PHYS 2030 (Winter 2018) - HW 3

Solutions

1. Consider the differential equation

$$\frac{dy}{dx} = y^2 - c$$

where c is a constant greater than zero.

(a.) Determine all equilibrium solutions and their stability.

Answer: Factoring the right-hand side of the differential equation, we can see that there will be equilibria solutions at $y = \pm\sqrt{c}$. There are many ways one could determine their stability (e.g. use DFIELD, linearize about the fixed point and find the eigenvalues, etc..).
One approach is to plot the phase-line portrait (Fig.4). The right-hand side is just a concave-up parabola and the points where it crosses the horizontal-axis indicates the equilibrium locations. Whether the curve is above or below the axis indicates the directionality of the phase-line. We can see that $y = \sqrt{c}$ is unstable and $y = -\sqrt{c}$ is a stable equilibrium.

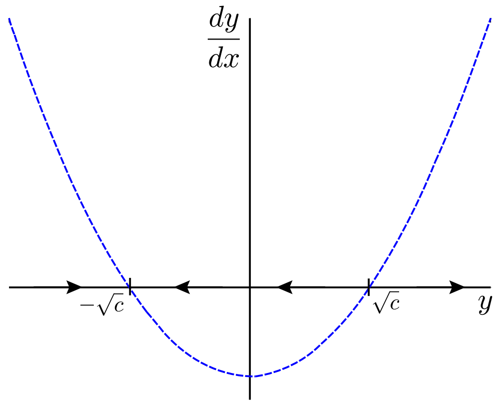


Figure 1: Phase-line portrait for $dy/dx = y^2 - c$. Note that the stability of the equilibria can be determined by whether the quadratic function is above or below the y-axis.

(b.) Solve this equation analytically to obtain an expression for $y(x)$. Your answer should depend upon c and contain an arbitrary constant.

Answer: Similar to question 3, we would use separation of variables, then integrate both sides. For this case, we could use a partial fraction expansion to simplify the denominator, yielding

$$\int \frac{dy}{y^2 - c} = \int \left[\frac{-1}{2\sqrt{c}} \frac{1}{(y + \sqrt{c})} + \frac{1}{2\sqrt{c}} \frac{1}{(y - \sqrt{c})} \right] dy = x + C$$

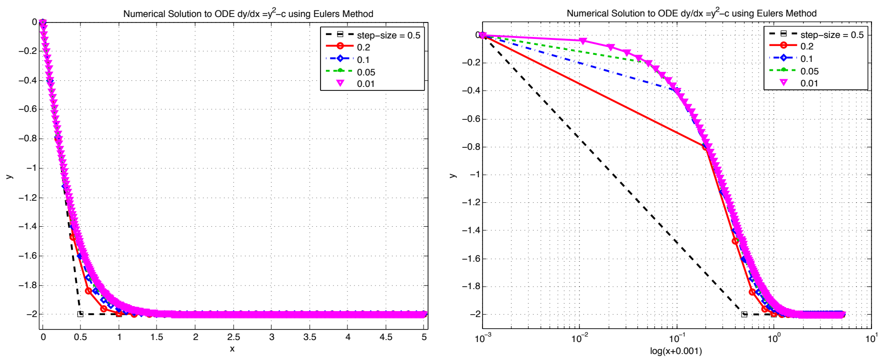


Figure 2: Comparison of exact solution for the differential equation $y' = y^2 - c$ with solution estimated numerically using Euler's method for different step-sizes. Also shown using a logarithmic x-axis (where the x values were slightly offset so to visualize the initial point).

The resulting integral is easily solved (make a substitution and use $\int \frac{du}{u} = \ln|u| + c$) and solving for y, we have the final solution

$$y(x) = \sqrt{c}\,\frac{1 + Ae^{2\sqrt{c}x}}{1 - Ae^{2\sqrt{c}x}}$$

where A is an arbitrary constant.

(c.) Write a code to solve the equation numerically using Euler's method on the interval $x \in [0, 5]$ for the initial condition $y(0) = 0$ and with $c = 4$. On a single figure, plot your estimated solution curve using the following step sizes for Δx: 0.5, 0.2, 0.1, 0.05, and 0.01. Make clear which curve corresponds to each step-size. How does the solution depend upon Δx?

