Effat University
Numerical Analysis Project

Ameera Attiah – S21107316

Leen Sharab – S21107195

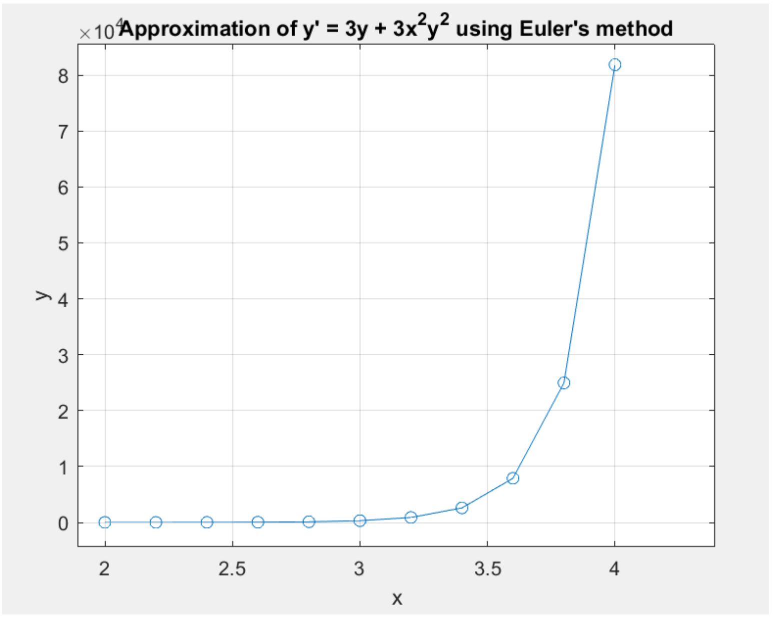
Instructor: Dr. Narjisse Kabbaj

Date: 29/05/2023

Euler's Method

$$y' = 3xy + x^2$$

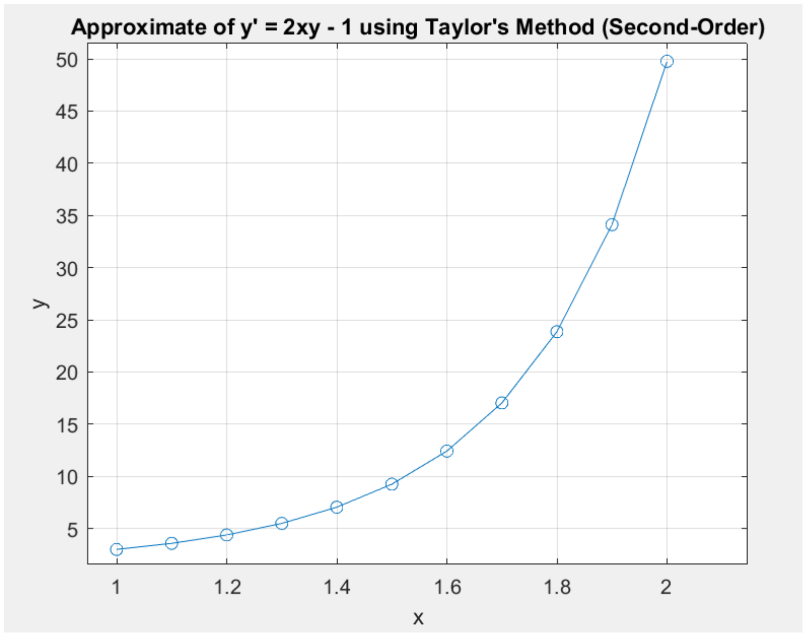
```
clear all
% Initial condition: y(1) = 1.5
f = @(x, y) 3*x*y + x^2;
% Define the initial condition
x0 = 2;     % Initial x-value
y0 = 3;   % Initial y-value
% Define the step size
h = 0.2;
% Define the number of steps
N = 10;     % Number of iterations
% Initialize arrays to store x and y values
x = zeros(N+1, 1);
y = zeros(N+1, 1);
% Set initial values
x(1) = x0;
y(1) = y0;
% Perform Euler's method iterations
for i = 1:N
    x(i+1) = x(i) + h;
    y(i+1) = y(i) + h * f(x(i), y(i));
end

% Plot the result
plot(x, y, '-o');
xlabel('x');
ylabel('y');
title("Approximation of y' = 3y + 3x^2y^2 using Euler's method");
grid on;
```



