Nanyang Technological University, Singapore
MATLAB activities in MH1801

[This comprehensive handout was entirely created by Dr. Fedor Duzhin for MH1801 in 2014. The grading system was adapted slightly by Conan Wong for MH1801 in January 2015.]

Full handout for all lab sessions

General instructions.

There are 12 MATLAB activities. In each activity, you are supposed to read a brief description and complete the task using the MATLAB code provided.

For 9 activities, you don't need to submit the task. It doesn't mean that you can skip the activity. Later you may need to do something related to this task within another activity.

3 activities (#4, #8 and #12) are to be submitted and graded by your lab instructor. They will make up your lab grade (out of 10), which is 10% of your overall grade for the MH1801 course. Of these 3 activities, I believe #8 is the most challenging.

If you prefer to work on your own laptop, then you can either run MATLAB on the SPMS server or you can install GNU Octave, which is a very similar programming language and most of MATLAB code should work under GNU Octave without any change. The user interface of GNU Octave is probably a bit less convenient and it might not be straightforward to install it (unless you work in Linux - then it's very easy), but these problems are solvable. Relevant links:

Run MATLAB on the SPMS server Install GNU Octave

You can even use any other software that plots vector fields and differential equations as long as it is able to produce illustrations for your reports, but then, of course, don't rely on lab instructors too much as they will not be familiar with your software. Example of such a software is Maxima.

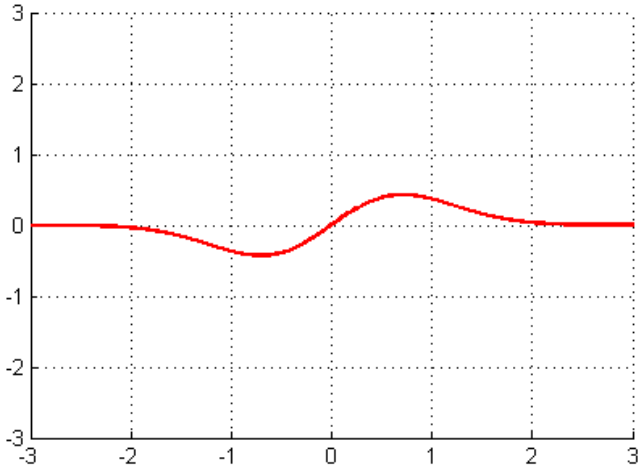
Activity 1: MATLAB command “plot”.

Objective: recall the command “plot” that we used to draw graphs in Calculus I.

The following piece of code plots the graph $y = xe^{-x^2}$ for $-3 \le x, y \le 3$.

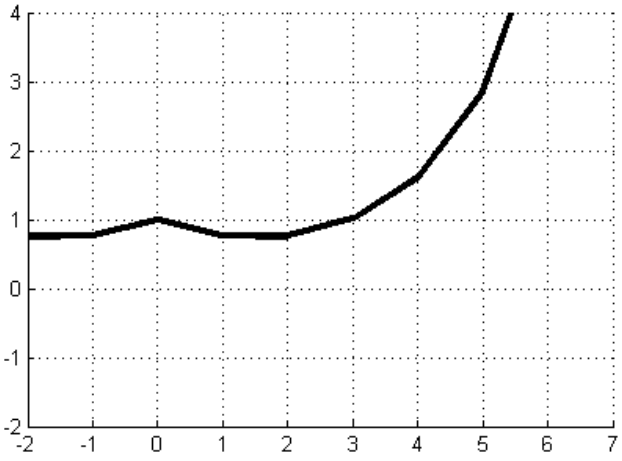
```
% Activity 1: plot
% Here we plot the graph of the function x*e^(-x^2)
% Recall that ; prevents output (try to remove ; and run the code to see
% what happens without it)

%The following code initializes the plotting window.
%You don't have to understand what these commands do.
%If you want, read MATLAB manual by selecting a command and pressing F1
%or do a google search like 'clc command MATLAB'
clc
clf
hold on
grid on

%We are going to plot the graph for a<=x<=b and c<=y<=d
%Besides, we'll use 1001 approximation points for plotting
%The parameters a, b, c, d, n can be modified
a=-3;
b=3;
c=-3;
d=3;
n=1000;

%The following code prepares the data for plotting.
%It creates two arrays for x and y values.
x=a:(b-a)/n:b;
y=exp(-x.^2).*x;
%Notice that the exponential function is denoted exp and that operations of
%square and multiplication are preceded by a dot. The reason is without
%this dot, MATLAB will try to use matrix multiplication instead of
%component-wise multiplication, which will cause an error.

%The following code plots the graph in red colour and line of width 2
%pixels.
plot(x,y,'color','red','linewidth',2)

%Finally, the following command changes the size of the plane that is
%actually seen on the plot to the rectangle a<=x<=b and c<=y<=d
axis([a b c d])
```

