.

25.10 The calculation of the *k*'s can be summarized in the following table:

	x	*y*	*f(x,y)*
k_1	0	2	3
k_2	0.2	2.6	3.394043
k_3	0.3	2.98866	3.590667
k_4	0.6	4.154161	4.387217
k_5	1	6.251995	5.776166
k_6	0.875	5.511094	5.299464

These can then be used to compute the 4$^{\text{th}}$-order prediction

$$y_1 = 2 + \left(\frac{37}{378}3 + \frac{250}{621}3.590667 + \frac{125}{594}4.387217 + \frac{512}{1771}5.299464\right)1 = 6.194491$$

along with a fifth-order formula:

$$y_1 = 2 + \left(\frac{2825}{27{,}648}3 + \frac{18{,}575}{48{,}384}3.590667 + \frac{13{,}525}{55{,}296}4.387217 + \frac{277}{14{,}336}5.776166 + \frac{1}{4}5.299464\right)1 = 6.194572$$

The error estimate is obtained by subtracting these two equations to give

$$E_a = 6.194572 - 6.194491 = 0.0000805$$

25.11

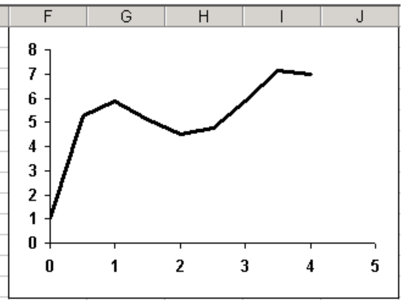
```
Option Explicit

Sub EulerTest()
Dim i As Integer, m As Integer
```

```
Dim xi As Double, yi As Double, xf As Double, dx As Double, xout As Double
Dim xp(200) As Double, yp(200) As Double
'Assign values
yi = 1
xi = 0
xf = 4
dx = 0.5
xout = 0.5
'Perform numerical Integration of ODE
Call ODESolver(xi, yi, xf, dx, xout, xp, yp, m)
'Display results
Sheets("Sheet1").Select
Range("a5:b205").ClearContents
Range("a5").Select
For i = 0 To m
  ActiveCell.Value = xp(i)
  ActiveCell.Offset(0, 1).Select
  ActiveCell.Value = yp(i)
  ActiveCell.Offset(1, -1).Select
Next i
Range("a5").Select
End Sub

Sub ODESolver(xi, yi, xf, dx, xout, xp, yp, m)
'Generate an array that holds the solution
Dim x As Double, y As Double, xend As Double
Dim h As Double
m = 0
xp(m) = xi
yp(m) = yi
x = xi
y = yi
Do       'Print loop
  xend = x + xout
  If (xend > xf) Then xend = xf  'Trim step if increment exceeds end
  h = dx
  Call Integrator(x, y, h, xend)
  m = m + 1
  xp(m) = x
  yp(m) = y
  If (x >= xf) Then Exit Do
Loop
End Sub

Sub Integrator(x, y, h, xend)
Dim ynew As Double
Do       'Calculation loop
  If (xend - x < h) Then h = xend - x  'Trim step if increment exceeds end
  Call Euler(x, y, h, ynew)
  y = ynew
  If (x >= xend) Then Exit Do
Loop
End Sub

Sub Euler(x, y, h, ynew)
Dim dydx As Double
'Implement Euler's method
Call Derivs(x, y, dydx)
ynew = y + dydx * h
x = x + h
End Sub
```

```
Sub Derivs(x, y, dydx)
'Define ODE
dydx = -2 * x ^ 3 + 12 * x ^ 2 - 20 * x + 8.5
End Sub
```

25.12 Example 25.1:

	A	B	C	D	E
4	x	y			
5	0	1		RUN	
6	0.5	5.25			
7	1	5.875			
8	1.5	5.125			
9	2	4.5			
10	2.5	4.75			
11	3	5.875			
12	3.5	7.125			
13	4	7			
14					
15					
16					



