Problem 1. (a) For (i) write \( y_{n+1} = y_n + h \lambda y_n = (1 + \lambda h)y_n \). With \( y_0 = 1 \) we have \( y_n = (1 + \lambda h)^n \).

For (ii) note that \( y_n \) converges to 0 if and only if \(-1 < 1 + \lambda h < 1 \) or \(-2 < \lambda h < 0 \). For (iii) write \( y_n(t) = (1 + \lambda t/n)^n = \exp(n \log(1 + \lambda t/n)) \). By Taylor’s theorem for any \( x \), \( \log(1 + x) = x - x^2/2(1 + x)^2 \) for some \( \xi \in [0, x] \). Thus

\[
\exp \left( n \log \left( 1 + \frac{\lambda t}{n} \right) \right) = \exp \left( \lambda t - \frac{\lambda^2 t^2}{2n(1 + \xi)^2} \right) \to \exp(\lambda t),
\]
or in other words, with \( y(t) = \exp(\lambda t) \) we have \( y_n(t) \to y(t) \). Now write the series expansion

\[
\exp \left( \frac{\lambda t}{n} \right) = 1 + \frac{\lambda t}{n} + \frac{\lambda^2 t^2}{2n^2} + O(n^{-3}),
\]
so that

\[
1 + \frac{\lambda t}{n} = \exp \left( \frac{\lambda t}{n} \right) - \frac{\lambda^2 t^2}{2n^2} + O(n^{-3})
\]

\[
= \exp \left( \frac{\lambda t}{n} \right) \left[ 1 - \frac{\lambda^2 t^2}{2n^2} \exp \left( -\frac{\lambda t}{n} \right) + O(n^{-3}) \right]
\]

\[
= \exp \left( \frac{\lambda t}{n} \right) \left[ 1 - \frac{\lambda^2 t^2}{2n^2} (1 + O(n^{-1})) + O(n^{-3}) \right]
\]

\[
= \exp \left( \frac{\lambda t}{n} \right) \left[ 1 - \frac{\lambda^2 t^2}{2n^2} + O(n^{-3}) \right].
\]

Then

\[
y_n(t) - y(t) = \left( 1 + \frac{\lambda t}{n} \right)^n - \exp(\lambda t)
\]

\[
= \exp(\lambda t) \left( 1 - \frac{\lambda^2 t^2}{2n^2} + O(n^{-3}) \right)^n - \exp(\lambda t)
\]

\[
= \exp(\lambda t) \exp \left( n \log \left( 1 - \frac{\lambda^2 t^2}{2n^2} + O(n^{-3}) \right) \right) - \exp(\lambda t)
\]

\[
= \exp(\lambda t) \exp \left( -\frac{\lambda^2 t^2}{2n} + O(n^{-2}) \right) - \exp(\lambda t)
\]

\[
= \exp(\lambda t) \left( 1 - \frac{\lambda^2 t^2}{2n} + O(n^{-2}) \right) - \exp(\lambda t)
\]

\[
= -\frac{\lambda^2 t^2}{2n} \exp(\lambda t) + O(n^{-2}).
\]

For (iv) write \( \delta'(t) = \lambda \delta(t) + \lambda^2 \exp(\lambda t)/2 \). This is a 1st-order nonhomogeneous linear ODE, so we use the method of variation of parameters to obtain the general solution \( \delta(t) = c \exp(\lambda t) + \lambda^2 t \exp(\lambda t)/2 \), where \( \delta(0) = 0 \) lets us pick \( c = 0 \). Evidently \( -\delta(t)h + O(h^2) = -\lambda^2 t^2 \exp(\lambda t)/2n + O(n^{-2}) \).

(b) For (i) write \( k_1 = f(t_n, y_n) = \lambda y_n \) and \( k_2 = f(t_n + h, y_n + h k_1) = \lambda(1 + \lambda h) y_n \). Then

\[
y_{n+1} = y_n + \frac{h}{2} (k_1 + k_2) = \left( 1 + \lambda h + \frac{\lambda^2 h^2}{2} \right) y_n.
\]

