**Question 1** The position \((x(t), y(t))\) of a satellite orbiting around the earth and moon is described by the second-order system of ordinary differential equations

\[
\begin{align*}
x'' &= x + 2y' - b \frac{x + a}{\left((x + a)^2 + y^2\right)^{3/2}} - a \frac{x - b}{\left((x - b)^2 + y^2\right)^{3/2}} \\
y'' &= y - 2x' - b \frac{y}{\left((x + a)^2 + y^2\right)^{3/2}} - a \frac{y}{\left((x - b)^2 + y^2\right)^{3/2}}
\end{align*}
\]

where \(a = 0.012277471\) and \(b = 1 - a\). When the initial conditions

\[
\begin{align*}
x(0) &= 0.994 \\
x'(0) &= 0 \\
y(0) &= 0 \\
y'(0) &= -2.00158510637908
\end{align*}
\]

are satisfied, there is a periodic orbit with period \(T = 17.06521656015796\).

(a) Convert this problem to a \(4 \times 4\) first-order system \(u' = f(t, u)\), \(u(0) = u_0\), by introducing

\[u = [x, x', y, y'] = [u_1, u_2, u_3, u_4]\]

as a new vector unknown function and defining \(f\) appropriately.

(b) Choose a rectangular region in the 4-dimensional \(u\) space which includes the initial point \(u_0\), and find a Lipschitz constant \(L\) for \(f\) on that region.

**Proof.** (a) We have

\[
\begin{align*}
u_1' &= f_1(t, u) = x' = u_2 \\
u_2' &= f_2(t, u) = y' = u_4 \\
u_3' &= f_3(t, u) = x'' \\
u_4' &= f_4(t, u) = y''
\end{align*}
\]

\[
\begin{align*}
u_3' &= u_3 - 2u_2 - b \frac{u_3}{\left((u_1 + a)^2 + u_3^2\right)^{3/2}} - a \frac{u_3}{\left((u_1 - b)^2 + u_3^2\right)^{3/2}} \\
u_4' &= u_4 + 2u_4 - b \frac{u_4}{\left((u_1 + a)^2 + u_4^2\right)^{3/2}} - a \frac{u_4}{\left((u_1 - b)^2 + u_4^2\right)^{3/2}}
\end{align*}
\]

(b) We take two points \(u, \tilde{u} \in \mathbb{R}^4\) and seek to show that

\[\|f(t, u) - f(t, \tilde{u})\|_1 \leq L \|u - \tilde{u}\|_1\]

for some \(L > 0\) (we use the \(L^1\) norm).

We break the difference down via

\[
\begin{align*}
\|f(t, u) - f(t, \tilde{u})\|_1 &= |f_1(t, u) - f_1(t, \tilde{u})| + |f_2(t, u) - f_2(t, \tilde{u})| \\
&\quad + |f_3(t, u) - f_3(t, \tilde{u})| + |f_4(t, u) - f_4(t, \tilde{u})| \\
&= |u_2 - \tilde{u}_2| + |f_2(t, u) - f_2(t, \tilde{u})| \\
&\quad + |u_4 - \tilde{u}_4| + |f_4(t, u) - f_4(t, \tilde{u})|
\end{align*}
\]
For the other more complex functions, we split apart the linear parts

\[ f_2(t, u) = u_1 + 2u_4 - bA_2(u_1, u_3) - aB_2(u_1, u_3), \quad f_4(t, u) = u_3 - 2u_2 - bA_4(u_1, u_3) - aB_4(u_1, u_3). \]

then use the triangle inequality to separate the linear parts from the rest:

\[
|f_2(t, u) - f_2(t, \tilde{u})| \leq |u_1 - \tilde{u}_1| + 2|u_4 - \tilde{u}_4| + b|A_2(u_1, u_3) - A_2(\tilde{u}_1, \tilde{u}_3)| + a|B_2(u_1, u_3) - B_2(\tilde{u}_1, \tilde{u}_3)|
\]

\[
|f_4(t, u) - f_4(t, \tilde{u})| \leq |u_3 - \tilde{u}_3| + 2|u_2 - \tilde{u}_2| + b|A_4(u_1, u_3) - A_4(\tilde{u}_1, \tilde{u}_3)| + a|B_4(u_1, u_3) - B_4(\tilde{u}_1, \tilde{u}_3)|.
\]

In order to bound the differences above, consider a function \( g(r, s) \). We can use the mean-value theorem in \( r \) and \( s \) in order to bound \( g(r + \Delta r, s + \Delta s) - g(r, s) \).