Create a new MATLAB script. Copy the code above into your script and save it. You'll need to give it a name. MATLAB has certain restrictions on file names - for instance, you cannot begin the name with a digit or have a space in it. Run the code by clicking F5. You are supposed to get the following figure:



Your task:

Modify the code so it plots the function $y = \frac{\cosh(x)}{x^2+1}$ for $-2 \le x \le 7$ and $-2 \le y \le 4$ using only 10 approximation points, line of width 3, and black colour. You should get the following picture:



There is nothing to submit in Activity 1. The goal is to understand the importance of a large number of approximation points for fine plotting.

Activity 2: Advanced plot.

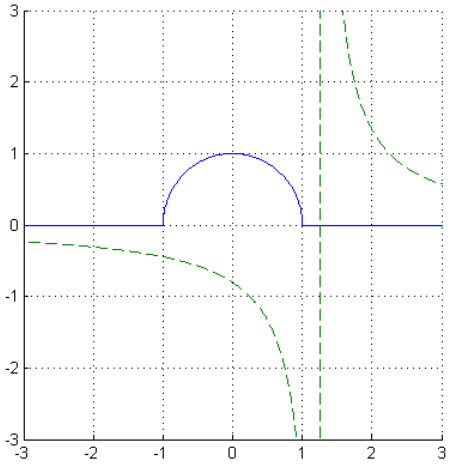
Objective: understand that you shouldn't apply "plot" blindly. Try to analyse your data first.

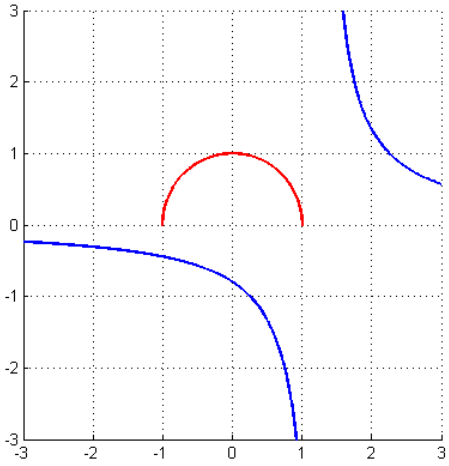
We already know the MATLAB command "plot". Applying it could be sometimes tricky since it doesn't account for the domain of the functions that are being plotted. The following example produces two plots: an incorrect and a correct version of graphs of functions $y=\sqrt{1-x^2}$ and $z=\frac{1}{x-1.25}$ for $-3\le x,y\le 3$. The problem is that the function $y(x)$ is only defined for $-1\le x\le 1$. In other words, its domain is the interval $[-1,1]$. The function $z(x)$ is only defined for $x\neq -1.25$. We need to take care of the domains somehow.

```
% Activity 2: more advanced plotting
% Here we plot the graphs of two functions. In both cases, the domain is
% not the whole set of real numbers. There will be two plots: good and bad.
% Recall that ; prevents output (try to remove ; and run the code to see
% what happens without it)

%We are going to plot the graph for a<=x<=b and c<=y<=d
%Besides, we'll use 1001 approximation points for plotting
%The parameters a, b, c, d, n can be modified
a=-3;
b=3;
c=-3;
d=3;
n=1000;

%The following code initializes two plotting windows and takes you to
%the first window. The point is to create a bad plot in the first window
%and a good plot in the second window.
clc
clf
subplot(1,2,1)
hold on
grid on

%The following code prepares the data for bad plotting.
%It creates three arrays:
%array for independent variable x, array for y(x), and array for z(x)
```

```
x=a:(b-a)/n:b;
y=sqrt(1-x.^2);
z=1./(x-1.25);

%The following code plots the graph in default colours and lines of default
%width 1 and specified style: '-' for a solid and '--' for dashed
plot(x,y,'-',x,z,'--')
%Notice that both graphs are wrong: the graph for y(x) doesn't account for
%the fact that x can only be from the interval [-1,1] and the graph for
%z(x) has a strange vertical piece of line.

%Here we set the visible window to the rectangle a<=x<=b and c<=y<=d
axis([a b c d])

%Now we initialize the second plotting window. Here we need the 'hold on'
%command because we will be adding things to the plot
subplot(1,2,2)
hold on
grid on

%The following code prepares the data for good plotting. Notice that the
%domain of the function y(x) is the interval [-1,1] and the domain of the
%function z(x) is everything except for 1.25. We'll create two different
%arrays for the independent variable x1 and x2, for the two functions.
%Notice that x1 only runs between -1 and 1 while x2 explicitly skips the
%value 1.25 and inserts NaN into the array at the place where 1.25 should
%have been.
x1=-1:2/n:1;
y=sqrt(1-x1.^2);
x2=[a:(b-a)/n:1.25-(b-a)/n NaN 1.25+(b-a)/n:(b-a)/n:b];
z=1./(x2-1.25);

%The following code plots the correct graphs:
plot(x1,y,'color','red','linewidth',2)
plot(x2,z,'color','blue','linewidth',2)

%Here we set the visible window to the rectangle a<=x<=b and c<=y<=d
axis([a b c d])
```

