Dr. Leon VanDommelen, Exam 1, 10/04/18, Question 1

Table of Contents

IMPORTANT:

1) Do not change **anything** in this header! Put the solution to the question completely at the end of the file.

2) Since the solution requires a custom function Kepler, put a

```
%% Additional m-File: Kepler.m
%
% <include>Kepler.m</include>
```

block behind your main solution. (Copy the above three lines and paste them completely at the end of this file, behind a blank line. For each line, get rid of the first percent and the two spaces behind it.)

Initialize

```
format compact
more off
```

(6) know must form error
(8) write Kepler.m
(2) plot curves
(2) grid
(2) legend
(5) find intervals
(5) fzero call
(5) anonymous function

My Solution:

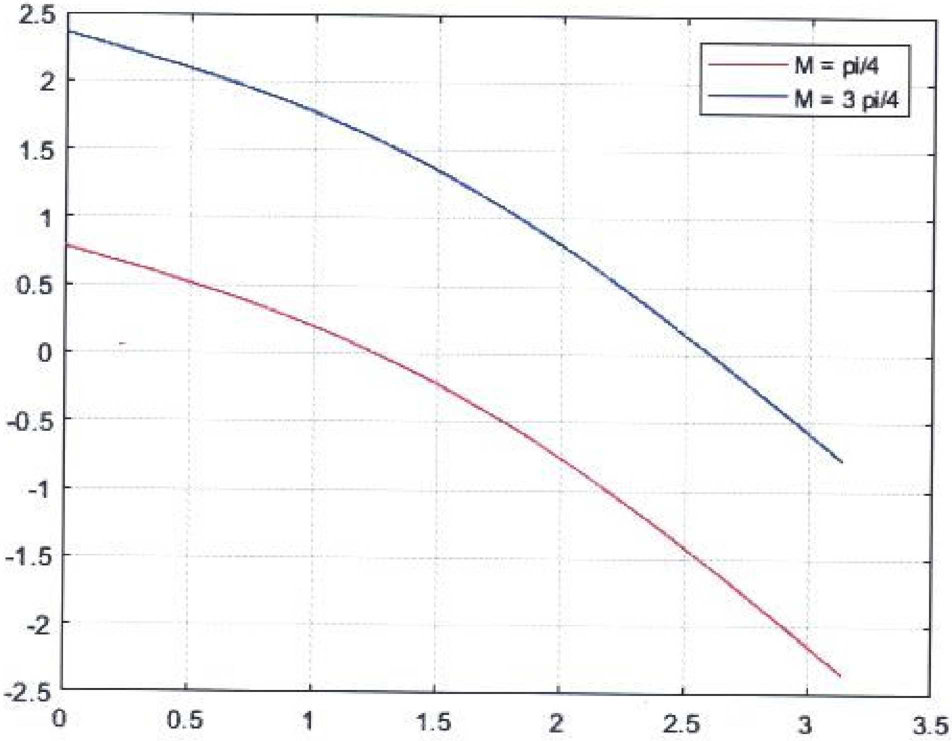
```
% data for the two cases
e=0.5;
M1=pi/4;
M2=3*pi/4;

% plot curves
EPlot=linspace(0,pi,100);
error1Plot=Kepler(EPlot,M1,e);
error2Plot=Kepler(EPlot,M2,e);
plot(EPlot,error1Plot,'r',...
     EPlot,error2Plot,'b')
grid on
legend('M = pi/4','M = 3 pi/4')

% find values of E
M=M1;
E1=1;
E2=1.5;
E=fzero(@(E) Kepler(E,M,e),[E1 E2]);
```

```
fprintf('For e = %.3f and M = %.5f, E = %.5f (interval [%.3f %.3f]).
\n',...
        e,M,E,E1,E2)
M=M2;
E1=2.5;
E2=3;
E=fzero(@(E) Kepler(E,M,e),[E1 E2]);
fprintf('For e = %.3f and M = %.5f, E = %.5f (interval [%.3f %.3f]).
\n',...
        e,M,E,E1,E2)
```

```
For e = 0.500 and M = 0.78540, E = 1.26170 (interval [1.000 1.500]).
For e = 0.500 and M = 2.35619, E = 2.60975 (interval [2.500 3.000]).
```



Additional m-File: Kepler.m

```
function error = Kepler(E,M,e)

% evaluate the error in the equation
error=M-E+e*sin(E);

end
```

Dr. Leon VanDommelen, Exam 1, 10/04/18, Question 2

Table of Contents

IMPORTANT:

Do not change **anything** in this header! Put the solution to the question completely at the end of the file.