\[
g(r + \Delta r, s + \Delta s) - g(r, s) = \frac{\partial g}{\partial r} (\xi_r, s + \Delta s) \quad \text{for} \quad \xi_r \text{ between } r \text{ and } r + \Delta r
\]

\[
g(r, s + \Delta s) - g(r, s) = \frac{\partial g}{\partial s} (r, \xi_s) \quad \text{for} \quad \xi_s \text{ between } s \text{ and } s + \Delta s
\]

\[
\Rightarrow g(r + \Delta r, s + \Delta s) - g(r, s) = \left[ \frac{\partial g}{\partial r} (\xi_r, s + \Delta s) \right] \Delta r + \left[ \frac{\partial g}{\partial s} (r, \xi_s) \right] \Delta s
\]

\[
\Rightarrow |g(r + \Delta r, s + \Delta s) - g(r, s)| \leq |\Delta r| \max \left| \frac{\partial g}{\partial r} \right| + |\Delta s| \max \left| \frac{\partial g}{\partial s} \right|
\]

We can apply this to \( A_2, A_4, B_2, B_4 \) to get bounds. For example,

\[
|A_2(u_1, u_3) - A_2(\tilde{u}_1, \tilde{u}_3)| \leq |u_1 - \tilde{u}_1| \max \left| \frac{\partial A_2}{\partial u_1} \right| + |u_3 - \tilde{u}_3| \max \left| \frac{\partial A_2}{\partial u_3} \right|.
\]

These components are a bit difficult to differentiate, so we define some intermediate terms to aid in the computation. Defining the square roots (from the denominators) implicitly we find

\[
D_A = \sqrt{(u_1 + a)^2 + u_3^2}
\]

\[
\Rightarrow D_A^2 = (u_1 + a)^2 + u_3^2
\]

\[
\Rightarrow 2D_A \frac{\partial D_A}{\partial u_1} = 2(u_1 + a)
\]

\[
\Rightarrow \frac{\partial D_A}{\partial u_1} = \frac{u_1 + a}{D_A}
\]

and

\[
2D_A \frac{\partial D_A}{\partial u_3} = 2u_3
\]

\[
\Rightarrow \frac{\partial D_A}{\partial u_3} = \frac{u_3}{D_A}
\]

Similarly, \( D_B^2 = (u_1 - b)^2 + u_3^2 \) gives us

\[
\frac{\partial D_B}{\partial u_1} = \frac{u_1 - b}{D_B}, \quad \frac{\partial D_B}{\partial u_3} = \frac{u_3}{D_B}
\]
Using these
\[ A_2 = \frac{u_1 + a}{D_A^3} \implies \frac{\partial A_2}{\partial u_1} = \frac{1}{D_A^3} - 3 \frac{u_1 + a}{D_A^4} \frac{\partial D_A}{\partial u_1} = \frac{1}{D_A^3} - 3 \frac{u_1 + a u_1 + a}{D_A^4} \frac{\partial D_A}{\partial u_1} \]
\[ = \frac{D_A^2 - 3(u_1 + a)^2}{D_A^3} \]
and \[ \frac{\partial A_2}{\partial u_3} = -3 \frac{u_1 + a}{D_A^4} \frac{\partial D_A}{\partial u_3} = -3 \frac{u_1 + a u_3}{D_A^4} \frac{\partial D_A}{\partial u_3} \]
\[ A_4 = \frac{u_3}{D_A^4} \implies \frac{\partial A_4}{\partial u_1} = -3 \frac{u_3 (u_1 + a)}{D_A^5} \]
and \[ \frac{\partial A_4}{\partial u_3} = \frac{D_A^2 - 3u_3^2}{D_A^5} \]
\[ B_2 = \frac{u_1 - b}{D_B^4} \implies \frac{\partial B_2}{\partial u_1} = \frac{D_B^2 - 3(u_1 - b)^2}{D_B^5} \]
and \[ \frac{\partial B_2}{\partial u_3} = -3 \frac{(u_1 - b) u_3}{D_B^5} \]
\[ B_4 = \frac{u_3}{D_B^4} \implies \frac{\partial B_4}{\partial u_1} = -3 \frac{u_3 (u_1 - b)}{D_B^5} \]
and \[ \frac{\partial B_4}{\partial u_3} = \frac{D_B^2 - 3u_3^2}{D_B^5} \]

In order simplify things for this values note that
\[ D_A^2 - 3(u_1 + a)^2 = u_3^2 - 2(u_1 + a)^2 \]
\[ \implies \left| D_A^2 - 3(u_1 + a)^2 \right| \leq u_3^2 + 2(u_1 + a)^2 < 2D_A^2 \]
\[ \implies \left| \frac{\partial A_2}{\partial u_1} \right| < \frac{2D_A^2}{D_A^3} = \frac{2}{D_A^3}. \]

Applying similar logic gives
\[ \left| \frac{\partial A_4}{\partial u_3} \right| < \frac{2}{D_A^3}, \quad \left| \frac{\partial B_2}{\partial u_1} \right| < \frac{2}{D_B^3}, \quad \left| \frac{\partial B_4}{\partial u_3} \right| < \frac{2}{D_B^3}. \]

