MATH 211.3 Winter Term 2024 Assignment

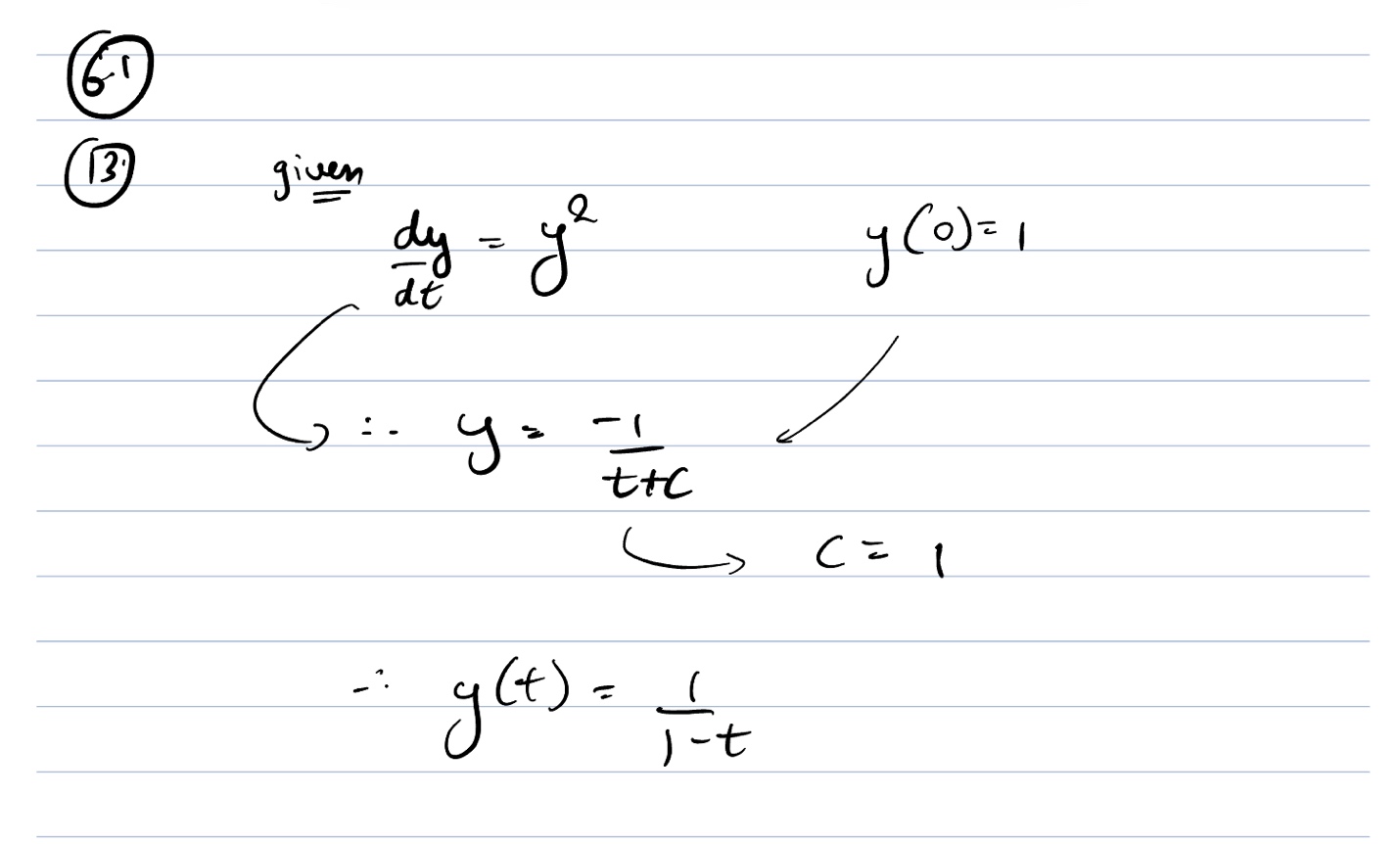
Assignment #08

Name: Kanwar Pannu

Student Number: 11336732

NSID: eza147

**PROBLEM 1**

****

**9**

clear;

clc;

dy = @(t, y) sin(y);

tSpan = [0, 4];

y0\_a = 0;

y0\_b = 100;

% Step sizes

h = 0.1 \* 2.^(-[0:5]);

errors\_a = zeros(length(h), 1);

errors\_b = zeros(length(h), 1);

for k = 1:length(h)