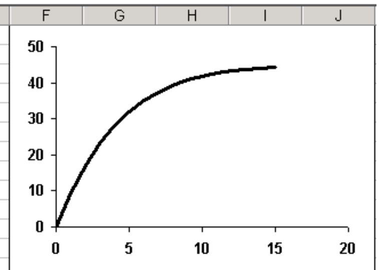
Example 25.4 (nonlinear model). Change time steps and initial conditions to

```
'Assign values
yi = 0
xi = 0
xf = 15
dx = 0.5
xout = 1
```

Change Derivs Sub to

```
Sub Derivs(t, v, dvdt)
'Define ODE
dvdt = 9.8 - 12.5 / 68.1 * (v + 8.3 * (v / 46) ^ 2.2)
End Sub
```

	A	B	C	D	E
4	x	y			
5	0	0		RUN	
6	1	9.34477			
7	2	16.98754			
8	3	23.1603			
9	4	28.08878			
10	5	31.98533			
11	6	35.04111			
12	7	37.42184			
13	8	39.26696			
14	9	40.69109			
15	10	41.78674			
16	11	42.62757			
17	12	43.2716			
18	13	43.76415			
19	14	44.14042			
20	15	44.42762			



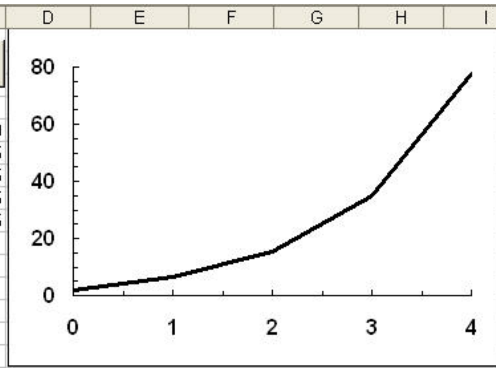
25.13

```
Option Explicit

Sub Heun()
Dim maxit As Integer, es As Double
Dim n As Integer, m As Integer, i As Integer, iter As Integer
Dim xi As Double, xf As Double, yi As Double, h As Double
Dim x As Double, y As Double, y2 As Double, y2old As Double
Dim k1 As Double, k2 As Double, slope As Double
Dim xp(1000) As Double, yp(1000) As Double, itr(1000) As Integer
Dim ea As Double
```

```
maxit = 15: es = 0.0000001
xi = 0: xf = 4: yi = 2
n = 4
h = (xf - xi) / n
x = xi
y = yi
m = 0
xp(m) = x
yp(m) = y
For i = 1 To n
  Call Derivs(x, y, k1)
  y2 = y + k1 * h
  iter = 0
  Do
    y2old = y2
    Call Derivs(x + h, y2, k2)
    slope = (k1 + k2) / 2
    y2 = y + slope * h
    iter = iter + 1
    ea = Abs((y2 - y2old) / y2) * 100
    If ea < es Or iter >= maxit Then Exit Do
  Loop
  m = m + 1
  x = x + h
  xp(m) = x
  yp(m) = y2
  itr(m) = iter
  y = y2
Next i
Sheets("Heun").Select
Range("a5:b1005").ClearContents
Range("a5").Select
For i = 0 To m
  ActiveCell.Value = xp(i)
  ActiveCell.Offset(0, 1).Select
  ActiveCell.Value = yp(i)
  ActiveCell.Offset(0, 1).Select
  ActiveCell.Value = itr(i)
  ActiveCell.Offset(1, -2).Select
Next i
Range("a5").Select
End Sub

Sub Derivs(x, y, dydx)
'Define ODE
dydx = 4 * Exp(0.8 * x) - 0.5 * y
End Sub
```

RUN

x	y	iterations
0	2.0000000	0
1	6.3608655	15
2	15.3022367	15
3	34.7432761	15
4	77.7350962	15