To cover the other four partial derivatives, note that applying the Arithmetic Mean-Geometric Mean inequality to squares gives
\[ 2\sqrt{rs} \leq r + s \implies 2|rs| \leq r^2 + s^2. \]

Hence
\[ 2 |(u_1 + a) u_3| \leq (u_1 + a)^2 + u_3^2 = D_A^2 \]
\[ \implies \left| \frac{\partial A_2}{\partial u_3} \right| = 3 \frac{|(u_1 + a) u_3|}{D_A^5} \leq 3 \frac{1}{2} \frac{D_A^2}{D_A^5} = 3 \frac{1}{2} \frac{D_A^2}{D_A^5}. \]

Again, applying similar logic gives
\[ \left| \frac{\partial A_4}{\partial u_1} \right| < \frac{3}{2} \frac{1}{D_A^5}, \quad \left| \frac{\partial B_2}{\partial u_3} \right| < \frac{3}{2} \frac{1}{D_B^5}, \quad \left| \frac{\partial B_4}{\partial u_1} \right| < \frac{3}{2} \frac{1}{D_B^5}. \]

Putting it all together and defining
\[ M_A = \max \frac{1}{D_A^3} = \frac{1}{(\min D_A)^3}, \quad M_B = \max \frac{1}{D_B^3} = \frac{1}{(\min D_B)^3} \]
we find
\[
|A_2(u_1, u_3) - A_2(\hat{u}_1, \hat{u}_3)| \leq |u_1 - \hat{u}_1| \max \left| \frac{\partial A_2}{\partial u_1} \right| + |u_3 - \hat{u}_3| \max \left| \frac{\partial A_2}{\partial u_3} \right| \\
< |u_1 - \hat{u}_1| \max \left( \frac{2}{D_A^3} + \frac{3}{2} \frac{1}{D_A^3} \right) + |u_3 - \hat{u}_3| \max \left( \frac{2}{D_A^3} + \frac{3}{2} \frac{1}{D_A^3} \right)
\]
\[
= M \left( \frac{3}{2} |u_1 - \hat{u}_1| + \frac{3}{2} |u_3 - \hat{u}_3| \right)
\]
\[
|A_4(u_1, u_3) - A_4(\hat{u}_1, \hat{u}_3)| < M \left( \frac{3}{2} |u_1 - \hat{u}_1| + 2 |u_3 - \hat{u}_3| \right)
\]
\[
|B_2(u_1, u_3) - B_2(\hat{u}_1, \hat{u}_3)| < M_B \left( \frac{3}{2} |u_1 - \hat{u}_1| + \frac{3}{2} |u_3 - \hat{u}_3| \right)
\]
\[
|B_4(u_1, u_3) - B_4(\hat{u}_1, \hat{u}_3)| < M_B \left( \frac{3}{2} |u_1 - \hat{u}_1| + 2 |u_3 - \hat{u}_3| \right).
\]

Now we can return to our bounds on the \( f_2 \) and \( f_4 \) errors
\[
|f_2(t, u) - f_2(t, \hat{u})| \leq |u_1 - \hat{u}_1| + 2 |u_4 - \hat{u}_4| + b |A_2(u_1, u_3) - A_2(\hat{u}_1, \hat{u}_3)| + a |B_2(u_1, u_3) - B_2(\hat{u}_1, \hat{u}_3)|
\]
\[
= \left( 1 + 2bM_A + 2bM_B \right) |u_1 - \hat{u}_1| + \frac{3}{2} (bM_A + aM_B) |u_3 - \hat{u}_3| + 2 |u_4 - \hat{u}_4|
\]
\[
|f_4(t, u) - f_4(t, \hat{u})| \leq |u_3 - \hat{u}_3| + 2 |u_2 - \hat{u}_2|
\]
\[
= \left( 3 \frac{2}{2} |u_1 - \hat{u}_1| + 2 |u_3 - \hat{u}_3| \right) + aM_B \left( \frac{3}{2} |u_1 - \hat{u}_1| + 2 |u_3 - \hat{u}_3| \right)
\]
\[
= \frac{3}{2} (bM_A + aM_B) |u_1 - \hat{u}_1| + 2 |u_2 - \hat{u}_2| + \left( 1 + 2bM_A + 2bM_B \right) |u_3 - \hat{u}_3|.
\]

Bringing this back to the \( L^1 \) norm of the vector difference
\[
\|f(t, u) - f(t, \hat{u})\|_1 = |u_2 - \hat{u}_2| + |u_4 - \hat{u}_4|
\]
\[
= |f_2(t, u) - f_2(t, \hat{u})| + |f_4(t, u) - f_4(t, \hat{u})|
\]
\[
= \left( \frac{3}{2} |u_1 - \hat{u}_1| + 2 |u_3 - \hat{u}_3| \right) + aM_B \left( \frac{3}{2} |u_1 - \hat{u}_1| + 2 |u_3 - \hat{u}_3| \right)
\]
\[
= \left( 1 + \frac{7}{2} (bM_A + aM_B) \right) |u_1 - \hat{u}_1| + 3 |u_2 - \hat{u}_2| + \left( 1 + \frac{7}{2} (bM_A + aM_B) \right) |u_3 - \hat{u}_3| + 3 |u_4 - \hat{u}_4|.
\]