Answer: An example code is given below as well as the figure (Fig. 5) that was asked for. The solution approaches the equilibrium value faster for smaller values of Δx.

```
% ### odeSOLVEptII.m ### 11.10.08

% Matlab code to use Euler's method to solve the differential equation
% y' = yˆ2 - c
% (where c is a positive const.)

clear
clf

% ————————————-
% User Input Parameters

xMIN= 0;      % starting x-value
xMAX= 5;      % ending x-value
deltaX= 0.01;      % step-size

% initial conditions
y0= 0;

% ————————————-

x0= xMIN;
nsteps= (xMAX-xMIN)/deltaX;
```

```
yS(1)= y0; xS(1)= x0;

for nn=2:nsteps+1

    % note the difference here from the code for Part I
    yS(nn) = yS(nn-1) + deltaX*(yS(nn-1)^2 - c);

    % update x-array
    xS(nn) = xS(nn-1)+ deltaX;

end

% plot the numerical solution
plot(xS,yS)
hold on;
xlabel('x')
ylabel('y')
title('Numerical Solution to ODE dy/dx =y^2-c using Eulers Method')

% plot exact (i.e., analytically-derived) solution (only true if y(0)=0)
yA= sqrt(c)*(1-exp(2*sqrt(c)*xS))./((1+exp(2*sqrt(c)*xS)));
% could also have used -sqrt(c)*tanh(sqrt(c)*xS) here too

plot(xS,yA,'rx')
grid on
legend('Eulers method','Exact','Location','SouthEast')
```

(d.) Using $\Delta x = 0.01$, find solution curves for different initial conditions $y(0) = y_o$. How do the solutions depend upon y_o?

Answer: See Fig.6.

(e.) Explain your answer to the last part in terms of your analytic solution. Are the two results consistent?

Answer: The constant A will depend upon the initial condition such that

$$A = \frac{y_o - \sqrt{c}}{y_o + \sqrt{c}}$$

where $y_o = y(0)$. Solutions will diverge towards $+\infty$ (for increasing x) if $y_o > \sqrt{c}$. Solutions will be sigmoidal when $|y_o| < \sqrt{c}$. The shape will be a reverse-S since solutions will move away from the unstable equilibrium (i.e., $\sqrt{c}$) and towards the stable one ($-\sqrt{c}$) as x increases. Decreasing y_o from around 2 towards -2 will move the center of the S-shape to the left. For $y_o < \sqrt{c}$, solution curves will asymptotically approach the stable equilibrium. When $|y_o| = \sqrt{c}$, the solutions will be constant (i.e., equilibrium solutions). Both the numerical ($\Delta x = 0.01$) and analytical solutions are consistent with one another, though the numerical solution will underestimate for the case $y_o > \sqrt{c}$ as the solution diverges.

Note that for the initial condition $y(0) = 0$, we have the solution

$$y(x) = \sqrt{c}\,\frac{1 - e^{2\sqrt{c}x}}{1 + e^{2\sqrt{c}x}} = -\sqrt{c}\tanh\left(\sqrt{c}x\right)$$

.

(f.) What is the effect of varying c? Explain in the contexts of both your analytical answer and numerical simulations. Do both agree?