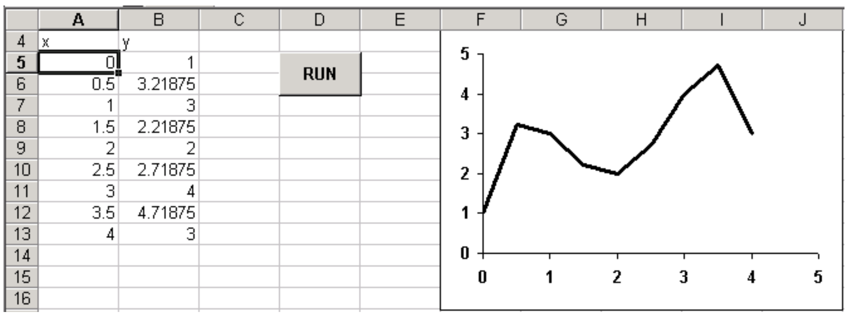


25.14

```
Option Explicit

Sub RK4Test()
Dim i As Integer, m As Integer
Dim xi As Double, yi As Double, xf As Double, dx As Double, xout As Double
Dim xp(200) As Double, yp(200) As Double
'Assign values
yi = 1
xi = 0
xf = 4
dx = 0.5
xout = 0.5
'Perform numerical Integration of ODE
Call ODESolver(xi, yi, xf, dx, xout, xp, yp, m)
'Display results
Sheets("Sheet1").Select
Range("a5:b205").ClearContents
Range("a5").Select
For i = 0 To m
  ActiveCell.Value = xp(i)
  ActiveCell.Offset(0, 1).Select
  ActiveCell.Value = yp(i)
  ActiveCell.Offset(1, -1).Select
Next i
Range("a5").Select
End Sub
Sub ODESolver(xi, yi, xf, dx, xout, xp, yp, m)
'Generate an array that holds the solution
Dim x As Double, y As Double, xend As Double
Dim h As Double
m = 0
xp(m) = xi
yp(m) = yi
x = xi
y = yi
Do       'Print loop
  xend = x + xout
  If (xend > xf) Then xend = xf  'Trim step if increment exceeds end
  h = dx
  Call Integrator(x, y, h, xend)
  m = m + 1
  xp(m) = x
  yp(m) = y
  If (x >= xf) Then Exit Do
Loop
End Sub
Sub Integrator(x, y, h, xend)
Dim ynew As Double
Do       'Calculation loop
  If (xend - x < h) Then h = xend - x  'Trim step if increment exceeds end
  Call RK4(x, y, h, ynew)
  y = ynew
  If (x >= xend) Then Exit Do
Loop
End Sub
Sub RK4(x, y, h, ynew)
'Implement RK4 method
Dim k1 As Double, k2 As Double, k3 As Double, k4 As Double
Dim ym As Double, ye As Double, slope As Double
```

```
Call Derivs(x, y, k1)
ym = y + k1 * h / 2
Call Derivs(x + h / 2, ym, k2)
ym = y + k2 * h / 2
Call Derivs(x + h / 2, ym, k3)
ye = y + k3 * h
Call Derivs(x + h, ye, k4)
slope = (k1 + 2 * (k2 + k3) + k4) / 6
ynew = y + slope * h
x = x + h
End Sub
Sub Derivs(x, y, dydx)
'Define ODE
dydx = -2 * x ^ 3 + 12 * x ^ 2 - 20 * x + 8.5
End Sub
```

<u>Example 25.7*a*:</u>



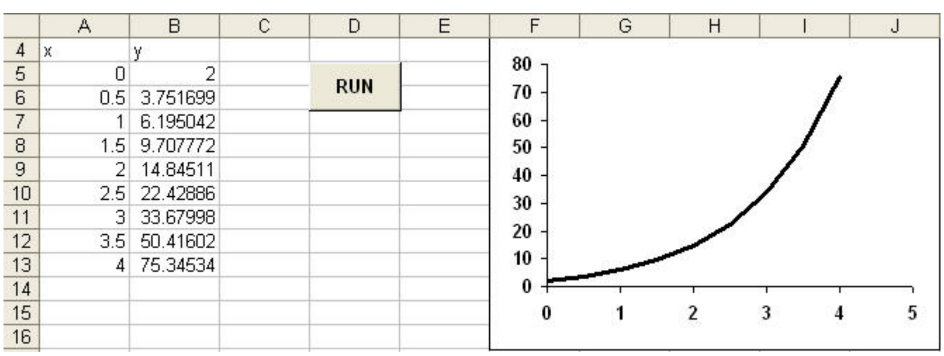
	A	B
4	x	y
5	0	1
6	0.5	3.21875
7	1	3
8	1.5	2.21875
9	2	2
10	2.5	2.71875
11	3	4
12	3.5	4.71875
13	4	3