Now, if we let
\[
L = \max \left\{ 3, 1 + \frac{7}{2} (bM_A + aM_B) \right\}
\]
then we know
\[
\|f(t, u) - f(t, \hat{u})\|_1 \leq L \left( |u_1 - \hat{u}_1| + |u_2 - \hat{u}_2| + |u_3 - \hat{u}_3| + |u_4 - \hat{u}_4| \right) = L \|u - \hat{u}\|_1.
\]
Thus it remains to find a rectangular region containing \( u_0 = \begin{bmatrix} 0.994 \\ 0 \\ 0 \\ -2.00158510637908 \end{bmatrix} \) such that

\[
L = \max \left\{ 3, 1 + \frac{7}{2} \left( bM_A + aM_B \right) \right\}
\]

is bounded. Note that the values \( M_A \) and \( M_B \) really only depend on \( u_1, u_3 \) so those are the components we focus on. With \( a = 0.012277471, b = 1 - a = 0.987722529 \), we have

\[
D_A(u_0) = \sqrt{(u_1 + a)^2 + u_3^2} = \sqrt{(0.994 + 0.012277471)^2 + 0^2} = 1.006277471
\]

\[
D_B(u_0) = \sqrt{(u_1 - b)^2 + u_3^2} = \sqrt{(0.994 - 0.987722529)^2 + 0^2} = 0.006277471.
\]

As defined, \( D_B \) is the distance from the point \((u_1, u_3)\) to the point \((b, 0)\) while \( D_A \) is the distance to \((-a, 0)\). As we see above with \( D_B(u_0) \), the initial point is very near to \((b, 0)\) so this limits us to a very small rectangle in the \( u_1 \)-direction.

As long as we stay to the right of \( u_1 = b \), we’ll have \( D_B > 0 \) and \( D_A > 0 \), which means that \( M_A \left( = \max \frac{1}{D_A} \right) \) and \( M_B \) will be bounded. Thus we choose the rectangle

\[
(u_1, u_3) \in [0.99, 1.05] \times [-1, 1]
\]

In both cases, the point \((u_1, u_3) = (0.99, 0)\) is nearest to \((b, 0)\) and \((-a, 0)\) hence

\[
\min D_A = \sqrt{(0.99 + 0.012277471)^2 + 0^2} = 1.002277471
\]

\[
\min D_B = \sqrt{(0.99 - 0.987722529)^2 + 0^2} = 0.002277471
\]

which means

\[
M_A = \frac{1}{1.002277471^3} \approx 0.9931985905176525
\]

\[
M_B = \frac{1}{0.002277471^3} \approx 84652820.58251296
\]

\[
\Rightarrow bM_A + aM_B \approx 1039323.5307746296
\]

\[
\Rightarrow L = 1 + \frac{7}{2} (bM_A + aM_B) \approx 3637633.3577112034.
\]

We can choose an arbitrary rectangle in \( u_2u_4 \) space containing the initial points \((0, -2.00158510637908)\)

\[
(u_1, u_2, u_3, u_4) \in [0.99, 1.05] \times [-0.5, 0.5] \times [-1, 1] \times [-2.5, -1.5].
\]

**Question 2** Define a family of explicit Runge-Kutta methods parametrized by order \( p \), by extrapolating Euler’s method. i.e. for \( q = 1 \) to \( p \) define stages \( k_{qj} \) for \( 1 \leq j \leq q \) by

\[
k_{q1} = f(t_n, u_n),
\]

\[
k_{q2} = f(t_n + \frac{1}{q} h, u_n + h \cdot \frac{1}{q} k_{q1}),
\]

\[
k_{q3} = f(t_n + \frac{2}{q} h, u_n + h \left( \frac{1}{q} k_{q1} + \frac{1}{q} k_{q2} \right)),
\]
and so forth. Each of the \( p \) approximate solutions

\[
u_{n+1}^q = u_n + \frac{1}{q} h (k_{q1} + \cdots + k_{qq})
\]
is the result of \( q \) steps of Euler with step size \( \frac{h}{q} \), so

\[
u_{n+1}^q = U_p \left( \frac{h}{q} \right) + O \left( \frac{h^{p+1}}{q} \right)
\]
where \( U_p(h) \) is a polynomial of degree \( p \) in \( h \). Apply Richardson extrapolation (i.e. degree \( p - 1 \) polynomial interpolation from \( u_{n+1}^1 \) at \( h \), \( u_{n+1}^2 \) at \( h/2 \), \( u_{n+1}^3 \) at \( h/3 \), and so forth, to \( u_{n+1} \) at \( h = 0 \)) to the collection of approximate solutions \( u_{n+1}^q \) to design a Runge-Kutta method

\[
u_{n+1} = U_p(0) = u_n + h \left( \sum_{q=1}^{p} \sum_{j=1}^{q} h_{jq} k_{qj} \right)
\]
which is accurate of order \( p \).