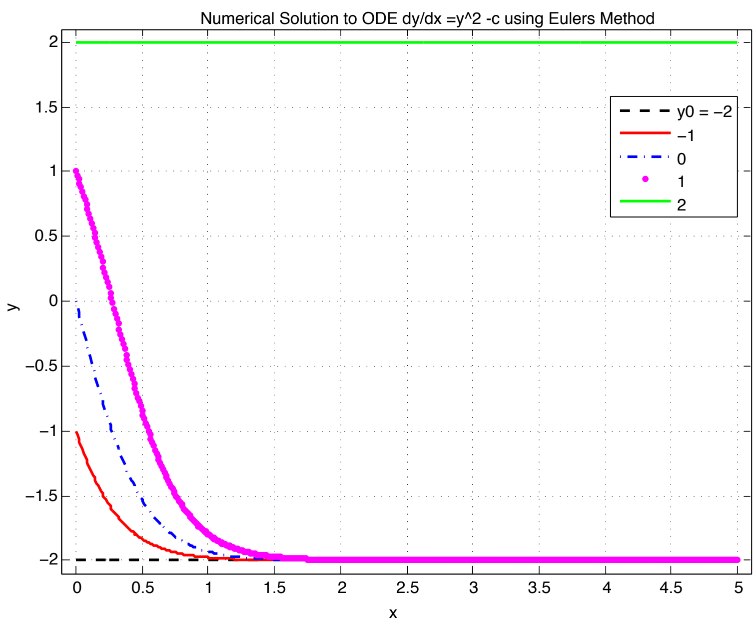


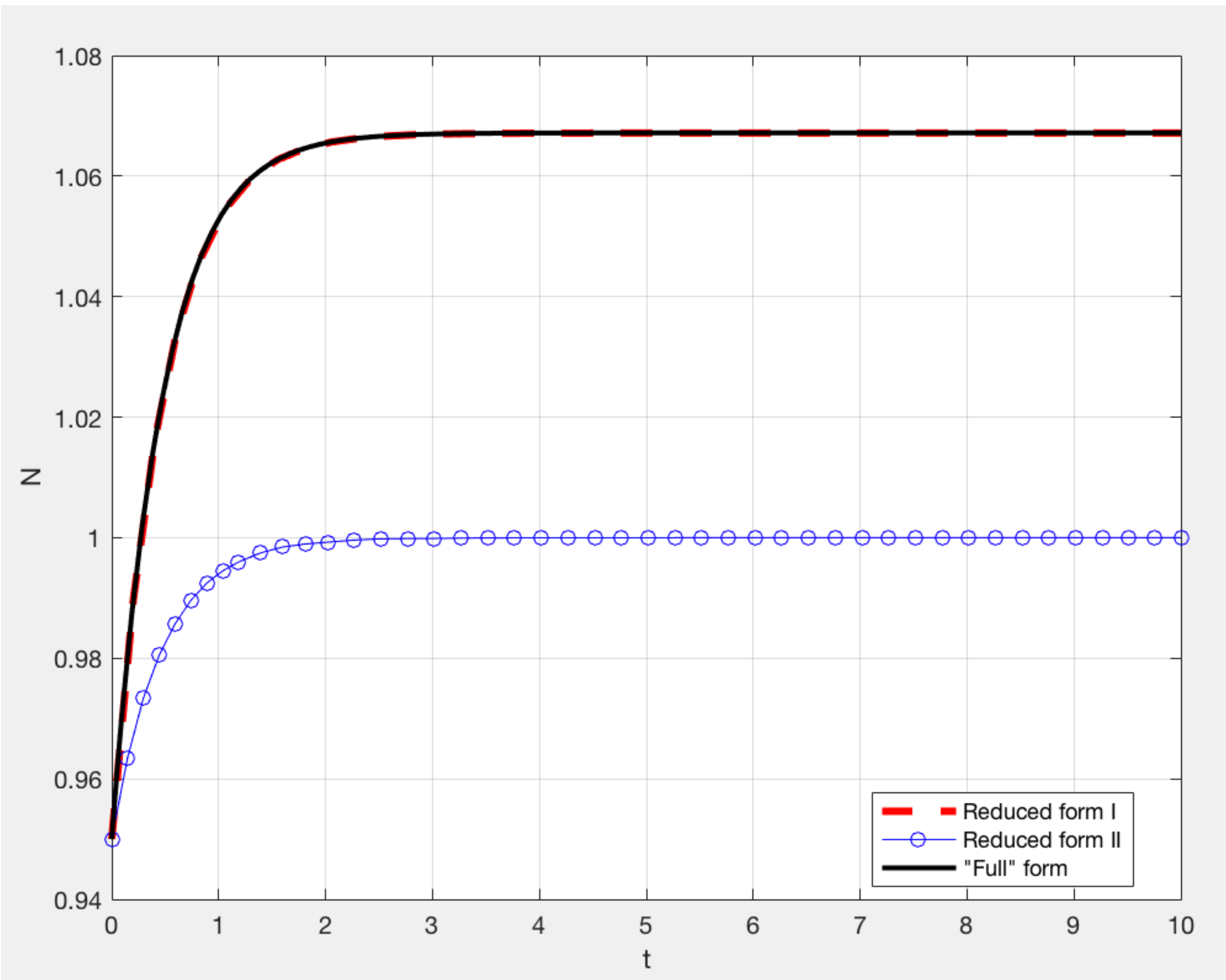
Figure 3: Comparison of numerical solutions for the differential equation $y' = y^2 - 4$ with solution estimated numerically using Euler's method with a step-size of $\Delta x = 0.01$ for different initial y values at $x = 0$. Though not shown on this figure, initial conditions where $y_o > 2$ will diverge towards $+\infty$ (as $x \to \infty$) and $y_o < 2$ will converge towards $y = 2$.