With \( y_0 = 0 \) we thus have have \( y_n = (1 + \lambda h + \lambda^2 h^2/2)^n \). For (ii) the sequence \( y_n \) converges only if \(-1 < 1 + \lambda h + \lambda^2 h^2/2 < 1 \) or \(-1 < 1/2 + (1 + \lambda h)^2/2 < 1 \), meaning \(-2 < \lambda h < 0 \). For (iii) write the series expansion
\[ \exp\left(\frac{\lambda t}{n}\right) = 1 + \frac{\lambda t}{n} + \frac{\lambda^2 t^2}{2n^2} + \frac{\lambda^3 t^3}{6n^3} + O(n^{-4}). \]

This gives
\[ 1 + \frac{\lambda t}{n} + \frac{\lambda^2 t^2}{2n^2} = \exp\left(\frac{\lambda t}{n}\right) - \frac{\lambda^3 t^3}{6n^3} + O(n^{-4}) = \exp\left(\frac{\lambda t}{n}\right) \left(1 - \frac{\lambda^3 t^3}{6n^3} + O(n^{-4})\right), \]

where as before we use \(\exp(-\lambda t/n) = 1 + O(n^{-1})\). We finish our argument in a similar manner
\[
y_n(t) - y(t) = \left(1 + \frac{\lambda t}{n} + \frac{\lambda^2 t^2}{2n^2}\right)^n - \exp(\lambda t)
= \exp(\lambda t) \left(1 - \frac{\lambda^3 t^3}{6n^3} + O(n^{-4})\right)^n - \exp(\lambda t)
= \exp(\lambda t) \exp\left(n \log\left(1 - \frac{\lambda^3 t^3}{6n^3} + O(n^{-4})\right)\right) - \exp(\lambda t)
= \exp(\lambda t) \exp\left(-\frac{\lambda^3 t^3}{6n^2} + O(n^{-3})\right) - \exp(\lambda t)
= \exp(\lambda t) \left(1 - \frac{\lambda^3 t^3}{6n^2} + O(n^{-3})\right) - \exp(\lambda t)
= -\frac{\lambda^3 t^3}{6n^2} \exp(\lambda t) + O(n^{-3}). \]

For (iv) write \(\delta'(t) = \lambda\delta(t) + \lambda^3 \exp(\lambda t)/6\). We solve this in an identical manner to find the general solution \(\delta(t) = c \exp(\lambda t) + \lambda t \exp(\lambda t)/6\), where \(y(0) = 0\) gives us \(c = 0\). We find that \(-\delta(t)h^2 + O(h^3) = -\lambda^3 t^3 \exp(\lambda t)/n^2 + O(n^{-3})\), as desired.

**Problem 2.** (a) Write the truncation error
\[ T_n = y(t_n + h) - y(t_n) - h \left[ \gamma_1 K_1 + \gamma_2 K_2 + \gamma_3 K_3 \right]. \]

If \(T_n = O(h^4)\) then \(\tau_n = T_n/h = O(h^3)\). We’ll thus try to find conditions for the former. First note the series expansion \(y(t_n + h) - y(t_n) = y'(t_n)h + y''(t_n)h^2/2 + y'''(t_n)h^3/6 + O(h^4)\). We find that
\[
y' = f \\
y'' = f_t + y'f_y = f_t + f_y f \\
y''' = f_{tt} + y'f_{ty} + f_{ty}f_y + f_tf_y + y'f_{yy}f = f_{tt} + 2f_tf_y + f_{yy}f^2 + f_tf_y + f_y^2f. \]

Series expand the \(K_i\) to obtain
\[
K_1 = f \\
K_2 = f + \alpha_2 h f_t + \beta_{21} h f_y K_1 + \alpha_2^2 h^2 f_{tt}/2 + \alpha_2 \beta_{21} h^2 f_y K_1 + \beta_{21}^2 h^2 f_{yy}K_2^2/2 + O(h^3) \\
= f + (\alpha_2 f_t + \beta_{21} f_y h) + (\alpha_2^2 h^2 f_{tt}/2 + \alpha_2 \beta_{21} h^2 f_y + \beta_{21}^2 h^2 f_{yy})/2) h^2 + O(h^3) \\
K_3 = f + \alpha_3 h f_t + (\beta_{31} K_1 + \beta_{32} K_2) h f_y + \alpha_3^2 h^2 f_{tt}/2 + \alpha_3 (\beta_{31} + \beta_{32} K_2) h f_y + \beta_{31}^2 h^2 f_{yy}/2 + O(h^3) \\
+ (\beta_{31} K_1 + \beta_{32} K_2) h^2 f_{yy}/2 + O(h^3) \\
= f + (\beta_{31} + \beta_{32}) f_y h) + (\alpha_3 \beta_{32} f_{tt} + \beta_{32} \beta_{31} f_{yy}) f + \alpha_3^2 f_{tt}/2 \\
+ \alpha_3 \beta_{32} f_{tf_y} + \beta_{31} + \beta_{32}^2 f_{yy}/2) h^2 + O(h^3). \]