(a) Verify that \( p = 1 \) gives Euler’s method and \( p = 2 \) gives the explicit midpoint rule

\[
u_{n+1} = u_n + hf \left( t_n + \frac{1}{2} h, u_n + \frac{1}{2} f(t_n, u_n) \right).
\]

(b) Write out the method that you get for \( p = 3 \).

(c) For arbitrary \( p \), verify that your method gives the correct Taylor expansion up to order \( p \) when you solve \( y' = y \).

**Proof.**

(a) When \( p = 1 \) we have

\[
k_{11} = f \left( t_n, u_n \right), \quad u_{n+1}^1 = u_n + h k_{11}.
\]
The interpolating polynomial satisfying \( U_1(h) = u_{n+1}^1 \) must be \( U_1(x) = u_{n+1}^1 \) (constant) hence the method

\[
u_{n+1} = U_1(0) = u_n + hf \left( t_n, u_n \right)
\]
is indeed Euler’s method.

When \( p = 2 \),

\[
k_{21} = k_{11}, \quad k_{22} = f \left( t_n + \frac{h}{2}, u_n + \frac{h}{2} k_{21} \right).
\]

Note that \( k_{22} \) is the slope estimate used in the explicit midpoint rule. The interpolating polynomial satisfying \( U_2(h) = u_{n+1}^1 \) and \( U_2 \left( \frac{h}{2} \right) = u_{n+1}^2 \) is

\[
U_2(x) = u_{n+1}^1 \frac{x - h}{h - \frac{h}{2}} + u_{n+1}^2 \frac{x - \frac{h}{2}}{\frac{h}{2}} = -u_{n+1}^1 + 2u_{n+1}^2.
\]

This gives the scheme

\[
u_{n+1} = - \left( u_n + h k_{11} \right) + 2 \left( u_n + \frac{h}{2} k_{21} + \frac{h}{2} k_{22} \right) = u_n + h \left( -k_{11} + k_{21} + k_{22} \right).
\]

Since \( k_{21} = k_{11} \) this means that we have

\[
u_{n+1} = u_n + h k_{22} = u_n + hf \left( t_n + \frac{h}{2}, u_n + \frac{h}{2} f(t_n, u_n) \right)
\]
which is the explicit midpoint rule as expected.
(b) For \( p = 3 \), we have \( k_{31} = k_{11} \),
\[
k_{32} = f \left( t_n + \frac{h}{3}, u_n + \frac{h}{3} k_{31} \right), \quad k_{33} = f \left( t_n + \frac{2h}{3}, u_n + \frac{h}{3} k_{31} + \frac{h}{3} k_{32} \right)
\]
The interpolating polynomial satisfying \( U_3(h) = u_{n+1}^1, U_3 \left( \frac{h}{2} \right) = u_{n+1}^2 \) and \( U_3 \left( \frac{3h}{2} \right) = u_{n+1}^3 \) is
\[
U_3(x) = u_{n+1}^1 \frac{(x - \frac{h}{2}) (x - \frac{3h}{2})}{\left( \frac{h}{2} - h \right) \left( \frac{3h}{2} - h \right)} + u_{n+1}^2 \frac{(x - h) (x - \frac{h}{2})}{\left( \frac{2h}{3} - h \right) \left( \frac{h}{2} - \frac{h}{3} \right)} + u_{n+1}^3 \frac{(x - h) (x - \frac{3h}{2})}{\left( \frac{3h}{2} - h \right) \left( \frac{3h}{2} - \frac{3h}{3} \right)} \Rightarrow U_3(0) = u_{n+1}^1 \left( \frac{h}{2} \right) \left( \frac{h}{3} \right) + u_{n+1}^2 \left( -h \right) \left( \frac{h}{3} \right) + u_{n+1}^3 \left( -h \right) \left( \frac{h}{3} \right) \Rightarrow U_3(0) = \frac{1}{2} u_{n+1}^1 - 4 u_{n+1}^2 + \frac{9}{2} u_{n+1}^3 = \frac{1}{2} (u_n + h k_{11}) - 4 \left( u_n + \frac{h}{2} k_{21} + \frac{h}{2} k_{22} \right) + \frac{9}{2} \left( u_n + \frac{h}{3} k_{31} + \frac{h}{3} k_{21} + \frac{h}{3} k_{22} \right) \Rightarrow u_n + h \left( \frac{1}{2} k_{11} - 2 k_{21} - 2 k_{22} + \frac{3}{2} k_{31} + \frac{3}{2} k_{21} + \frac{3}{2} k_{22} \right) = u_n + h \left( \frac{1}{2} k_{11} - 2 k_{21} + \frac{3}{2} k_{31} \right).
\]
Since \( k_{11} = k_{21} = k_{31} \) we have \( \frac{1}{2} k_{11} - 2 k_{21} + \frac{3}{2} k_{31} = 0 \) and the above simplifies to
\[
u_{n+1} = U_3(0) = u_n + h \left( -2 k_{22} + \frac{3}{2} k_{31} + \frac{3}{2} k_{31} \right).
\]