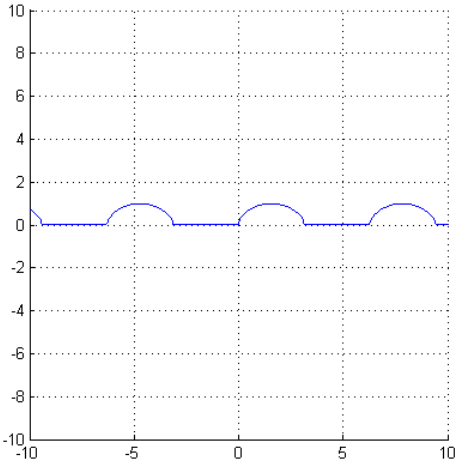
Create a new MATLAB script. Copy the code above into your script, save it and run by clicking F5. You are supposed to get the following figure:

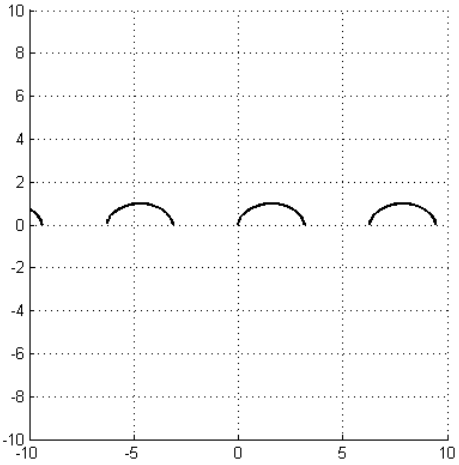




Your task:

Modify the code so it plots the function $y = \sqrt{\sin x}$ for $-10 \le x \le 10$ and $-10 \le y \le 10$. Again, it's supposed to produce two versions of the plot - the wrong 'straightforward' one and the more tricky one that accounts for the domain of the function. You should get the following picture:





There is nothing to submit in Activity 2. The goal is to understand the importance of analysing your functions before plotting.

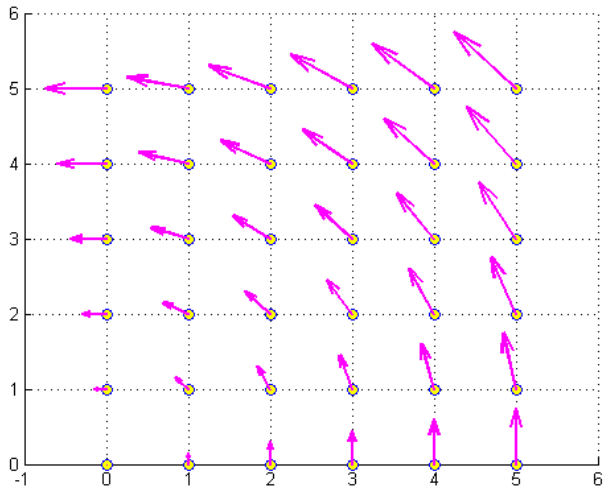
Activity 3: Commands “meshgrid” and “quiver” to plot vector fields.

Objective: learn the commands “meshgrid” and “quiver” to plot vector fields.

The native MATLAB command for plotting slope fields draws little arrows rather than just small intervals. It is, in fact, more convenient because the direction of the arrow shows how the solution of the differential equation is going to change as the independent variable grows. The command “quiver” plots a vector field over a grid - at each point (x,y) of the grid it draws a little arrow with coordinates (A,B), where A and B are some explicit expressions in x and y. The following code plots the vector $(-y,x)$ for all points (x,y) with $x=0,1,2,3,4,5$ and $y=0,1,2,3,4,5$. Thus you’ll see a 6×6 grid of little arrows.