Plug \(y', y'', y'''\), \(K_1, K_2, K_3\) into \(T_n\) with our series expansion to find the coefficient on \(h\) to be
\[ f - \gamma_1 f - \gamma_2 f - \gamma_3 f, \]
on \(h^2\) to be
\[
\left[ 1 - \frac{1}{2} \gamma_2 \alpha_2 - \gamma_3 \alpha_3 \right] f_t + \left[ 1 - \frac{1}{2} \gamma_2 \beta_{21} - \gamma_3 (\beta_{31} - \beta_{32}) \right] f_y f,
\]

and on \( h^3 \) to be

\[
\left[ \frac{1}{6} - \frac{1}{6}(\gamma_2 \alpha_2^2 + \gamma_3 \alpha_3^2) \right] f_t + \left[ \frac{1}{3} - (\gamma_2 \alpha_2 \beta_{21} + \gamma_3 \alpha_3 (\beta_{31} + \beta_{32})) \right] f_y f
\]

\[
+ \left[ \frac{1}{6} - \frac{1}{2} (\gamma_2 \beta_{21}^2 + \gamma_3 (\beta_{31} + \beta_{32})^2) \right] f_y f^2 + \left[ \frac{1}{6} - \gamma_3 \alpha_2 \beta_{32} \right] f_t f_y + \left[ \frac{1}{6} - \gamma_3 \beta_{32} \beta_{31} \right] f_y f^3.
\]

All coefficients must vanish in order for \( T_n \) to be \( O(h^4) \), universally for any choice of \( f \). In other words, for \( \tau_n \) to be \( O(h^3) \) the following are collectively necessary and sufficient:

\[
\begin{align*}
\gamma_1 + \gamma_2 + \gamma_3 &= 1 \\
\gamma_2 \alpha_2 + \gamma_3 \alpha_3 &= 1/2 \\
\gamma_2 \beta_{21} + \gamma_3 (\beta_{31} + \beta_{32}) &= 1/2 \\
\gamma_2 \alpha_2^2 + \gamma_3 \alpha_3^2 &= 1/3 \\
\gamma_2 \alpha_2 \beta_{21} + \gamma_3 \alpha_3 (\beta_{31} + \beta_{32}) &= 1/3 \\
\gamma_2 \beta_{21}^2 + \gamma_3 (\beta_{31} + \beta_{32})^2 &= 1/3 \\
\gamma_3 \alpha_2 \beta_{32} &= 1/6 \\
\gamma_3 \beta_{32} \beta_{31} &= 1/6.
\end{align*}
\] (1)

(b) Plugging into (1) we find the conditions are satisfied, and thus the Runge–Kutta method with these specific parameters is of 3rd order.

(c) See Program 1 for our implementation of \( \text{rk3.m} \). See Program 2 for our solution to the given IVP. Below we summarize our findings. By inspection we see that errors decrease approximate as \( h^3 \), in agreement with our work above.

<table>
<thead>
<tr>
<th>Increments</th>
<th>Runge–Kutta</th>
<th>Exact</th>
<th>Error</th>
<th>Log-Error Ratio</th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>1.359</td>
<td>1.359</td>
<td>0.0001508</td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>1.3591</td>
<td>1.3591</td>
<td>1.8634 \times 10^{-9}</td>
<td>3.0166</td>
</tr>
<tr>
<td>16</td>
<td>1.3591</td>
<td>1.3591</td>
<td>2.308 \times 10^{-6}</td>
<td>3.0133</td>
</tr>
<tr>
<td>32</td>
<td>1.3591</td>
<td>1.3591</td>
<td>2.8686 \times 10^{-4}</td>
<td>3.0082</td>
</tr>
<tr>
<td>64</td>
<td>1.3591</td>
<td>1.3591</td>
<td>3.5745 \times 10^{-8}</td>
<td>3.0045</td>
</tr>
<tr>
<td>128</td>
<td>1.3591</td>
<td>1.3591</td>
<td>4.4608 \times 10^{-9}</td>
<td>3.0024</td>
</tr>
<tr>
<td>256</td>
<td>1.3591</td>
<td>1.3591</td>
<td>5.5713 \times 10^{-10}</td>
<td>3.0012</td>
</tr>
<tr>
<td>512</td>
<td>1.3591</td>
<td>1.3591</td>
<td>6.9613 \times 10^{-11}</td>
<td>3.0006</td>
</tr>
<tr>
<td>1024</td>
<td>1.3591</td>
<td>1.3591</td>
<td>8.6957 \times 10^{-12}</td>
<td>3.001</td>
</tr>
<tr>
<td>2048</td>
<td>1.3591</td>
<td>1.3591</td>
<td>1.0889 \times 10^{-12}</td>
<td>2.9974</td>
</tr>
</tbody>
</table>