<u>Example 25.7*b*:</u> Change time steps and initial conditions to

```
'Assign values
yi = 2
xi = 0
xf = 4
dx = 0.5
xout = 0.5
```

Change Derivs Sub to

```
Sub Derivs(x, y, dydx)
'Define ODE
dydx = 4 * Exp(0.8 * x) - 0.5 * y
End Sub
```



	A	B
4	x	y
5	0	2
6	0.5	3.751699
7	1	6.195042
8	1.5	9.707772
9	2	14.84511
10	2.5	22.42886
11	3	33.67998
12	3.5	50.41602
13	4	75.34534

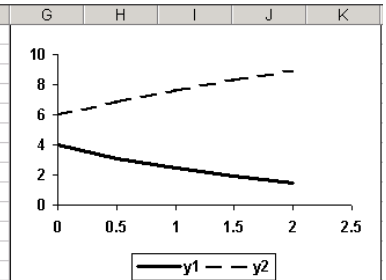
25.15

```
Option Explicit
Sub RK4SysTest()
Dim i As Integer, m As Integer, n As Integer, j As Integer
Dim xi As Double, yi(10) As Double, xf As Double
Dim dx As Double, xout As Double
Dim xp(200) As Double, yp(200, 10) As Double
'Assign values
n = 2
xi = 0
xf = 2
yi(1) = 4
yi(2) = 6
dx = 0.5
xout = 0.5
'Perform numerical Integration of ODE
Call ODESolver(xi, yi, xf, dx, xout, xp, yp, m, n)
'Display results
Sheets("Sheet1").Select
Range("a5:n205").ClearContents
Range("a5").Select
For i = 0 To m
  ActiveCell.Value = xp(i)
  For j = 1 To n
    ActiveCell.Offset(0, 1).Select
    ActiveCell.Value = yp(i, j)
  Next j
  ActiveCell.Offset(1, -n).Select
Next i
Range("a5").Select
End Sub

Sub ODESolver(xi, yi, xf, dx, xout, xp, yp, m, n)
'Generate an array that holds the solution
Dim i As Integer
Dim x As Double, y(10) As Double, xend As Double
Dim h As Double
m = 0
x = xi
'set initial conditions
For i = 1 To n
  y(i) = yi(i)
Next i
'save output values
xp(m) = x
For i = 1 To n
  yp(m, i) = y(i)
Next i
Do       'Print loop
  xend = x + xout
  If (xend > xf) Then xend = xf  'Trim step if increment exceeds end
  h = dx
  Call Integrator(x, y, h, n, xend)
  m = m + 1
  'save output values
  xp(m) = x
  For i = 1 To n
    yp(m, i) = y(i)
  Next i
  If (x >= xf) Then Exit Do
Loop
```

```
End Sub
Sub Integrator(x, y, h, n, xend)
Dim j As Integer
Dim ynew(10) As Double
Do       'Calculation loop
  If (xend - x < h) Then h = xend - x  'Trim step if increment exceeds end
  Call RK4Sys(x, y, h, n, ynew)
  For j = 1 To n
    y(j) = ynew(j)
  Next j
  If (x >= xend) Then Exit Do
Loop
End Sub
Sub RK4Sys(x, y, h, n, ynew)
Dim j As Integer
Dim ym(10) As Double, ye(10) As Double
Dim k1(10) As Double, k2(10) As Double, k3(10) As Double, k4(10) As Double
Dim slope(10)
'Implement RK4 method for systems of ODEs
Call Derivs(x, y, k1)
For j = 1 To n
  ym(j) = y(j) + k1(j) * h / 2
Next j
Call Derivs(x + h / 2, ym, k2)
For j = 1 To n
  ym(j) = y(j) + k2(j) * h / 2
Next j
Call Derivs(x + h / 2, ym, k3)
For j = 1 To n
  ye(j) = y(j) + k3(j) * h
Next j
Call Derivs(x + h, ye, k4)
For j = 1 To n
  slope(j) = (k1(j) + 2 * (k2(j) + k3(j)) + k4(j)) / 6
Next j
For j = 1 To n
  ynew(j) = y(j) + slope(j) * h
Next j
x = x + h
End Sub

Sub Derivs(x, y, dydx)
'Define ODE
dydx(1) = -0.5 * y(1)
dydx(2) = 4 - 0.3 * y(2) - 0.1 * y(1)
End Sub
```