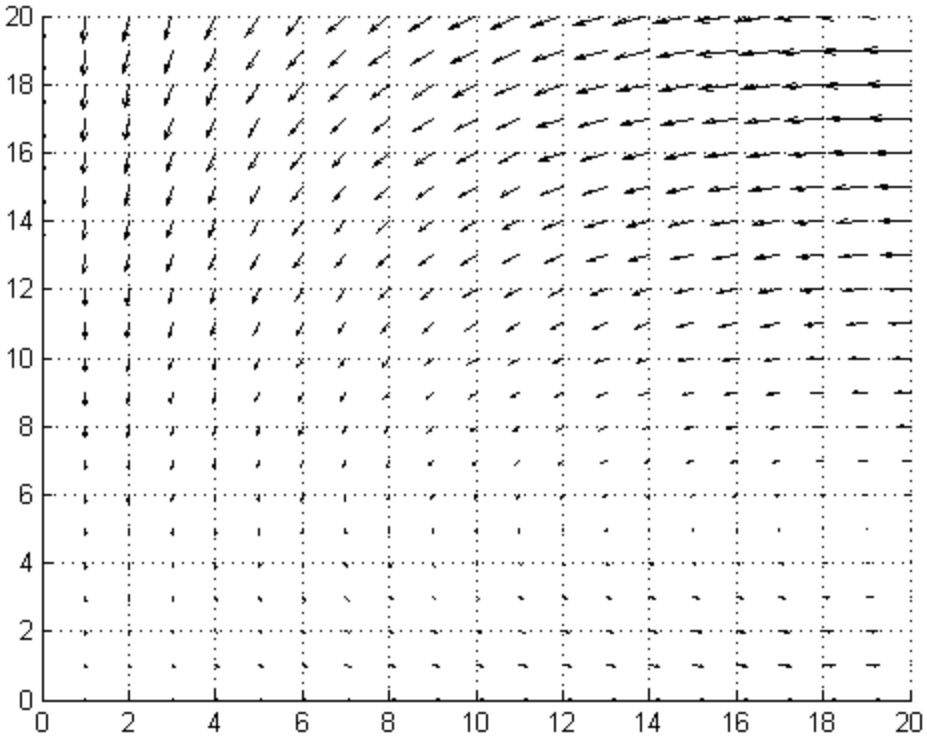
```
% Activity 3: plotting vector fields.
clc
clf
hold on
grid on

%The following command prepares the grid for the plotting.
%Here, both x and y take integer values between 0 and 5.
%Since there is no semicolon, you'll see the values of x and y in the main
%MATLAB window
%Besides, the grid is explicitly plotted as a collection of little yellow
%circles with blue outline
[x y]=meshgrid(0:5,0:5)
plot(x,y,'o','color','blue','markerfacecolor','yellow')

%The following code introduces A and B as explicit expressions in x and y
A=@(x,y)-y;
B=@(x,y)x;

%Now we'll plot the vector field (-y,x) in magenta color and line of width 2
quiver(x,y,A(x,y),B(x,y),'linewidth',2,'color','magenta')
```

Create a new MATLAB script. Copy the code above into your script, save it and run by clicking F5. You are supposed to get the following figure:



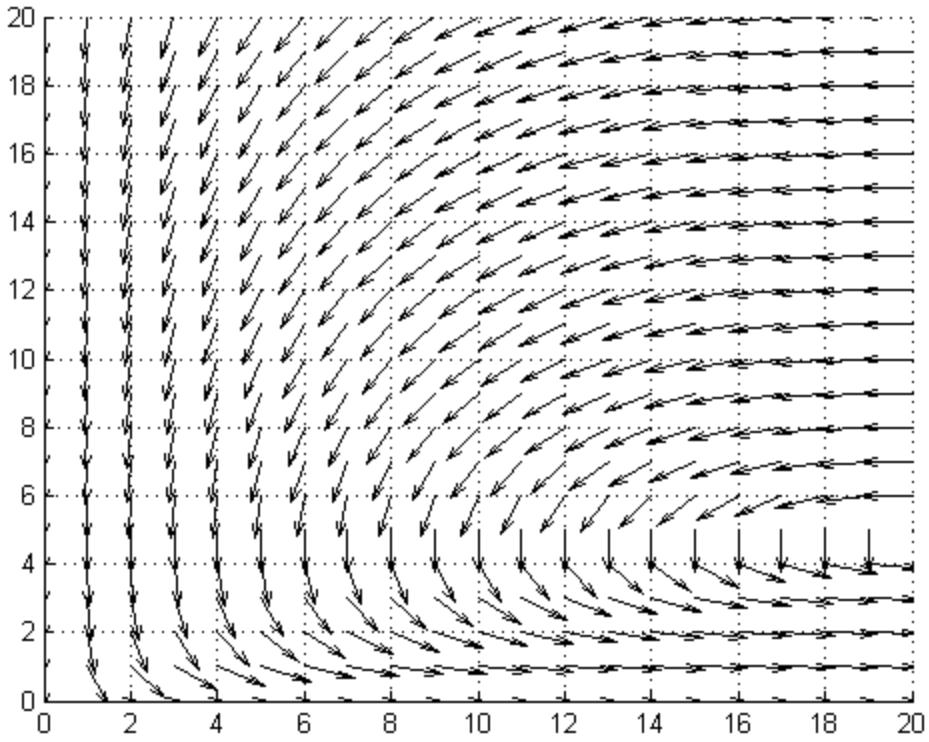
Your task:

Modify the code so it plots the vector field given by $A(x,y) = 0.1x - 0.02xy$, $B(x,y) = -0.2y + 0.01xy$ for $0 \le x,y \le 20$. Make it draw black arrows of line width 1. Besides, make sure that the only part of the plane shown on your plot is the square $0 \le x,y \le 20$. You should get the following picture:



Challenge:

In the plot above some arrows are long and some are short. It's really hard to see the direction for those short ones. Modify the again so it plots arrows that point in the same direction as in the plot above but all have the same length. You should get the following picture:



There is nothing to submit in Activity 3.

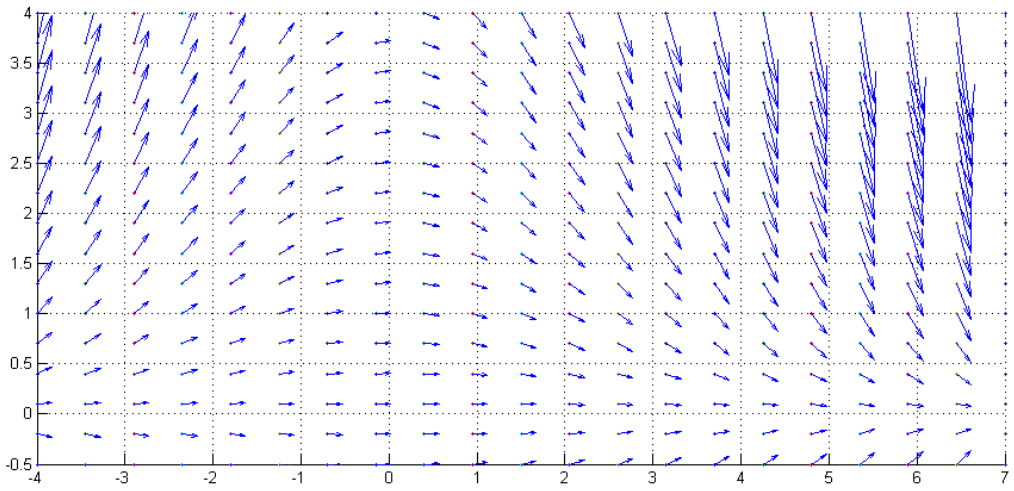
[GRADED] Activity 4: Plot direction fields for a first order equation.

Objective: apply “meshgrid” and “quiver” to plot vector fields; solve and submit the little online assignment.

Consider a first order differential equation $\frac{dy}{dx} = f(x,y)$. Its slope field is a diagram that shows a collection of little intervals of slope $f(x,y)$ for each point (x,y) over some grid. Instead of an interval, we can draw a vector of the same slope. Thus the vector should be $(1, f(x,y))$. Thus essentially the slope field is the same as the vector field given by $A(x,y) = 1$ and $B(x,y) = f(x,y)$. The following code plots the direction field of the equation $\frac{dy}{dx} = -\frac{xy}{4}$:

```
% Activity 4: plotting direction fields.
% The direction field for a differential equation dy/dx=f(x,y) is
% the same as the vector field given by A=1, B=f(x,y)
clc
clf
hold on
grid on

%We'll plot our vector field for a<=x<=b and x<=y<=d
%over the grid of dimensions (m+1) by (n+1).
%First, we initialize parameters (so it's easier to change them in future)
%and then use the meshgrid command. We'll plot the grid too.
a=-4; b=7; c=-0.5; d=4; m=20; n=15;
[x y]=meshgrid(a:(b-a)/m:b,c:(d-c)/n:d);
plot(x,y,'o','markersize',2)

%The following code introduces A and B as explicit expressions in x and y
f=@(x,y)-x.*y./4;%it's important to use .* and ./ instead of just * and /
A=@(x,y)x-x+1;   %adding x-x is needed to sync the dimensions of A and x
B=@(x,y)f(x,y);

%Now we'll plot the slope field
%Notice the additional argument 2 of the quiver function - it makes each
%little arrow twice as long as it is supposed to be otherwise. Without it,
%some arrows will be too small (of course, now some of them are too big)
quiver(x,y,A(x,y),B(x,y),2,'linewidth',1,'color','blue','linewidth',1)

axis([a b c d])
```

Create a new MATLAB script. Copy the code above into your script, save it and run by clicking F5. You are supposed to get the following figure:



Your task:

Modify the code to produce plots of the following differential equations. Make the plot for $-3 \le x, y \le 3$ and set the distance between grid points to 0.25. In other words, values for both x and y should be $-3, -2.75, -2.5, \cdots, 2.75, 3$.