Problem 3. (a) See Program 3 for our solution. Below we summarize our findings.

<table>
<thead>
<tr>
<th>Increments</th>
<th>Heun</th>
<th>Exact</th>
<th>Error</th>
<th>Log-Error Ratio</th>
</tr>
</thead>
<tbody>
<tr>
<td>100</td>
<td>0.57163</td>
<td>0.57143</td>
<td>0.00019797</td>
<td></td>
</tr>
<tr>
<td>200</td>
<td>0.57151</td>
<td>0.57143</td>
<td>7.8174 \times 10^{-9}</td>
<td>1.3405</td>
</tr>
<tr>
<td>400</td>
<td>0.57146</td>
<td>0.57143</td>
<td>3.0919 \times 10^{-9}</td>
<td>1.3382</td>
</tr>
<tr>
<td>800</td>
<td>0.57144</td>
<td>0.57143</td>
<td>1.2243 \times 10^{-9}</td>
<td>1.3365</td>
</tr>
<tr>
<td>1600</td>
<td>0.57143</td>
<td>0.57143</td>
<td>4.8516 \times 10^{-9}</td>
<td>1.3354</td>
</tr>
<tr>
<td>3200</td>
<td>0.57143</td>
<td>0.57143</td>
<td>1.9236 \times 10^{-9}</td>
<td>1.3347</td>
</tr>
<tr>
<td>6400</td>
<td>0.57143</td>
<td>0.57143</td>
<td>7.6292 \times 10^{-9}</td>
<td>1.3342</td>
</tr>
</tbody>
</table>
By inspection we see our convergence isn't quadratic as we expect of Heun's method. Notice that

\[ Y'(t) = -\frac{16t^{1/3}}{(3t^{1/3} + 4)^2}, \quad Y''(t) = \frac{336t^{4/3} - 64}{4t^{2/3}(3t^{1/3} + 4)^3}. \]

In other words, \( Y' \) is non-differentiable at \( t = 0 \). This was however a crucial assumption in the series expansions used to show the order of convergence of Heun's method.

(b) See Program 4 for our solution. Below we summarize our findings. For the same reason as in (a) we do not see the expected 4th-order convergence.

<table>
<thead>
<tr>
<th>Increments</th>
<th>Runge–Kutta</th>
<th>Exact ( \times 10^{-6} )</th>
<th>Error</th>
<th>Log-Error Ratio</th>
</tr>
</thead>
<tbody>
<tr>
<td>100</td>
<td>0.57147</td>
<td>0.57143</td>
<td>3.819</td>
<td>-</td>
</tr>
<tr>
<td>200</td>
<td>0.57144</td>
<td>0.57143</td>
<td>1.5159</td>
<td>1.3331</td>
</tr>
<tr>
<td>400</td>
<td>0.57143</td>
<td>0.57143</td>
<td>6.0162</td>
<td>1.3332</td>
</tr>
<tr>
<td>800</td>
<td>0.57143</td>
<td>0.57143</td>
<td>2.3876</td>
<td>1.3333</td>
</tr>
<tr>
<td>1600</td>
<td>0.57143</td>
<td>0.57143</td>
<td>9.4753</td>
<td>1.3333</td>
</tr>
<tr>
<td>3200</td>
<td>0.57143</td>
<td>0.57143</td>
<td>3.7603</td>
<td>1.3333</td>
</tr>
<tr>
<td>6400</td>
<td>0.57143</td>
<td>0.57143</td>
<td>1.4923</td>
<td>1.3333</td>
</tr>
</tbody>
</table>

