Problem 1 (a) For arbitrary real s find the exact solution of the initial value problem

$$y'(t) = \frac{1}{2}\left(y(t) + y(t)^3\right)$$

with $y(0) = s > 0$.

(b) Show that the solution blows up when $t = \log(1 + 1/s^2)$.

Problem 2 (a) Find the general solution of the difference equation

$$u_{j+2} = u_{j+1} + u_j.$$

(b) Find all initial values u_0 and u_1 such that u_j remains bounded by a constant as $j \to \infty$.

Problem 3 (a) Write, test and debug a matlab function

```
function u = euler(a, b, ya, f, r, n)
% a,b: interval endpoints with a < b
% n: number of steps with h = (b-a)/n
% ya: vector y(a) of initial conditions
% f: function handle f(t, y, r) to integrate
% r: parameters to f
% u: output approximation to the final solution vector y(b)
```

which approximates the final solution vector $y(b)$ of the vector initial value problem

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

by the numerical solution vector u_n of Euler's method

$$u_{j+1} = u_j + hf(t_j, u_j, r) \qquad j = 0, 1, \ldots, n-1$$

with $h = (b-a)/n$ and $u_0 = y_a$.

(b) Use `euler.m` to approximate the solution $z(T)$ at $T = 4\pi$ of the initial value problem

$$z' = \begin{bmatrix} x \\ y \\ u \\ v \end{bmatrix}' = f(t, z) = \begin{bmatrix} u \\ v \\ -x/(x^2+y^2) \\ -y/(x^2+y^2) \end{bmatrix}$$

with initial conditions

$$z = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 1 \end{bmatrix}$$

at $t = 0$ which cause the solution to move in a unit circle forever. Measure the maximum error

$$E_N = \max(|x_N - \cos t_N|, |y_N - \sin t_N|, |u_N + \sin t_N|, |v_N - \cos t_N|)$$

after 2 revolutions ($T = 4\pi$) with time steps $h = T/N$ for $N = 1000$, 2000, $\ldots$, 16000. Estimate the constant C such that the error behaves like Ch. Measure the CPU time for each run and estimate the total CPU time necessary to obtain the solution to three–digit, six–digit and twelve–digit accuracy. Plot the solutions.

(c) Use `euler.m` with $s = [512, 64, 8, 1]$ and $N = [10^3, 10^4, 10^5, 10^6]$ to verify conclusion (b) of problem 1.

Problem 4 (See GGK 10.1) 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

$$x'' = x + 2y' - b\frac{x+a}{((x+a)^2+y^2)^{3/2}} - a\frac{x-b}{((x-b)^2+y^2)^{3/2}}$$

$$y'' = y - 2x' - b\frac{y}{((x+a)^2+y^2)^{3/2}} - a\frac{y}{((x-b)^2+y^2)^{3/2}}$$

where $a = 0.012277471$ and $b = 1 - a$. When the initial conditions

$$x(0) = 0.994$$

$$x'(0) = 0$$

$$y(0) = 0$$

$$y'(0) = -2.00158510637908$$

are satisfied, there is a periodic orbit with period $T = 17.06521656015796$.

(a) Convert this problem to a 4×4 *first-order* system $u' = f(t, u, r)$, $u(0) = u_0$, 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) Use `euler.m` to approximate $u(T)$ and plot the error vs. N for $N = 1000$, $2000, \ldots, 1024000$ steps. 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.

Problem 5 Suppose $y(t)$ is the exact solution of the initial value problem

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

$$y(0) = y_0,$$

and $u(t)$ is any approximation to $y(t)$ with $u(0) = y(0)$. Define the error $e(t) = y(t) - u(t)$.

(a) Show that $e(t)$ satisfies the initial value problem

$$e'(t) = f(t, u(t) + e(t)) - u'(t)$$

$$e(0) = 0$$

(b) Suppose $f(t, y) = \lambda y$ for some constant λ. Solve the initial value problem from (a) exactly to show that $u(t) + e(t) = y(t)$.

Problem 6 Define a family of explicit Runge-Kutta methods parametrized by order p, by applying $p-1$ passes of deferred correction to p steps of Euler's method. I.e. starting from u_n define the uncorrected solution by

$$u^1_{n+j+1} = u^1_{n+j} + hf(t_{n+j}, u^1_{n+j})$$

for $0 \leq j \leq p-1$. Let $u(t) = U_1(t)$ be the degree-p polynomial that interpolates the $p+1$ values u^1_{n+j} at the $p+1$ points $t = t_{n+j}$ for $0 \leq j \leq p$. Solve the error equation from question 5 by Euler's method, yielding approximate errors e^1_{n+1}, e^1_{n+2}, …, e^1_{n+p}. Produce a second-order accurate corrected solution

$$u^2_{n+j} = u^1_{n+j} + e^1_{n+j}$$

for $1 \leq j \leq p$. Repeat the procedure to produce u^2_{n+j}, …, u^p_{n+j}.

(a) Verify that $p = 1$ gives Euler's method.

(b) For $p = 2$ express your method as a Runge-Kutta method in the form

$$k_1 = f(t_n, u_n)$$

$$\begin{aligned} k_2 &= f(t_n + c_2 2h, u_n + 2ha_{21}k_1) \\ k_3 &= f(t_n + c_3 2h, u_n + 2h(a_{31}k_1 + a_{32}k_2)) \\ u_{n+2} &= u_n + 2h(b_1k_1 + b_2k_2 + b_3k_3). \end{aligned}$$

Find all the constants c_i, a_{ij} and b_j and arrange them in a Butcher array.

(c) For $p = 2$, ignore the t argument of $f(t, u)$ and Taylor expand $k_2(h)$ and $k_3(h)$ to $O(h^2)$. Show that your method has local truncation error $\tau = O(h^2)$ and find the coefficient of the $O(h^2)$ term.

(d) For arbitrary p, verify that your method is equivalent to using fixed point iteration to solve an implicit Runge-Kutta method.

Problem 7 Write, test and debug a matlab function

```
function yb = idec(a, b, ya, f, r, 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)
% r: parameters to f
% p: number of euler substeps / correction passes
% n: number of time steps
% yb: output approximation to the final solution vector y(b)
```

which approximates the final solution vector $y(b)$ of the vector initial value problem

$$y' = f(t, y, r)$$

$$y(a) = y_a$$

by the method you derived in problem 4, with $u_0 = y_a$.

(a) Use `idec.m` with orders $p = 1$ through 7 and $N = 10000$, 20000, 40000 and 80000 steps to approximate the final solution vector $u(T)$ of the initial value problem derived in problem 4. Tabulate the errors

$$E_{pN} = \max_{1 \le j \le 4} |u_j(T) - u_j(0)|.$$

Estimate the constant C_p such that the error behaves like $C_p h^p$.

(b) 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.

(c) 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 for minimal CPU time.