A. $\frac{dy}{dx} = \sin(x - y^2)$
B. $\frac{dy}{dx} = \cos(x + y^2)$
C. $\frac{dy}{dx} = \sin(x + y) \cdot \cos y$
D. $\frac{dy}{dx} = -\sin\frac{x}{2} \cdot \cos y$

[Graded Activity for Submission]

Submit your 4 plots to your Lab Instructor in softcopy.

You should include your name, your matric number and the 4 plots prepared in MATLAB. Convert your file to **PDF** format and submit it to your lab instructor.

Please **don't** submit text files, word documents, photographed reports etc. If you wish, you can use MS Word, Open Office, Google Drive or LaTeX to collate the diagrams and save it as a **PDF**.

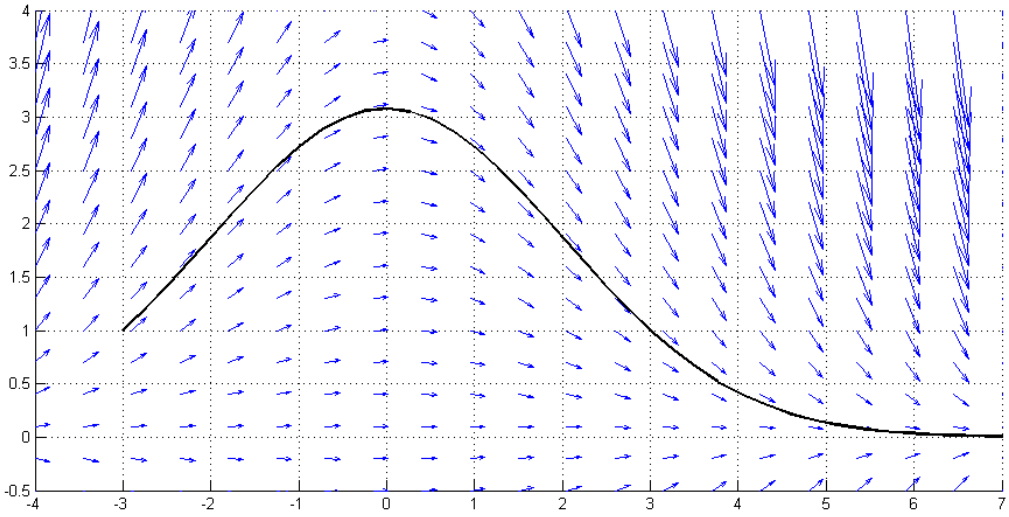
Activity 5: Solve differential equations numerically.

Objective: learn the command “ode45” to solve differential equations numerically.

Euler’s method is used to solve the initial value problem $\frac{dy}{dx} = f(x,y),\ y(x_0) = y_0$ numerically. Many more sophisticated variations of Euler’s method are now known to the mankind (if you’re interested, go here). You don’t have to know them unless you’re an expert in numerical methods. All you need to know is that MATLAB has a number of commands that will solve differential equations for you. Different commands correspond to different methods and the most basic and usual one is “ode45”.

```
% Activity 5: plotting solutions on top of direction fields.
% We'll re-produce the illustration from slide 34 of Lecture 4:
% direction fields of the DE dy/dx=-xy/4 and the initial condition y(-3)=1.

% First, plot the direction field with the code from Activity 4
clc
clf
hold on
grid on

a=-4; b=7; c=-0.5; d=4; m=20; n=15;
[x y]=meshgrid(a:(b-a)/m:b,c:(d-c)/n:d);

f=@(x,y)-x.*y./4;
A=@(x,y)x-x+1;
B=@(x,y)f(x,y);
quiver(x,y,A(x,y),B(x,y),2,'linewidth',1,'color','blue','linewidth',1)
% The direction field has been plotted.

%Setting the initial condition:
x0=-3; y0=1;

%The following code solve the DE dy/dx=f(x,y), where f(x,y) is defined
%above. Specifically, it produces the table of values for x and y, where x
%is in the interval from x0 (initial condition) to b (end of the interval)
%Notice the syntax of the command "ode45"
[x,y] = ode45(f,[x0 b],y0)

%The following command just plots the graph of the solution from its table
%of values, which was written into x, y by the previous command
plot(x,y,'linewidth',2,'color','black')

axis([a b c d])
```