**Problem 4.** *(a)* Write the series expansion and use Taylor's theorem to get that, for some \( \theta \in [0, 1] \),

\[
 f(t+h) = f(t) + f'(t)h + \frac{f''(t)h^2}{2} + \cdots + \frac{f^{(k-1)}(t)h^{k-1}}{(k-1)!} + \frac{f^{(k)}(t + \theta h)h^k}{k!}.
\]

But using \( \alpha \)-Hölder continuity we have \( |f^{(k)}(t + \theta h) - f^{(k)}(t)| \leq K|\theta h|^{\alpha} \) and thus

\[
 f^{(k)}(t + \theta h) = f^{(k)}(t) + [f^{(k)}(t + \theta h) - f^{(k)}(t)] = f^{(k)}(t) + O(h^{\alpha}).
\]

Thus

\[
 f(t+h) = f(t) + f'(t)h + \frac{f''(t)h^2}{2} + \cdots + \frac{f^{(k)}(t)h^k}{k!} + O(h^{k+\alpha}),
\]

which lets us write

\[
 Y(t+h) = Y(t) + Y'(t)h + \frac{Y''(t)h^2}{2} + \cdots + \frac{Y^{(k+1)}(t)h^{k+1}}{(k+1)!} + O(h^{k+\alpha+1}).
\]

We perform a multivariate series expansion using Taylor's theorem such that for some \( \theta \in [0, 1] \):

\[
 f(t + ah, y + bh) = f(t, y) + (ah \partial_t + bh \partial_y) f(t, y) + \cdots + \frac{1}{k!} (ah \partial_t + bh \partial_y)^k f(t + \theta ah, y + \theta bh).
\]

Just as before we apply \( \alpha \)-Hölder continuity to get

\[
 \partial_t^k f(t + \theta ah, y + \theta bh) = \partial_t^k f(t, y) + O(h^{\alpha}).
\]

Note the second argument is smooth and bounded on the \((k + 1)\)st derivative, and thus the mean value theorem guarantees it is \( \alpha \)-Hölder continuous for every \( \alpha \). Then

\[
 f(t + ah, y + bh) = f(t, y) + (ah \partial_t + bh \partial_y) f(t, y) + \cdots + \frac{1}{k!} (ah \partial_t + bh \partial_y)^k f(t, y) + O(h^{k+\alpha}),
\]

and we write the truncation error of the explicit Runge–Kutta update, while recalling our assumption that the update is of \( n \)th order:

\[
 T(t, Y) := \left\{ Y(t + h) - Y(t) \right\} = \left\{ h \sum_{i=1}^{p} c_i \left[ t + ah, y + h \sum_{j=1}^{i-1} \beta_{ij}k_i \right] \right\} = \left\{ O(h^{n+1}) \right\} \text{ if } k \geq n \quad \text{and} \quad \left\{ O(h^{k+\alpha+1}) \right\} \text{ if } k < n.
\]
First, if \( k \geq n \) the order of convergence is immediate—all the coefficients of \( 1, h, h^2, \ldots, h^n \) in both the blue and red brackets agree and the remaining terms are of order \( n \) or higher. Next, if \( k < n \), we are still automatically granted that all coefficients \( 1, h, h^2, \ldots, h^k \) agree in both the blue and red brackets. However, we observe that in the red brackets the order of the residual is, via our work in (3), \( hO(h^{k+\alpha}) = O(h^{k+\alpha+1}) \). In the blue brackets we already showed in (2) that the order of the residual is \( O(h^{k+\alpha+1}) \). Thus (4) is \( O(h^{k+\alpha+1}) \) and we’re done.

(b) In the very first step, the arguments of part (a) apply and since \( k < n \) thus
\[
Y(t_1) - y_1 = T(t_0, Y) = O(h^{k+\alpha+1}),
\]
assuming that \( Y(0) = y_0 \).