Answer: The effect of changing c is two-fold. First, it changes the equilibrium values and thus what the asymptotic limits are. Second, because $\sqrt{c}$ appears in the argument of the exponent, increasing c will increase the rate at which solution asymptotically approach (or move away from) the equilibria. For example, if $|y_o| < \sqrt{c}$, then a larger value of c means a sharper transition in the S-shape that the solution takes. One could also introduce the variables $u = y/\sqrt{c}$ and $\tau = \sqrt{c}x$. This change of variables would reformulate the differential equation as

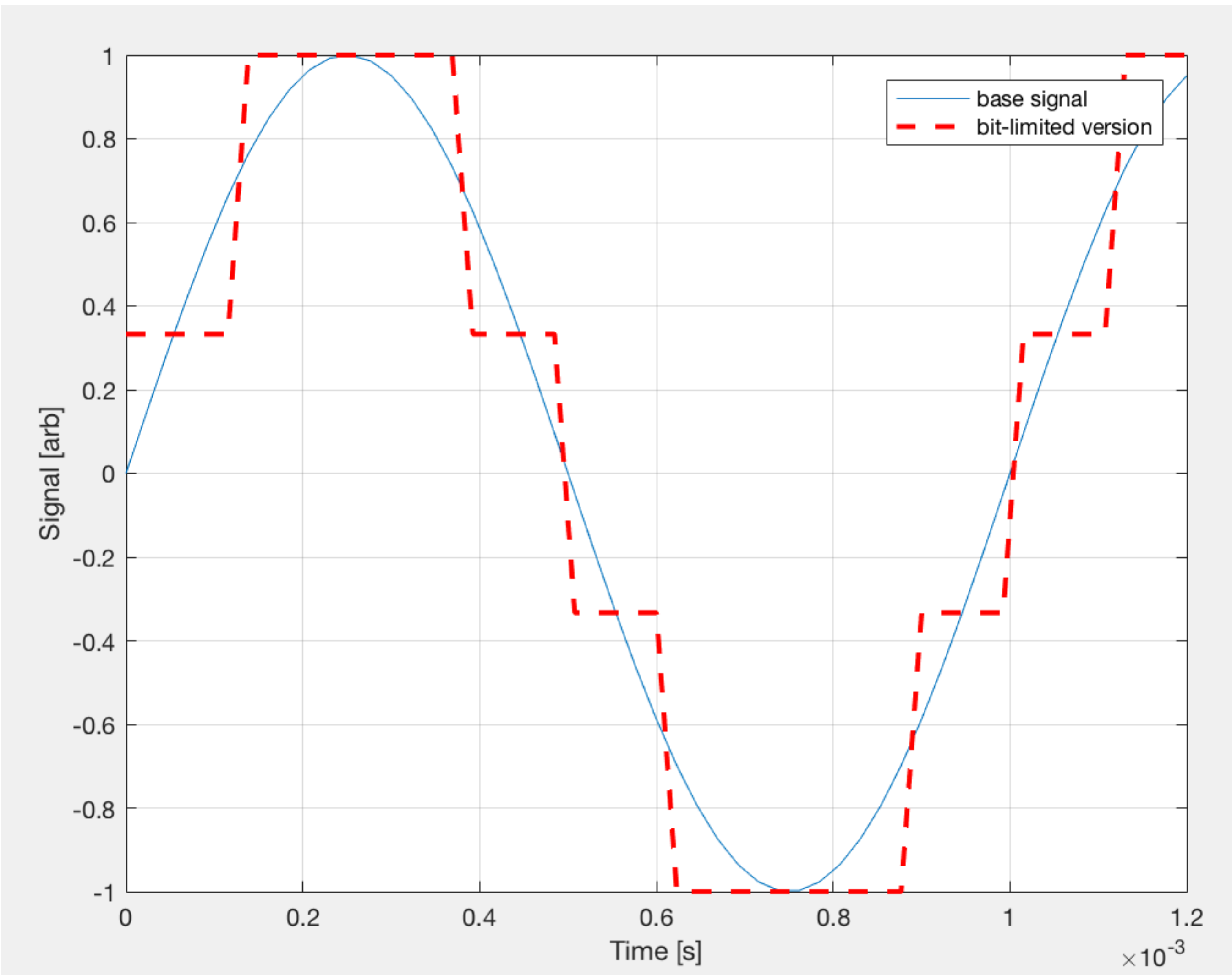
$$\frac{du}{d\tau} = u^2 - 1$$

and thereby removing the parameter-dependence in terms of understanding the underlying dynamics.

```
% ### EXgompertz.m ###      2018.01.19  C. Bergevin

% [REF: pg.157 of Edelstein-Keshet]
% Purpose: Solve/plot the Gompertz equation for tumor growth in three
different ways:
% form A: system of two coupled (nonlinear, autonomous) ODEs
%            out1(1)= y(2)* y(1);
%            out1(2)= -P.alpha*y(2);
% form B: reduced version down to single (non-autonomous) ODE
%            out1(1)= P.gamma0*y(1)*exp(-P.alpha*t);
% form C: further reduction to a single (autonomous) ODE
%            out1(1)= -P.alpha* y(1)* log(y(1));

% - forms A & B (here on out called form A/B) are indeed equivalent; however
from C is not
% - limiting equilibrium value (for form A/B) is a mix of P.y0(1), P.y0(2), &
P.alpha
% - depending upon params, form A/B can appear simply asymptotic or sigmoidal
% - to ensure the "form C" heads in the right direction, need to have P.y0(1)
% (i.e., initial val. of N) less than unity (i.e., expressed as a fractional
amount)
% --> I am missing a piece of the puzzle there is an error in Edelstein-
Keshet's logic

% ------------------------------------------
P.alpha= 2.15;      % alpha {1?}
P.y0(1) = 0.95;    % initial N {10?}
P.y0(2) = 0.25;    % initial gamma {0.1?}
% Integration limits
P.t0 = 0.0;    % Start value
P.tf = 10.0;    % Finish value
% ------------------------------------------

P.gamma0= P.y0(2);  % separate out intial tumor growth rate
% +++
% --- form A ("full" form I)
[tF1 yF1] = ode45('GompertzFuncF1', [P.t0 P.tf],P.y0,[],P);
% --- form B (reduced form I)