Initialize

xPlot
5 create data xVals
5 polyfit
5 polyval (harder)
15

5 maxerror
2 fprintf
3 polyder
10

4 plot
2 symbols, lines, colours
2 axes
1 grid
9

```
format compact
more off
```

My Solution:

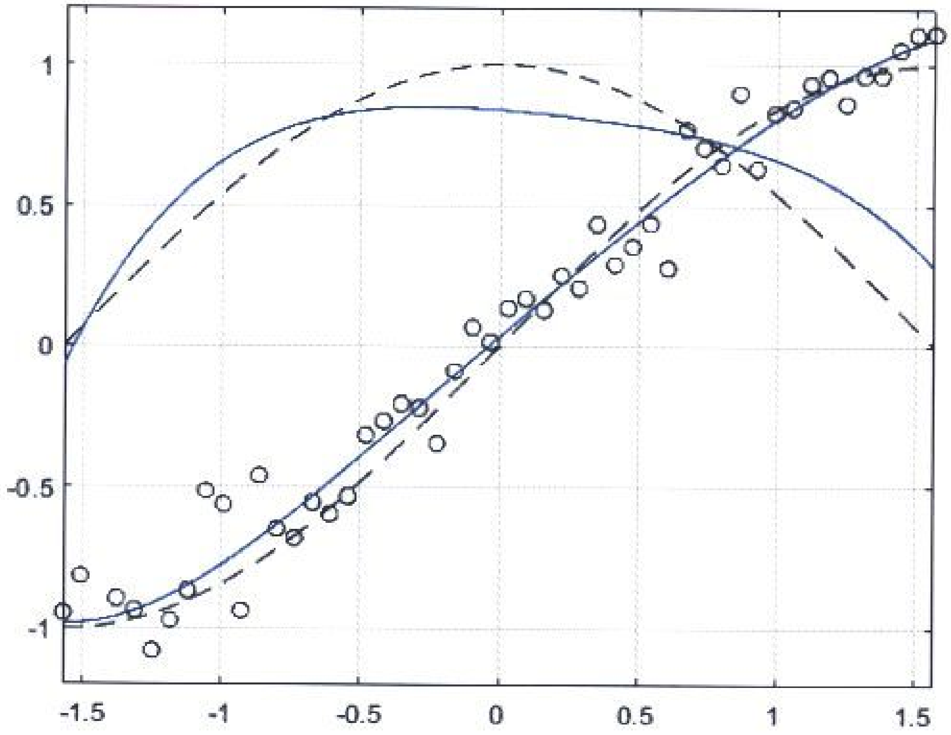
```
% create noisy data
xVals=linspace(-pi/2,pi/2,50);
rng('default')
fVals=sin(xVals)+0.1*randn(size(xVals));

% create fit
CoefQuintic=polyfit(xVals,fVals,5);

% maximum error at the plot points
xPlot=linspace(-pi/2,pi/2,100);
fQuinticPlot=polyval(CoefQuintic,xPlot);
errQuintic=max(abs(fQuinticPlot-sin(xPlot)));
fprintf('The quintic fit has a maximum error %.2E\n',errQuintic)

% plot
fExactPlot=sin(xPlot);
derfExactPlot=cos(xPlot);
derCoefQuintic=polyder(CoefQuintic);
derfQuinticPlot=polyval(derCoefQuintic,xPlot);
plot(xPlot,fExactPlot,'--k',...
     xVals,fVals,'ok',...
     xPlot,derfExactPlot,'--k',...
     xPlot,fQuinticPlot,'b',...
     xPlot,derfQuinticPlot,'b')
grid on
axis([-pi/2 pi/2 -1.2 1.2])
```

```
The quintic fit has a maximum error 9.50E-02
```



Published with MATLAB® R2017b

Dr. Leon VanDommelen, Exam 1, 10/04/18, Question 3

Table of Contents

IMPORTANT:

1) Do not change **anything** in this header! Put the solution to the question completely at the end of the file.