Taylor's Method

$$y' = 2xy - 1$$

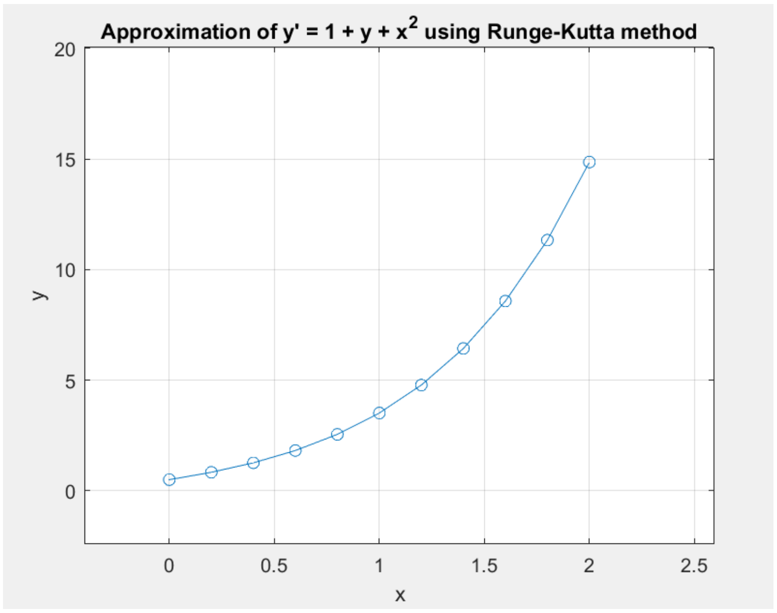
```
clear all
% Initial condition: y(1) = 3
% the first derivative
f1 = @(x, y) 2*x*y - 1;
% the second derivative
f2 = @(x, y) 2*y + 2*x*(2*x*y-1);
% Define the initial condition
x0 = 1;
y0 = 3;
% Define the step size and number of iterations
h = 0.1;
num_iterations = 10;
% Initialize arrays to store x and y values
x = zeros(num_iterations+1, 1);
y = zeros(num_iterations+1, 1);
% Set initial values
x(1) = x0;
y(1) = y0;
% Implement Taylor's second-order method
for i = 1:num_iterations
    x(i+1) = x(i) + h; % Update x
    y(i+1) = y(i) + h * f1(x(i), y(i)) + (h^2/2) * f2(x(i), y(i)); % Update y using
the second-order Taylor's method
end
% Plot the results
plot(x, y, '-o');
xlabel('x');
ylabel('y');
title("Approximate of y' = 2xy - 1 using Taylor's Method (Second-Order)");
grid on;
```



Runge Kutta’s Method

$$y' = 1 + y + x^2$$

```matlab
% Initial condition: y(0) = 0.5
% Define the differential equation function
f = @(x, y) 1 + y + x^2;
% Define the initial condition
x0 = 0;     % Initial x-value
y0 = 0.5;   % Initial y-value
% Define the step size
h = 0.2;
% Define the number of steps
N = 10;     % Number of iterations
% Initialize arrays to store x and y values
x = zeros(N+1, 1);
y = zeros(N+1, 1);
% Set initial values
x(1) = x0;
y(1) = y0;
% Perform Runge-Kutta iterations
for i = 1:N
    k1 = f(x(i), y(i));
    k2 = f(x(i) + h/2, y(i) + h * k1/2);
    k3 = f(x(i) + h/2, y(i) + h * k2/2);
    k4 = f(x(i) + h, y(i) + h * k3);

    y(i+1) = y(i) + h* (k1 + 2*k2 + 2*k3 + k4) / 6;
    x(i+1) = x(i) + h;
end
% Plot the result
plot(x, y, '-o');
xlabel('x');
ylabel('y');
title("Approximation of y' = 1 + y + x^2 using Runge-Kutta method");
grid on;
```


Approximation of y' = 1 + y + x^2 using Runge-Kutta method
y
x