However, since \( t_1 = t_0 + h > t_0 \) the solution \( Y(t) \) and the function \( f(t, y) \) are both \( C^\infty \) in time for all \( t \geq t_1 \). Thus, the method is order \( n \) after the first step! The general error bound derived in class therefore gives
\[
\max_{i=1,\ldots,n} |Y(t_i) - y_i| \leq \exp ([t_n - t_1]L) |Y(t_1) - y_1| + \frac{\exp([t_n - t_1]L) - 1}{L} \max_{j=1,\ldots,n} |\tau_j|.
\]

Here \( \max_{j=1,\ldots,n} |\tau_j| = O(h^n) \) so that the error is dominated for small \( h \) by the first term proportional to \( |Y(t_1) - y_1| = O(h^{k+\alpha+1}) \). This is the statement that the method is of order at least \( k + \alpha + 1 \).

Note \( f \) in Problem 3 is zero times continuously differentiable and 1/3-Hölder in \( t \) in any small neighborhood of the origin, but \( C^\infty \) elsewhere. We’re thus guaranteed an order of convergence at least 4/3, which is what we observe numerically for the order approximation in Problem 3(b).

(c) See Program 5 for the implementation of the solution. See Figure 1 for the approximate order of convergence. We find the approximate order of convergence to be 1.33108, which is greater than 1/3. This is a larger rate of convergence than implied by our work in (a), which only guaranteed a \((k + \alpha)\)th-order rate of convergence, which in this case is 1/3. There is no logical contradiction with part (a), just a bit of a mystery why the convergence rate is higher than guaranteed.

Note, however, that the Heun approximation for this problem is
\[
y_{Heun}(1) = y(0) + \sum_{k=0}^{n-1} \frac{h}{2} [f(t_k) + f(t_{k+1})].
\]

This is \( y(0) \) plus the trapezoidal approximation to the integral \( \int_0^1 f(t) \, dt \), whose order of convergence is \( 1 + \alpha \) if the integrand is \( \alpha \)-Hölder continuous. As it happens, this 4/3 rate for \( \alpha = 1/3 \) approximately agrees with our findings.

Likewise, the RK4 approximation is
\[
y_{RK4}(1) = y(0) + \sum_{k=0}^{n-1} \frac{h}{6} \left[ f(t_k) + 4f(t_{k+\frac{1}{2}}) + f(t_{k+1}) \right].
\]

and this is \( y(0) \) plus the Simpson approximation to the integral \( \int_0^1 f(t) \, dt \), whose order of convergence again is \( 1 + \alpha \) if the integrand is \( \alpha \)-Hölder continuous. The rate 4/3 is again quite close to what we observed numerically.

In general, convergence and stability is much better for numerical methods applied to ODE’s \( \dot{y}(t) = f(t) \) where \( f(t) \) is \( y \)-independent. In this case, solving the ODE just reduces to calculating the integral \( \int_0^1 f(t') \, dt' \).
Figure 1. Error in Heun method for Problem 4(c). Approximate order 1.3311.
MATLAB Code

Program 1. Implement rk3.m.

```matlab
function [ t,y ] = rk3(f,tspan,y_0,N_s,yes)

  t_0=tspan(1);
  t_f=tspan(2);
  D=length(y_0);

  dt = (t_f - t_0)/N_s;

  t = t_0 : dt : t_f;
  N=length(t);

j = 1;
y(1,:) = y_0(:)';

  while j < N
    yj=y(j,:);
    k1 = feval(f,t(j),yj);
    k2 = feval(f,t(j)+dt/2,yj+k1*dt/2);
    k3 = feval(f,t(j)+3*dt/4,yj+3*k2*dt/4);
    yj = yj + dt*(2* k1 +3* k2 +4* k3)/9;
    y(j +1,:) = yj';
    j = j + 1;
  end

if yes ==1
  for k=1:D,
    figure
    z=y(:,k);
    plot(t,z)
    xlabel('time t')
    ylabel(sprintf('y_%d', k))
  end
end

t=t';

return
```