(c) When \( y' = y \) we have \( f(t,y) = y \). Note that this has solution \( y(t) = y(0) e^t \). In particular,
\[
y(t_n + h) = y(0) e^{t_n + h} = y(0) e^{t_n} e^h = y(t_n) e^h.
\]
In order to show that the method gives the correct Taylor expansion up to order \( p \), we want to show that \( u_{n+1} \approx e^h u_n \) (as in the true solution) but with \( e^h \) approximated by a truncated Taylor expansion. So when \( p = 1 \) this means we want
\[
u_{n+1} = (1 + h) u_n,
\]
when \( p = 2 \)
\[
u_{n+1} = \left( 1 + h + \frac{h^2}{2} \right) u_n,
\]
and for general \( p \)
\[
u_{n+1} = \left( 1 + h + \frac{h^2}{2} + \cdots + \frac{h^p}{p!} \right) u_n.
\]
For general \( j \leq q \)
\[
k_{q,j} = f \left( t_n + \frac{(j-1)h}{q}, u_n + \frac{h}{q} k_{q1} + \frac{h}{q} k_{q2} + \cdots + \frac{h}{q} k_{q,j-1} \right)
\]
\[
= u_n + \frac{h}{q} k_{q1} + \frac{h}{q} k_{q2} + \cdots + \frac{h}{q} k_{q,j-1}
\]
\[
\Rightarrow k_{q2} = u_n + \frac{h}{q} k_{q1} = u_n + \frac{h}{q} u_n = \left( 1 + \frac{h}{q} \right) u_n
\]
\[
\Rightarrow k_{q3} = u_n + \frac{h}{q} k_{q1} + \frac{h}{q} k_{q2} = k_{q2} + \frac{h}{q} k_{q2}
\]
\[
= \left( 1 + \frac{h}{q} \right) k_{q2} = \left( 1 + \frac{h}{q} \right)^2 u_n.
\]
We can track this process indefinitely and in general have
\[ k_{q,j} = k_{q,j-1} + \frac{h}{q} k_{q,j-1} = \left(1 + \frac{h}{q}\right) k_{q,j-1} = \left(1 + \frac{h}{q}\right)^{j-1} u_n. \]

At the final stage, we use the same logic
\[ u_{n+1}^q = u_n + \frac{h}{q} k_{q1} + \frac{h}{q} k_{q2} + \cdots + \frac{h}{q} k_{qq} = k_{qq} + \frac{h}{q} k_{qq} = \left(1 + \frac{h}{q}\right)^q u_n. \]

We seek to interpolate these values to compute
\[ U_p(x) = u_{n+1}^1 L_{1,p}(x) + u_{n+1}^2 L_{2,p}(x) + \cdots + u_{n+1}^p L_{p,p}(x) \]
\[ = (1 + h) u_n L_{1,p}(x) + \left(1 + \frac{h}{2}\right)^2 u_n L_{2,p}(x) + \cdots + \left(1 + \frac{h}{p}\right)^p u_n L_{p,p}(x). \]

We’ll show that
\[ (1 + h) L_{1,p}(0) + \left(1 + \frac{h}{2}\right)^2 L_{2,p}(0) + \cdots + \left(1 + \frac{h}{p}\right)^p L_{p,p}(0) = 1 + h + \frac{h^2}{2} + \cdots + \frac{h^p}{p!} \]
by splitting each \( \left(1 + \frac{h}{q}\right)^q \) into powers of \( h \) and then computing the coefficient of each \( h^q \).

We have
\[ g_p(h) = \sum_{q=1}^{p} \left(1 + \frac{h}{q}\right)^q L_{q,p}(0) \]
and expanding each power via the Binomial theorem
\[ g_p(h) = \sum_{q=1}^{p} \sum_{j=0}^{q} \binom{q}{j} \left(\frac{h}{q}\right)^j L_{q,p}(0) = \sum_{j=0}^{p} \sum_{q=j}^{p} \binom{q}{j} \left(\frac{h}{q}\right)^j L_{q,p}(0). \]

From here, we’ll show that
\[ \sum_{q=j}^{p} \binom{q}{j} \left(\frac{h}{q}\right)^j L_{q,p}(0) = \frac{h^j}{j!}, \]

Once we do that, we’ll have
\[ g_p(h) = \sum_{j=0}^{p} \left[ \sum_{q=j}^{p} \binom{q}{j} \left(\frac{h}{q}\right)^j L_{q,p}(0) \right] = \sum_{j=0}^{p} \frac{h^j}{j!} = 1 + h + \frac{h^2}{2} + \cdots + \frac{h^p}{p!} \]
as desired. First, when \( j = 0 \)
\[ \sum_{q=0}^{p} \binom{q}{0} \left(\frac{h}{q}\right)^0 L_{q,p}(0) = \sum_{q=0}^{p} 1 \cdot L_{q,p}(0) = 1 = \frac{h^0}{0!}. \]

This is because the interpolating polynomial \( P(x) = 1 \cdot L_{1,p}(x) + 1 \cdot L_{2,p}(x) + \cdots \) interpolates 1 at all values, hence \( P(x) \equiv 1 \), in particular \( P(0) = 1 \).