% Time vector for kth step size

t = 0:h(k):4;

% Euler's method approximation

[t\_a, y\_a] = eulersMethod(dy, tSpan, y0\_a, h(k));

[t\_b, y\_b] = eulersMethod(dy, tSpan, y0\_b, h(k));

if k == 1 || k == 6

figure;

plot(t\_a, y\_a, '-o', t\_b, y\_b, '-\*');

title(['Euler''s Method Approximation with h = ', num2str(h(k))]);

xlabel('Time t');

ylabel('y(t)');

legend('y0 = 0', 'y0 = 100');

end

end

% Log-log plot of the error at t = 4 as a function of h

% figure;

% loglog(h, errors\_a, '-o', h, errors\_b, '-\*');

% title('Log-log plot of the Error at t = 4');

% xlabel('Step size h');

% ylabel('Error at t = 4');

% legend('Error with y0 = 0', 'Error with y0 = 100');

function [t, y] = eulersMethod(dy, tSpan, y0, h)

t = tSpan(1):h:tSpan(2);

y = zeros(1, length(t));

y(1) = y0;

for i = 1:(length(t)-1)

y(i+1) = y(i) + h \* dy(t(i), y(i));

end

end

**A screenshot of a computer

Description automatically generated**

**A screenshot of a computer

Description automatically generated**

**PROBLEM 2**

**A close-up of a paper with mathematical equations

Description automatically generated**

**A close-up of a math equation

Description automatically generated**

**1**

clear;

clc;

h = 0.1;

t = 0:h:1;

y0\_a = 0;

y0\_b = 1;

y\_a = zeros(size(t));

y\_b = zeros(size(t));

y\_a(1) = y0\_a;

y\_b(1) = y0\_b;

f\_a = @(t, y) t;

f\_b = @(t, y) (t^2) \* y;

for i = 1:length(t)-1

% For equation (a)

y\_a(i+1) = y\_a(i) + (h/2) \* (f\_a(t(i), y\_a(i)) + f\_a(t(i+1), y\_a(i) + h\*f\_a(t(i), y\_a(i))));

% For equation (b)

y\_b(i+1) = y\_b(i) + (h/2) \* (f\_b(t(i), y\_b(i)) + f\_b(t(i+1), y\_b(i) + h\*f\_b(t(i), y\_b(i))));

end

y\_exact\_a = t.^2 / 2;

y\_exact\_b = exp(t.^3 / 3);

error\_a = abs(y\_exact\_a - y\_a);

error\_b = abs(y\_exact\_b - y\_b);

% Print the table for (a) and (b)

fprintf('Table for equation (a): y'' = t\n');

fprintf('t\tApproximation\tExact\t\tError\n');

for i = 1:length(t)

fprintf('%1.1f\t%1.4f\t\t\t%1.4f\t%1.4e\n', t(i), y\_a(i), y\_exact\_a(i), error\_a(i));

end

fprintf('\nTable for equation (b): y'' = t^2 \* y\n');

fprintf('t\tApproximation\tExact\t\tError\n');

for i = 1:length(t)

fprintf('%1.1f\t%1.4f\t\t%1.4f\t%1.4e\n', t(i), y\_b(i), y\_exact\_b(i), error\_b(i));

end

**A screenshot of a computer

Description automatically generated**

**7**

clear;

clc;

dydt = @(t, y) sin(y);

y0\_values = [0, 100];

tspan = [0, 4];

h\_values = 0.1 \* 2.^(-[0:5]);

errors = zeros(length(h\_values), length(y0\_values));

options = odeset('RelTol', 1e-8, 'AbsTol', 1e-10);

for j = 1:length(y0\_values)

[T\_exact, Y\_exact] = ode45(dydt, tspan, y0\_values(j), options);

figure;

hold on;

plot(T\_exact, Y\_exact, 'k-', 'LineWidth', 2, 'DisplayName', 'Numerical Solution');

for k = 1:length(h\_values)

h = h\_values(k);

[T\_approx, Y\_approx] = trapezoid(dydt, tspan, y0\_values(j), h);

% Calculate error at t=4

Y\_exact\_at\_4 = interp1(T\_exact, Y\_exact, 4);

errors(k, j) = abs(Y\_exact\_at\_4 - Y\_approx(end));