Program 2. Solve Problem 2(c).

```matlab
f = @(t,y) t*y/(1+t);
tspan=[0 1]
y_0=1
Y = @(t) exp(t)./(1+t);

for i=1:10,
  N_s=2^(i+1);
```
Program 3. Solve Problem 3(a).

```matlab
f=@(t,y,p) -t^(p-1)*y.^2;
Y=@(t,p) 1./(1+t.^p/p);
y_0=1
tspan=[0 1]
p=4/3;
for i=1:7,
    N_s=50*2^i;
    [ t,y ] = heun(@f(t,y) f(t,y,p),tspan,y_0,N_s,0);
    s=y(1+N_s,:);
    y1(i,:)=s;
    S=feval(Y,t(1+N_s));
    Y1(i,:)=S;
    E1(i)=norm(s-S);
end
err1=y-Y(t);
disp(['N_s, RK3, exact, error, log error-ratio'])
disp(['4 ', num2str(y1(1,:)) , ' ', num2str(Y1(1,:)) , ' ... ' , num2str(E1(1)) , ' NaN'])
for i =2:10
    num=num2str(2^(i+1));
    for j =1:6-length(num)
        num=[num , ' '];
    end
    yst=num2str(y1(i,:));
    for j =1:9-length(yst)
        yst=[yst , ' '];
    end
    disp([num , ' ', yst , ' ', num2str(Y1(i,:)) , ' ... ' , num2str(E1(i)) , ' ', num2str(log2(E1(i-1)/E1(i)))])
end
```

```matlab
[ t,y ] = rk3(f,tspan,y_0,N_s,0);
s=y(1+N_s,:);
y1(i,:)=s;
S=feval(Y,t(1+N_s));
Y1(i,:)=S;
E1(i)=norm(s-S);
```
Program 4. Solve Problem 3(b).

```matlab
f=@(t,y,p) -t.^(p-1)*y.^2;
Y=@(t,p) 1./(1+t.^p/p);
y_0 =1

for i =1:7,
    N_s =50*2^i;
    [ t,y ] = rk4(@(t,y) f(t,y,p) ,tspan,y_0,N_s,0);
    s=y(1+N_s,:);
    y1(i,:)=s;
    S=feval(@(t) Y(t,p),t(1+N_s));
    Y1(i,:)=S;
    E1(i)=norm(s-S);
end
```

```matlab
disp([N_s, RK4, exact, error, log error-ratio'])
disp([100 ', num2str(y1(1,:)),', num2str(Y1(1,:)),', ...
',num2str(E1(1)),', NaN'])
```

```matlab
for i=2:7
    num=num2str(50*2^(i));
    for j=1:6-length(num)
        num=[num,' ']
    end
    yst=num2str(y1(i,:));
    for j=1:9-length(yst)
        yst=[yst,' ']
    end
    disp([num,yst, num2str(Y1(i,:)),', num2str(E1(i)),', ...
         ',num2str(log2(E1(i-1)/E1(i)))])
end
```
Program 5. Solve Problem 4(c).

\[
y_0 = 0
\]
\[
tspan = [0 \ 1]
\]
\[
alp = 1/3;
\]
\[
for \ i = 1:12,
\]
\[
N_s = 50*2^i;
\]
\[
[t, y] = heun(@(t, y) wcfun(t, y, alp), tspan, y_0, N_s, 0);
\]
\[
s = y(1+N_s, :);
\]
\[
y1(i, :) = s;
\]
\[
S = feval(@(t) wsfun(t, alp), t(1+N_s));
\]
\[
Y1(i, :) = S;
\]
\[
E1(i) = s - S;
\]
\[
end
\]
\[
disp(['N_s, heun, exact, error, log error-ratio'])
\]
\[
disp(['100 ', num2str(y1(1, :)), ' ', num2str(Y1(1, :)), ' ...
\]
\[
', num2str(abs(E1(1))), ' NaN'])
\]
\[
for \ i = 2:12
\]
\[
num = num2str(50*2^i);
\]
\[
for \ j = 1:6-length(num)
\]
\[
num = [num, ' '];
\]
\[
end
\]
\[
y1st = num2str(y1(i, :));
\]
\[
for \ j = 1:9-length(y1st)
\]
\[
y1st = [y1st, ' '];
\]
\[
end
\]
\[
disp([num, y1st, num2str(Y1(i, :)), ' ', num2str(E1(i)), ' ...
\]
\[
', num2str(log2(abs(E1(i-1)/E1(i))))])
\]
end

h=1./(50*2.^(1:12));
P=polyfit(log(h),log(abs(E1)),1);
m=P(1)
b=P(2)

figure; loglog(h,abs(E1),'-b',h,exp(polyval(P,log(h))),'-r','LineWidth',2)
xlabel('h','FontSize',15)
ylabel('E(h)','FontSize',15)
legend('error','fit')
title(['Error in Heun Method Showing Approximate Order n=',num2str(m), ...]
     
     'FontSize',15)