By similar logic, the function
\[ P(x) = \sum_{q=1}^{p} \left(\frac{h}{q}\right)^j L_{q,p}(x) \]
satisfies \( P \left( \frac{h}{q} \right) = \left(\frac{h}{q}\right)^j \) hence it agrees with the function \( x^j \). Since the \( p \) inputs \( h, \frac{h}{2}, \ldots, \frac{h}{p} \) unique define an interpolating polynomial of degree \( p - 1 \), we must have \( P(x) = x^j \) whenever \( j \leq p - 1 \).
This is useful because it tells us that
\[ \sum_{q=1}^{p} \left( \frac{h}{q} \right)^j L_{q,p}(0) = 0^j = 0 \]
(at least when \( j > 0 \), we already saw that this sum is 1 when \( j = 0 \)).

Returning to our original goal, we want
\[ \sum_{q=1}^{p} \left( \frac{h}{q} \right)^j L_{q,p}(0) = \frac{h^j}{j!} \]

Note that
\[ \sum_{q=1}^{p} \left( \frac{h}{q} \right)^j L_{q,p}(0) = \sum_{q=1}^{p} \frac{q(q-1)\cdots(q-j+1)}{j!} \left( \frac{h}{q} \right)^j L_{q,p}(0) \]
\[ = \sum_{q=1}^{p} \frac{q(q-1)\cdots(q-j+1)}{j!} L_{q,p}(0) \]
(since \( q = 1, \ldots j - 1 \implies q(q-1)\cdots(q-j+1) = 0 \) and 0 terms don’t change the sum)
\[ = \sum_{q=1}^{p} \frac{q^j + \text{lower order terms}}{j!} \left( \frac{h}{q} \right)^j L_{q,p}(0). \]

By what we showed above
\[ \sum_{q=1}^{p} \frac{q^j}{j!} \left( \frac{h}{q} \right)^j L_{q,p}(0) = \frac{h^j}{j!} \sum_{q=1}^{p} 1 \cdot L_{q,p}(0) = \frac{h^j}{j!} \cdot 1. \]

For any of the lower order terms in
\[ q(q-1)\cdots(q-j+1) = q^j + \text{lower order terms} = q^j + A_{j-1}q^{j-1} + \cdots A_1q + A_0 \]
we contribute
\[ \sum_{q=1}^{p} A_k q^k \left( \frac{h}{q} \right)^j L_{q,p}(0) = \frac{A_k h^k}{j!} \sum_{q=1}^{p} \left( \frac{h}{q} \right)^j L_{q,p}(0) = \frac{A_k h^k}{j!} \cdot 0 = 0. \]

Thus
\[ \sum_{q=1}^{p} \left( \frac{h}{q} \right)^j L_{q,p}(0) = \left[ \sum_{q=1}^{p} q^j \left( \frac{h}{q} \right)^j L_{q,p}(0) \right] + \left[ \sum_{q=1}^{p} \text{lower order terms} \frac{h^j}{j!} \left( \frac{h}{q} \right)^j L_{q,p}(0) \right] \]
\[ = \frac{h^j}{j!} + [0 + 0 + \cdots + 0] = \frac{h^j}{j!} \]
as desired. ■

Question 3

(a) Write, test and debug a MATLAB function

```matlab
function yb = rich(a, b, ya, f, p, n)
% a, b: interval endpoints with a < b
% ya: vector y(a) of initial conditions
% f: function handle f(t, y) to integrate (y is a vector)
% p: order of accuracy of the method
% n: number of steps with h = (b-a)/n
% yb: output approximation to the final solution vector y(b)
```
which approximates the final solution vector $y(b)$ of the vector initial value problem

$$y' = f(t, y)$$
$$y(a) = y_a$$

by the numerical solution vector $u_n$ of the method you derived in problem 2, with $h = \frac{b-a}{n}$ and $u_0 = y_a$.

(b) Use rich.m with a selection of orders $p$ and step sizes $N$ to approximate the final solution vector $u(T)$ of the initial value problem derived in problem 1.

Tabulate the errors

$$E_{p,N} = \max_{1 \leq j \leq 4} |u_j(T) - u_j(0)|$$

with time steps $h = \frac{T}{N}$ for $N = 100, 200, \ldots$. Estimate the constant $C_p$ such that the error behaves like $C_p h^p$.