% Plot for k=0 and k=5

if k == 1 || k == length(h\_values)

plot(T\_approx, Y\_approx, 'DisplayName', ['Trapezoid h = ', num2str(h)]);

end

end

title(['Solution and Approximation for y0 = ', num2str(y0\_values(j))]);

xlabel('t');

ylabel('y');

legend show;

hold off;

end

figure;

loglog(h\_values, errors(:, 1), 'o-', 'DisplayName', 'Error with y0 = 0');

hold on;

loglog(h\_values, errors(:, 2), 'x-', 'DisplayName', 'Error with y0 = 100');

title('Log-Log Plot of Error at t=4');

xlabel('Step size h');

ylabel('Error at t=4');

legend show;

hold off;

function [t, y] = trapezoid(f, tspan, y0, h)

% Implementation of the Trapezoid Method

N = floor((tspan(2) - tspan(1)) / h);

t = linspace(tspan(1), tspan(2), N+1);

y = zeros(1, length(t));

y(1) = y0;

for i = 1:N

y\_pred = y(i) + h \* f(t(i), y(i));

y(i+1) = y(i) + (h/2) \* (f(t(i), y(i)) + f(t(i+1), y\_pred));

end

end

**A screenshot of a computer

Description automatically generated**

**A screen shot of a graph

Description automatically generated**

**A screen shot of a graph

Description automatically generated**

**PROBLEM 3**

**A close-up of a notebook

Description automatically generated**

**A close-up of a math problem

Description automatically generated**

**1**

clear;

clc;

inter = [0, 1];

y0 = [1, 0];

n1 = 10;

n2 = 100;

[t1, y1] = euler2(inter, y0, n1);

[t2, y2] = euler2(inter, y0, n2);

figure;

plot(t1, y1(:, 1), 'r', t1, y1(:, 2), 'b');

hold on;

plot(t2, y2(:, 1), 'g', t2, y2(:, 2), 'k');

title('Euler''s Method Approximate Solutions');

xlabel('t');

ylabel('y');

legend('y1 h=0.1', 'y2 h=0.1', 'y1 h=0.01', 'y2 h=0.01');

hold off;

function [t, y] = euler2(inter, y0, n)

t(1) = inter(1);

y(1, :) = y0;

h = (inter(2) - inter(1)) / n;

for i = 1:n

t(i + 1) = t(i) + h;

y(i + 1, :) = eulerstep(t(i), y(i, :), h);

end

plot(t, y(:, 1), t, y(:, 2));

end

function y = eulerstep(t, y, h)

y = y + h \* ydot(t, y);

end

function z = ydot(t, y)

z(1) = y(1) + y(2);

z(2) = -y(1) + y(2);

end

**A screen shot of a graph

Description automatically generated**

**3**

clear;

clc;

pend([0 10], [pi 0], 200, 0.1);

function pend(inter, ic, n, d)

h = (inter(2) - inter(1)) / n;

y = zeros(n+1, 2);

y(1, :) = ic;

t = inter(1):h:inter(2);

figure;

axes;

xlim([-1.2 1.2]);

ylim([-1.2 1.2]);

axis square;

xlabel('x');

ylabel('y');

hold on;

bob = plot(0, 0, 'ro', 'MarkerSize', 8, 'MarkerFaceColor', 'r');

rod = plot([0, 0], [0, -1], 'b-', 'LineWidth', 2);

for k = 1:n

y(k + 1, :) = trapstep(t(k), y(k, :), h, d);

xbob = sin(y(k + 1, 1));

ybob = -cos(y(k + 1, 1));

set(rod, 'xdata', [0, xbob], 'ydata', [0, ybob]);

set(bob, 'xdata', xbob, 'ydata', ybob);

drawnow;

pause(h);

end

end

function y = trapstep(t, x, h, d)

z1 = ydot(t, x, d);

g = x + h \* z1;

z2 = ydot(t + h, g, d);

y = x + h \* (z1 + z2) / 2;

end

function z = ydot(t, y, d)

g = 9.81; length = 1;

z = zeros(1,2);

z(1) = y(2);

z(2) = -(g / length) \* sin(y(1)) - d \* y(2);

end

**PROBLEM 4**

**A paper with math equations

Description automatically generated**

**1**

clear;

clc;

f = @(t, y) t^2 \* y;

y0 = 1;

tspan = [0, 1];

h = 0.1;