Application to Example 25.10:

	A	B	C	D	E	F
1	Example 25.10					
2						
3						
4	x	y1	y2			
5	0	4	6		RUN	
6	0.5	3.115234	6.85767			
7	1	2.426171	7.632106			
8	1.5	1.889523	8.326886			
9	2	1.471577	8.946865			
10						
11						
12						
13						
14						

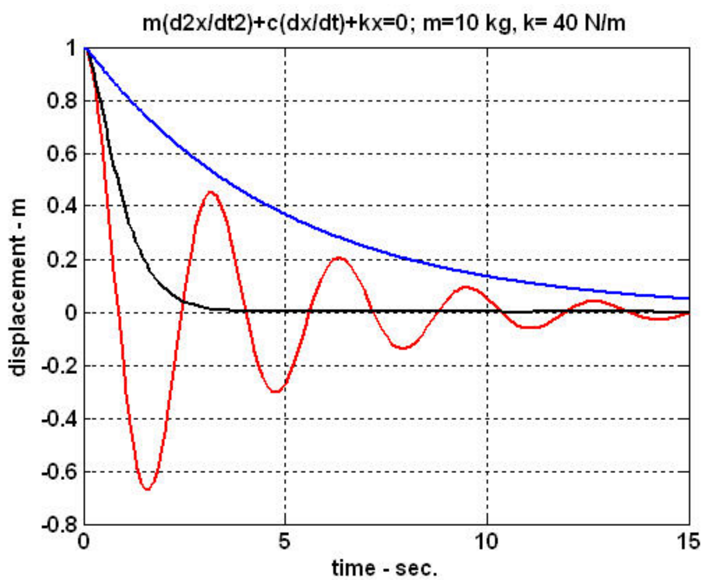


25.16 Main Program:

```
%Damped spring mass system
%mass:  m=10 kg
%damping: c=5,40,200 N/(m/s)
%spring: k=40 N/m
% MATLAB 5 version
%Independent Variable t, tspan=[tstart tstop]
%initial conditions [x(1)=velocity, x(2)=displacement];

tspan=[0 15];  ic=[0 1];

global cm km
m=10; c(1)=5; c(2)=40; c(3)=200; k=40;
km=k/m;

for n=1:3
  cm=c(n)/m;
  [t,x]=ode45('kc',tspan,ic);
  plot(t,x(:,2)); grid;
  xlabel('time - sec.'); ylabel('displacement - m');
  title('m(d2x/dt2)+c(dx/dt)+kx=0; m=10 kg, k= 40 N/m')
  hold on
end
```

Function ‘kc’:

```
%Damped spring mass system - m d2x/dt2 + c dx/dt + k x =0
%mass:  m=10 kg
%damping: c=5,40,200 N/(m/s)
%spring: k=40 N/m
%x(1)=velocity, x(2)=displacement

function dx=kc(t,x);
global cm km
dx=[-cm*x(1)-km*x(2); x(1)];
```



25.17 The MATLAB program shown below performs the Euler Method and displays the time just before the cylindrical tank empties ($y < 0$).

```
%prob2521.m
dt=0.5;
t=0;
y=3;
i=1;
while(1)
  y=y+dydt(t,y)*dt;
  if y<0, break, end
  t=t+dt;
  i=i+1;
end

function dy=dydt(t,y);
dy=-0.06*sqrt(y);
```