Measure the CPU time for each run and estimate the total CPU time necessary to obtain an orbit which is periodic to three-digit, six-digit and twelve-digit accuracy.

Plot some inaccurate solutions and some accurate solutions and draw conclusions about values of the order $p$ which give three, six or twelve digits of accuracy at minimal cost (i.e. CPU time).

Find a rectangular region in the 4-dimensional $u$ space which includes an accurate orbit, and find an approximate Lipschitz constant $L$ for $f$ on that region. Draw conclusions about uniqueness of the periodic orbit.

Proof. (a) See the solution rich.m on the course webpage. This method has 3 core helpers interpolationCoeffs.m, richSingleStep.m and euler.m (used only by richSingleStep.m). To verify the correctness, the example scripts richExample.m and richCircleExample.m are provided.

(b) We use $p = 1, 2, 3, 4, 5, 6, 7$ and $N = 4000, 5000, 6000, 7000, 8000$ to get a sense of how the method converges to a periodic solution.

We provide the function richToTheMoon.m which takes $p$ as a single argument and finds the errors with that $p$ and $N = 4000, 5000, 6000, 7000, 8000$. Then it approximates the slopes in the fit lines

$$\text{error} = C \cdot (\text{step size}), \quad \text{duration} = S \cdot (\text{number of steps})$$

and uses those slopes as in PS8 to determine approximate runtimes for three-digit, six-digit and twelve-digit accuracy.

```matlab
>> richToTheMoon(1)
p: 1
errorSlope: 87229.6
timeSlope: 5.0076e-05
threeDigitDuration: 74542.8
sixDigitDuration: 7.45428e+07
twelveDigitDuration: 7.45428e+13
>> richToTheMoon(2)
p: 2
errorSlope: 165838
timeSlope: 0.000146646
threeDigitDuration: 32.2274
sixDigitDuration: 1019.12
twelveDigitDuration: 1.01912e+06
>> richToTheMoon(3)
p: 3
errorSlope: 3.73133e+07
timeSlope: 0.000230582
```
threeDigitDuration: 13.149
sixDigitDuration: 131.49
twelveDigitDuration: 13149
>> richToTheMoon(4)
   p: 4
   errorSlope: 7.31901e+09
   timeSlope: 0.000330682
threeDigitDuration: 9.28187
sixDigitDuration: 52.1958
twelveDigitDuration: 1650.58
>> richToTheMoon(5)
   p: 5
   errorSlope: 2.95333e+11
   timeSlope: 0.000287147
threeDigitDuration: 3.83953
sixDigitDuration: 15.2855
twelveDigitDuration: 242.258
>> richToTheMoon(6)
   p: 6
   errorSlope: 1.04015e+14
   timeSlope: 0.00037814
threeDigitDuration: 4.42534
sixDigitDuration: 13.9941
twelveDigitDuration: 139.941
>> richToTheMoon(7)
   p: 7
   errorSlope: 9.26974e+15
   timeSlope: 0.000724862
threeDigitDuration: 6.33794
sixDigitDuration: 17.0028
twelveDigitDuration: 122.366

The trajectories are provided below:
as well as the points used to approximate the error / duration fit lines:
As drawing conclusions about inaccurate vs. accurate solutions, it’s clear that higher $p$ leads to much faster convergence and allows for larger steps to be used (i.e. lower $N$).

To find a rectangular region in the 4-dimensional $u$ space which includes an accurate orbit, we run `richMoonFinalRegion.m` which produces

and fits in the region

$$(x, x', y, y') \in [-1.25, 1.0] \times [-1.2, 1.2] \times [-1.15, 1.15] \times [-2.05, .75]$$

To find an approximate Lipschitz constant $L$ for $f$ on that region, note that

$$\|f(t,u) - f(t,\tilde{u})\| \leq L \|u - \tilde{u}\| \implies \frac{\|f(t,u) - f(t,\tilde{u})\|}{\|u - \tilde{u}\|} \leq L.$$

Thus if we take $u, \tilde{u}$ as random vectors in our rectangular region and take the maximum ratio $\frac{\|f(t,u) - f(t,\tilde{u})\|}{\|u - \tilde{u}\|}$ over all pairs of random vectors as our constant $L$.

By running `monteCarloLipschitz.m` with $10^5$ and $10^6$ samples we get:
>> monteCarloLipschitz
100000 samples
Max Lipschitz: 177695
Duration: 3.19726
>> monteCarloLipschitz
1000000 samples
Max Lipschitz: 3.72266e+06
Duration: 39.0007

The second constant $\approx 3.7 \times 10^6$ seems to agree with our computed value above: 3637633.3577112034.
(Though note the script uses the $L^2$ norm while our computation above used the $L^1$ norm.)

From this we conclude that the uniqueness of the periodic orbit is very sensitive to changes in the initial condition. ■