2) Since the solution requires a custom function ODE, put a

```
%% Additional m-File: ODE.m
%
% <include>ODE.m</include>
```

block behind your main solution. (Copy the above three lines and paste them completely at the end of this file, behind a blank line. For each line, get rid of the first percent and the two spaces behind it.)

Initialize

```
format compact
more off
```

(9) create function ODE

(4) ode45 call, generic

(4) put in initial conditions

(4) put in tRange

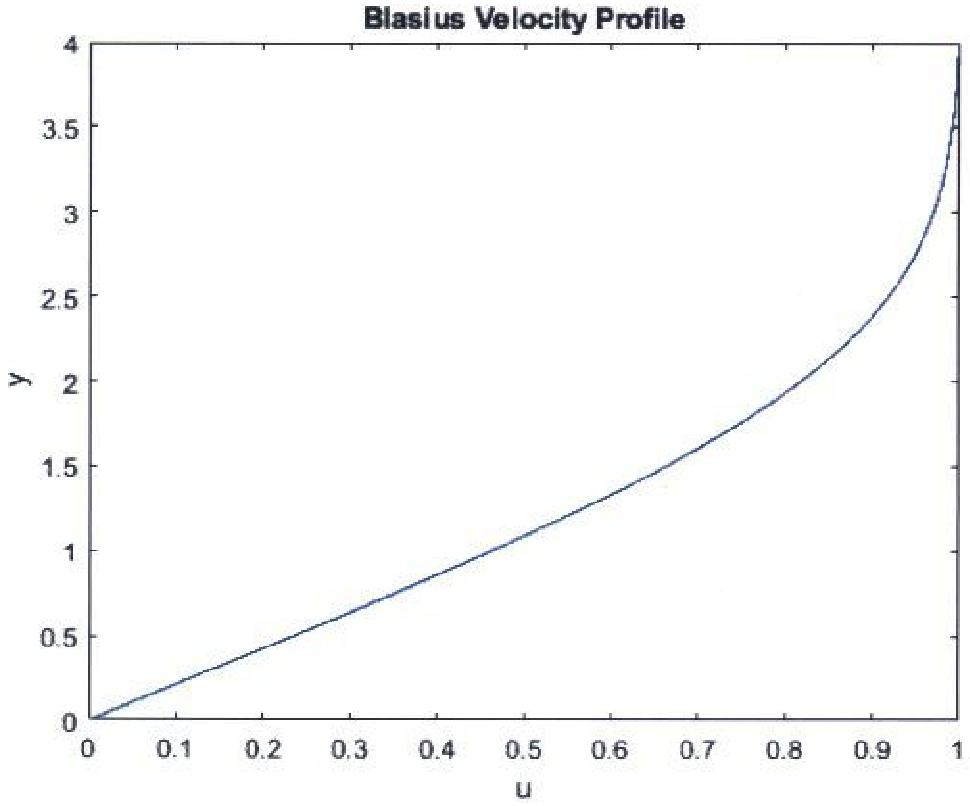
My Solution:

```
[yValues unknownsValues] = ...
    ode45('ODE',linspace(0,4,100),[0 0 .47]');
uValues=unknownsValues(:,2);
plot(uValues,yValues,'b')
title('Blasius Velocity Profile')
xlabel('u')
ylabel('y')
```

(5) take u out of unknowns values

(4) plot axes as requested

(3) title, xlabel, ylabel



Additional m-File: ODE.m

```
function unknownsDerivatives = ODE(y,unknowns)

% take the unknowns out of the vector for readability
p=unknowns(1);
u=unknowns(2);
o=unknowns(3);

% derivatives of the unknowns
dpdy=u;
dudy=o;
dody=-p*o;

% return the derivatives as a *column* vector
unknownsDerivatives=[dpdy dudy dody]';

end
```

Published with MATLAB® R2017b