Create a new MATLAB script. Copy the code above into your script, save it and run by clicking F5. You are supposed to get the following figure:



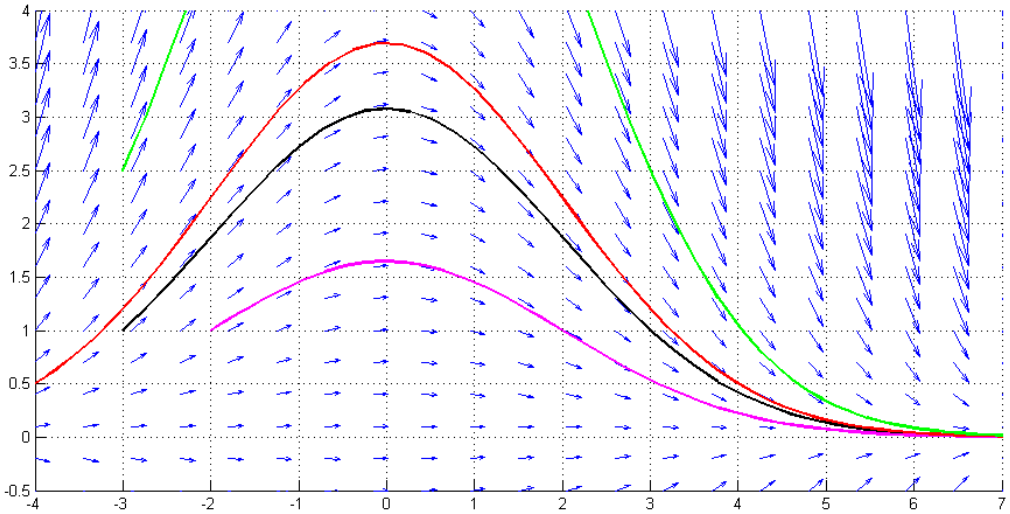
Your task:

Modify the code to add graphs of four other solutions of the same equation $\frac{dy}{dx} = f(x,y)$ satisfying the following initial conditions:

$y(-3) = 1$ - in black $\qquad$ $y(-2) = 1$ - in magenta

$y(-4) = 0.5$ - in red $\qquad$ $y(-3) = 2.5$ - in green

You are supposed to get the following picture:



There is nothing to submit.

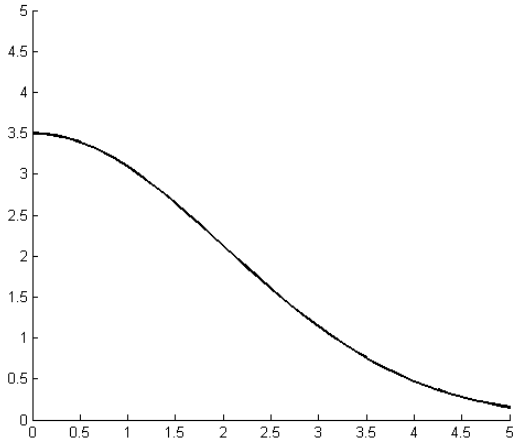
Activity 6: Solve differential equations, part 2.

Objective: apply the command “ode45” to solve and submit the little online assignment.

Of course, we don’t have to plot the direction field every time we need to solve a differential equation. The following code just graphs a particular solution of the equation $\frac{dy}{dx} = -\frac{xy}{4}$. Notice that you can read the initial condition from the graph but you cannot really guess the differential equation from the graph.

```
% Activity 6: solving first order differential equations. Such an equation
% is dy/dx=f(x,y) and the initial condition y(0)=y0.
% For simplicity, we don't plot the grid or the direction field.
% Here, the initial value problem is
% dy/dy=-xy/4, y(0)=3.5.

clc
clf
hold on
grid off

a=0; b=5; c=0; d=5;
x0=0; y0=3.5;

f=@(x,y)-x.*y./4;
[x,y] = ode45(f,[x0 b],y0)
plot(x,y,'linewidth',2,'color','black')

axis([a b c d])
```

It produces the following figure:



Task

Modify the code to plot solutions of the following differential equations. Submit your answer to edveNTUre by matching equations with graphs of their solutions. You'll need to get the initial conditions from the graphs.

A. $\frac{dy}{dx} = x - 0.9y - 1$
B. $\frac{dy}{dx} = x(0.2x - y)$
C. $\frac{dy}{dx} = e^{x-y}$
D. $\frac{dy}{dx} = -\frac{y(y-4)}{x+1}$

There is nothing to submit for this task.