The result is 56 minutes as shown below

```
>> prob2521
t =
    56
```

25.18

$$x = x(1)$$

$$v = \frac{dx}{dt} = x(2)$$

$$\frac{dv}{dt} = \frac{d}{dt}\left(\frac{dx}{dt}\right) = \frac{d^2x}{dt^2}$$

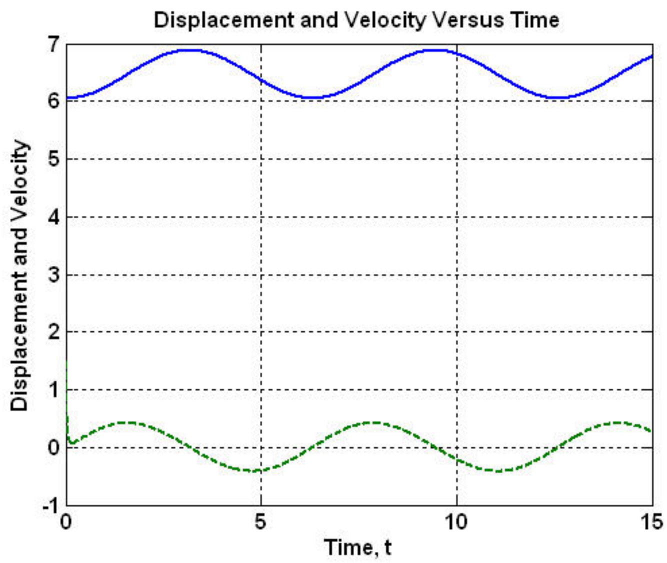
$$\frac{dx(1)}{dt} = x(2)$$

$$\frac{dx(2)}{dt} = -5x(1)x(2) - (x(1)+7)\sin(t)$$

$$x(1)(t=0) = 6$$

$$x(2)(t=0) = 1.5$$

```
tspan=[0,15]';
x0=[6,1.5]';
[t,x]=ode45('dxdt',tspan,x0);
plot(t,x(:,1),t,x(:,2),'--')
grid
title('Displacement and Velocity Versus Time')
xlabel('Time, t')
ylabel('Displacement and Velocity')

function dx=dxdt(t,x)
dx=[x(2);-5*x(1)*x(2)+(x(1)+7)*sin(1*t)];
```



25.19 The two differential equations to be solved are

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

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

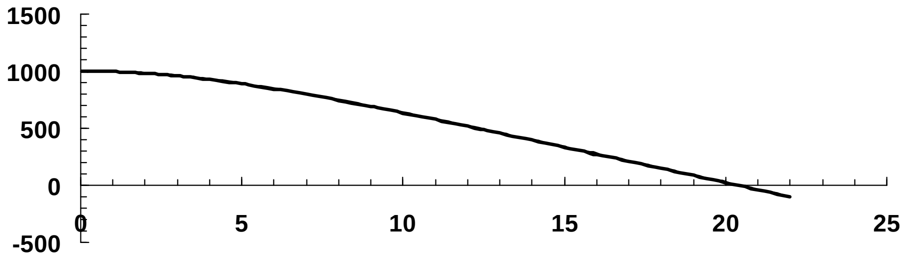
(a) Here are the first few steps of Euler's method with a step size of h = 0.2.

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.



25.20 The volume of the tank can be computed as

$$\frac{dV}{dt} = -CA\sqrt{2gH} \qquad (1)$$

This equation cannot be solved because it has 2 unknowns: V and H. The volume is related to the depth of liquid by

$$V = \frac{\pi H^2(3r - H)}{3} \qquad (2)$$

Equation (2) can be differentiated to give

$$\frac{dV}{dt} = (2\pi rH - \pi H^2)\frac{dH}{dt} \qquad (3)$$

This result can be substituted into Eq. (1) to give and equation with 1 unknown,

$$\frac{dH}{dt} = -\frac{CA\sqrt{2gH}}{2\pi rH - \pi H^2} \qquad (4)$$

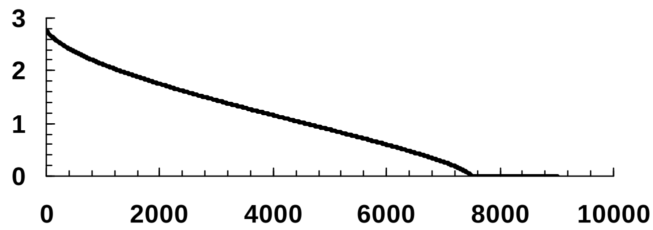
The area of the orifice can be computed as

$$A = \pi(0.015)^2 = 0.000707$$

Substituting this value along with the other parameters (C = 0.55, g = 9.81, r = 1.5) into Eq. (4) gives

$$\frac{dH}{dt} = -0.000548144\frac{\sqrt{H}}{3H - H^2} \qquad (5)$$

We can solve this equation with an initial condition of H = 2.75 m using the 4th-order RK method with a step size of 6 s. If this is done, the result can be plotted as shown,