[tR1 yR1] = ode45('GompertzFuncR1', [P.t0 P.tf],P.y0(1),[],P);
% --- form C (reduced form II)
[tR2 yR2] = ode45('GompertzFuncR2', [P.t0 P.tf],P.y0(1),[],P);
% +++
%yR1norm= yR1/yR1(end);  % "normalized" version of form B (re final val.)
% +++
% visualize
figure(1); clf;
h1= plot(tR1,yR1,'r--','LineWidth',3); hold on; grid on;
h2= plot(tR2,yR2,'bo-');
h3= plot(tF1,yF1(:,1),'k','LineWidth',2);
%h4= plot(tR1,yR1norm,'r-.','LineWidth',1);

xlabel('t');    ylabel('N');
legend([h1 h2 h3],'Reduced form I','Reduced form II','"Full"
form','Location','best');
```


1.08
1.06
1.04
1.02
1
0.98
0.96
0.94
N
0
1
2
3
4
5
6
7
8
9
10
t
Reduced form I
Reduced form II
"Full" form

```
% ### EXbitDepth.m ###      2018.02.01 C. Bergevin

% Goal: Simulate changing (e.g., lowering) the "bit depth" for a signal

% o base signal here is a sinusoid w/ amplitude In.A
% o very kludge way to "digitize" --> develop more efficient means

clear
% ============================================================
In.A= 1;  % amplitude for sinusoid {5.5}
In.f= 1000;    % sinusoid frequency [Hz] {1000}
In.N= 2;     % # of bits for digitization (must be int. >= 1) {2?}
In.tL= [0 0.0012];   % time limits
In.SR= 44100;
% ============================================================

% --- create base values
t= linspace(In.tL(1),In.tL(2),round((In.tL(2)-In.tL(1))*In.SR));  % time
vals.
sig= In.A*sin(2*pi*In.f*t); % non-"digitized" signal
% --- create a "bit-limited" digitized vertical scale
spacing= (2*In.A)/(2^In.N-1);  % determines spacing between quantized vals.
digiS= linspace(-In.A,In.A,2^(In.N));  % easy way to get quantized vals.
% NOTE: should have that spacing= any val. of diff(digiS)
% --- (very) kludge way to force sig vals. to take on "bit-limited" vals.
for nn=1:numel(sig)
    indx= max(find(sig(nn)>=digiS));
    if ((sig(nn)-digiS(indx))<=(digiS(indx+1)-sig(nn))), sigQ(nn)=
digiS(indx);
    elseif ((sig(nn)-digiS(indx))>(digiS(indx+1)-sig(nn))), sigQ(nn)=
digiS(indx+1);  end
end
% ---
figure(1); clf;
h1= plot(t,sig); hold on; grid on;
h2= plot(t,sigQ,'r--','LineWidth',2);
xlabel('Time [s]'); ylabel('Signal [arb]');
legend([h1 h2],'base signal','bit-limited version')
```



base signal
bit-limited version
Signal [arb]
1
0.8
0.6
0.4
0.2
0
-0.2
-0.4
-0.6
-0.8
-1
0
0.2
0.4
0.6
0.8
1
1.2
Time [s]
×10^-3