n = (tspan(2) - tspan(1)) / h;

t\_values = zeros(1, n+1);

y\_values\_midpoint = zeros(1, n+1);

errors = zeros(1, n+1);

t\_values(1) = tspan(1);

y\_values\_midpoint(1) = y0;

for i = 1:n

t = t\_values(i);

y = y\_values\_midpoint(i);

k1 = f(t, y);

k2 = f(t + h/2, y + (h/2)\*k1);

y\_next = y + h \* k2;

t\_next = t + h;

t\_values(i+1) = t\_next;

y\_values\_midpoint(i+1) = y\_next;

end

[t\_ode45, y\_ode45] = ode45(f, tspan, y0);

y\_accurate = interp1(t\_ode45, y\_ode45, t\_values);

errors = abs(y\_accurate' - y\_values\_midpoint);

fprintf('Step t Midpoint Approx Error\n');

fprintf('----------------------------------------\n');

for i = 1:length(t\_values)

fprintf('%4i %6.2f %16.8f %12.8f\n', i-1, t\_values(i), y\_values\_midpoint(i), errors(i));

end

**A screenshot of a computer code

Description automatically generated**

**5**

clear;

clc;

dydt = @(t, y) 1 + y^2;

y0\_values = [0, 1];

tspan = [0, 1];

h\_values = [0.1, 0.05];

for y0 = y0\_values

figure;

hold on;

for h = h\_values

[T, Y] = rk4(dydt, tspan, y0, h);

plot(T, Y, 'DisplayName', ['y0 = ', num2str(y0), ', h = ', num2str(h)]);

end

title(['Solution of y'' = 1 + y^2 with y0 = ', num2str(y0)]);

xlabel('t');

ylabel('y');

legend show;

hold off;

end

function [T, Y] = rk4(f, tspan, y0, h)

T = tspan(1):h:tspan(2);

if T(end) ~= tspan(2)

T = [T tspan(2)];

end

Y = zeros(size(T));

Y(1) = y0;

for i = 1:(length(T)-1)

ti = T(i);

yi = Y(i);

k1 = f(ti, yi);

k2 = f(ti + 0.5\*h, yi + 0.5\*h\*k1);

k3 = f(ti + 0.5\*h, yi + 0.5\*h\*k2);

k4 = f(ti + h, yi + h\*k3);

Y(i+1) = yi + (h/6)\*(k1 + 2\*k2 + 2\*k3 + k4);

end

end

**A screen shot of a graph

Description automatically generatedA screenshot of a computer

Description automatically generated**

**9**

clear;

clc;

dydt = @(t, y) 1 + y.^2;

y0\_values = [0, 1];

tspan = [0, 1];

y\_exact = zeros(length(y0\_values), 1);

for i = 1:length(y0\_values)

[~, Y] = ode45(dydt, tspan, y0\_values(i));

y\_exact(i) = Y(end);

end

h\_values = logspace(-4, -1, 20);

errors = zeros(length(h\_values), length(y0\_values));

for j = 1:length(y0\_values)

for i = 1:length(h\_values)

[~, Y] = rk4(dydt, tspan, y0\_values(j), h\_values(i));

errors(i, j) = abs(Y(end) - y\_exact(j));

end

end

figure;

loglog(h\_values, errors, 'LineWidth', 2);

xlabel('Step size h');

ylabel('Global error at t=1');

legend(arrayfun(@(y0) sprintf('y0 = %g', y0), y0\_values, 'UniformOutput', false), 'Location', 'NorthEast');

title('Global error of RK4 method at t=1 as a function of h');

grid on;

function [T, Y] = rk4(f, tspan, y0, h)

% Implementation of the RK4 method

T = tspan(1):h:tspan(2);

if T(end) < tspan(2)

T = [T, tspan(2)];

end

Y = zeros(1, length(T));

Y(1) = y0;

for i = 1:(length(T)-1)

ti = T(i);

yi = Y(i);

k1 = f(ti, yi);

k2 = f(ti + 0.5\*h, yi + 0.5\*h\*k1);

k3 = f(ti + 0.5\*h, yi + 0.5\*h\*k2);

k4 = f(ti + h, yi + h\*k3);

Y(i+1) = yi + (h/6)\*(k1 + 2\*k2 + 2\*k3 + k4);

end

Y = Y(end); % Return only the final value for global error computation

end

**A graph on a computer screen

Description automatically generated**