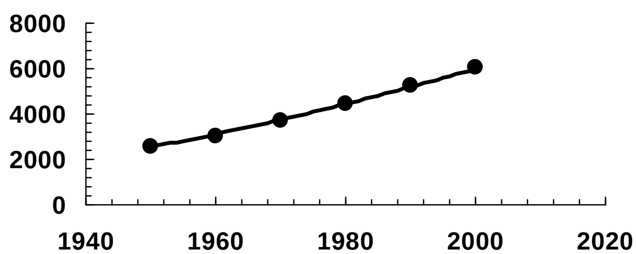


The results indicate that the tank empties at between t = 7482 and 7488 seconds.

25.21 The solution can be obtained with the 4th-order RK method with a step size of 1. Here are the results of the first few steps.

x	*y*
1950	2555
1951	2607.676
1952	2661.132
1953	2715.368
1954	2770.38
1955	2826.168

The entire solution along with the data can be plotted as



25.22 The equations to be integrated are

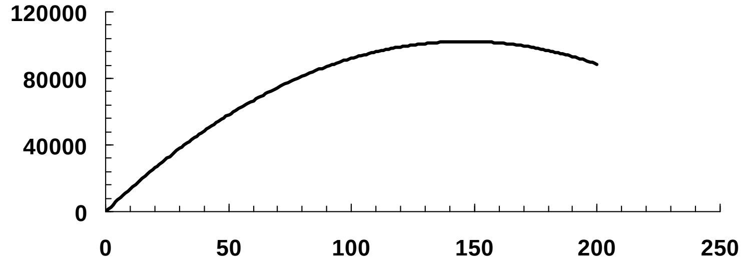
$$\frac{dv}{dt} = -9.81\frac{(6.37\times10^6)^2}{(6.37\times10^6 + x)^2}$$

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

The solution can be obtained with Euler's method with a step size of 1. Here are the results of the first few steps.

t	*v*	*x*	*dv/dt*	*dx/dt*
0	1400	0	-9.81	1400
1	1390.19	1400	-9.80569	1390.19
2	1380.384	2790.19	-9.80141	1380.384
3	1370.583	4170.574	-9.79717	1370.583
4	1360.786	5541.157	-9.79296	1360.786
5	1350.993	6901.943	-9.78878	1350.993

The entire solution for height can be plotted as



The maximum height occurs at about 146 s.

25.23 First, we must recognize that the evaluation of the definite integral

$$I = \int_a^b f(x)\, dx$$

is equivalent to solving the differential equation

$$\frac{dy}{dx} = f(x)$$

for $y(b)$ given the initial condition $y(a) = 0$. Thus, we must solve the following initial-value problem:

$$\frac{dy}{dx} = \frac{1}{(x-0.3)^2+0.01} + \frac{1}{(x-0.9)^2+0.04} - 6$$

where $y(0) = 0$. We can do this in a number of ways. One convenient approach is to use the MATLAB function `ode45` which implements an adaptive RK method. To do this, we must first set up an M-file to evaluate the right-hand side of the differential equation,

```
function dy = humpsODE(x,y)
dy = 1./((x-0.3).^2 + 0.01) + 1./((x-0.9).^2 + 0.04) - 6;
```

Then, the integral can be evaluated as

```
>> [x,y] = ode45(@humpsODE,[0 0.5 1],0);
>> disp([x,y])
                  0                  0
   0.50000000000000  21.78356481821654
   1.00000000000000  29.85525185285369
```

Thus, the integral estimate is approximately 29.86.