Predator Vs Prey Simulation

Matthew Ginelli

April 2012

NOTE

The attached code, `examplecode.m` runs the three methods to produce a 2x1 subplot to include the Euler'sapproximation of each population along with the ode45 approximation and the second is to include the Runge-Kutta approximation with the ode45 approximation.

Code

```
function [] = predatorPreyPlot(alpha,beta,gamma,∆,x0,y0, ...
    tmin,tmax,n)
% function [] = predatorPreyPlot(alpha,beta,gamma,∆,x0,y0, ...
    tmin,tmax,n)
%
% predatorPreyPlot runs the three methods to produce a 2x1 ...
    subplot to
% include the Euler's approximation of each population along ...
    with the ode45
% approximation and the second is to include the Runge—Kutta ...
    approximation
% with the ode45 approximation.
%
% Matt Ginelli
% Lab 016 (Tiffany Deng)
% 26 April 2012

alpha = 1.5;
beta = 0.1;
gamma = 0.25;
∆ = 0.01;
tmax = 30;
tmin = 0;
x0 = 20;
```

```
y0 = 15;
n = 200;
clc

close all;

g = @(tode45,f) [f(1).*(alpha-beta*f(2));-f(2).*(gamma - Δ*f(1))];
[tode45, f] = ode45(g, [tmin, tmax], [x0, y0]);

numPreyODE45 = f(:,1); % set to the first column
numPredODE45 = f(:,2); % set to the second column



h = tmax/(n-1);
t = linspace(0,tmax,n);
x(1) = x0;  % set initial conditions for population of Prey
y(1) = y0;  % set initial conditions for population of Predator
dxdt = @(x,y) x.*(alpha-beta.*y);  % set the dxdt
dydt = @(x,y) -y.*(gamma - Δ.*x);

for i = 1:length(t) - 1
    x(i+1) = x(i) + h.*dxdt(x(i), y(i)); %Step using Euler's ...
        method for x
    y(i+1) = y(i) + h.*dydt(x(i), y(i));  %Step using Euler's ...
        method for y
end

numPreyEuler = x;
numPredEuler = y;

% end Euler's method

subplot(2,1,1)
plot(t,numPreyEuler,t,numPredEuler, tode45,numPreyODE45, ...
    'k--', tode45,numPredODE45, 'g--')
xlabel('Time (Dimensionless)')
ylabel('Population')
title({'Matt Ginelli - 22857816 - Tiffany Deng - Lab 016'; ...
    'Euler''s Method vs. ode45'})
legend('Prey-Euler', 'Pred-Euler', 'Prey-ode45', 'Pred-ode45', ...
    'Location', 'NorthWest')

t = linspace(0,tmax, n);
h = tmax/(n-1); % define step
dx = @(x,y) x.*(alpha-beta*y);
dy = @(x,y) -y.*(gamma - Δ*x);
x(1) = x0;
y(1) = y0;

for i = 1:length(t) - 1
    kdx1(i)=dx(x(i),y(i));
    kdy1(i)=dy(x(i),y(i));
    kdx2(i)=dx(((x(i)+((h/2)*(kdx1(i))))),(y(i)+((h/2)*(kdy1(i)))));
    kdy2(i)=dy(((x(i)+((h/2)*(kdx1(i))))),(y(i)+((h/2)*(kdy1(i)))));
    kdx3(i)=dx(((x(i)+((h/2)*(kdx2(i))))),(y(i)+((h/2)*(kdy2(i)))));
    kdy3(i)=dy(((x(i)+((h/2)*(kdx2(i))))),(y(i)+((h/2)*(kdy2(i)))));
```

```
    kdx4(i)=dx(((x(i)+(kdx3(i)*h))),(y(i)+(kdy3(i)*h)));
    kdy4(i)=dy(((x(i)+(kdx3(i)*h))),(y(i)+(kdy3(i)*h)));
    x(i+1)=x(i)+((h/6)*(kdx1(i)+2*kdx2(i)+2*kdx3(i)+kdx4(i)));
    y(i+1)=y(i)+((h/6)*(kdy1(i)+2*kdy2(i)+2*kdy3(i)+kdy4(i)));
end
numPreyRK4 = x;
numPredRK4 = y;

subplot(2,1,2)
plot(t,numPreyRK4, 'k', t,numPredRK4, 'r', tode45,numPreyODE45, ...
    'g--', tode45,numPredODE45, 'b--', 'LineWidth',1.2)
xlabel('Time (Dimensionless)')
ylabel('Population')
title('Runge-Kutta Method vs. ode45')
legend('Prey-RK4', 'Pred-RK4', 'Prey-ode45', 'Pred-ode45', ...
    'Location', 'NorthWest')
end
```

Figures

Below are the outputs from the simulations

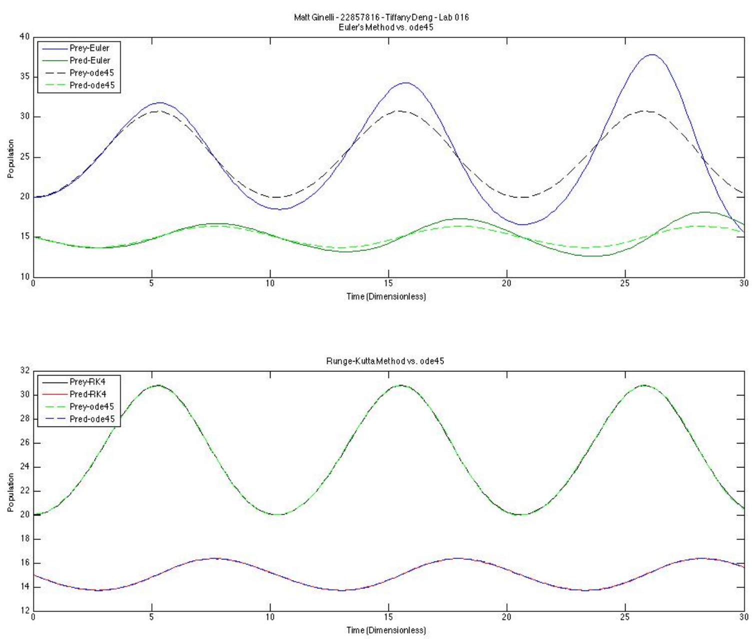

Figure 1: Simulator Results

Matthew F. Gienlli (`matt.uhs@gmail.com`)