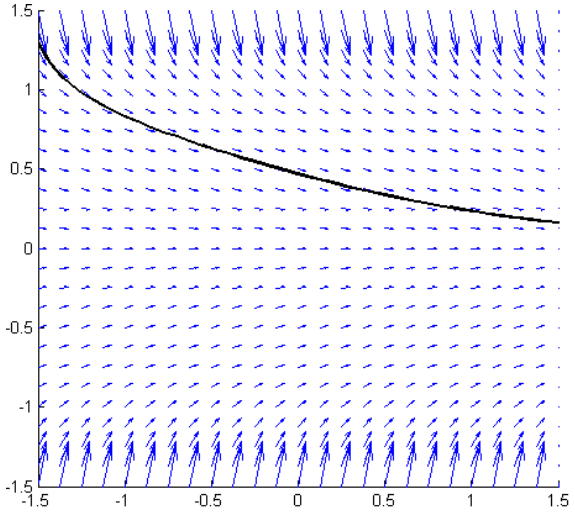
Activity 7: Analyse a differential equation using computer.

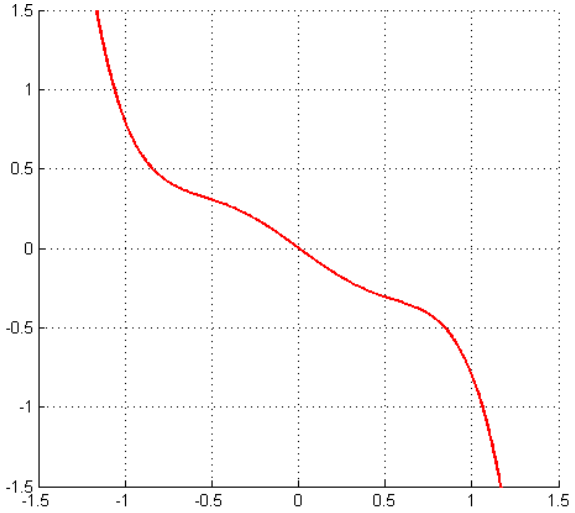
Objective: apply commands “plot”, “meshgrid”, “quiver”, and “ode45” together with your pen, paper, and powerful brain to analyse a differential equation.

Consider the equation $\frac{dy}{dx} = ry + y^3 - y^5$. It is related to hysteresis and has some applications in engineering. Here, r is a parameter. Depending on the value of r, solutions can exhibit different behavior. Here is the code for $r = -0.8$ and $y(-1.5) = 1.3$.

```
% Activity 7: analysing a differential equation using all the tools we have
% learned in MATLAB and in Calculus.
clc
clf

%Set the values of constants and define the equation and the vector field
a=-1.5; b=1.5; c=-1.5; d=1.5; m=24; n=24;
x0=-1.5; y0=1.3;
r=-0.8;
f=@(x,y)r.*y+y.^3-y.^5;
A=@(x,y)x-x+1;
B=@(x,y)f(x,y);

%plot the direction field and the solution:
subplot(1,2,1)
hold on
[x y]=meshgrid(a:(b-a)/m:b,c:(d-c)/n:d);
quiver(x,y,A(x,y),B(x,y),1.5,'linewidth',1,'color','blue','linewidth',1)
[x,y] = ode45(f,[x0 b],y0);
plot(x,y,'linewidth',2,'color','black')
axis([a b c d])

%Notice that the RHS is a function of only y. Let's plot the graph of this
%function to find its zeroes - they are constant solutions of the D.E.
subplot(1,2,2)
hold on
grid on

t=a:(b-a)/1000:b;
y=f(t,t);
plot(t,y,'linewidth',2,'color','red')
axis([a b c d])
```

It produces the following figure:





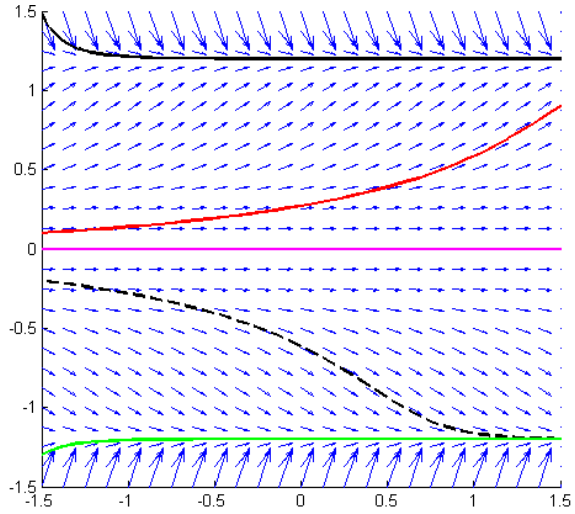
The left graph shows the solution of the initial value problem and the right graph is just the graph of the RHS of the equation, which is a function of only y. For instance, the right graph shows that the only constant solution of the equation is $y = 0$.

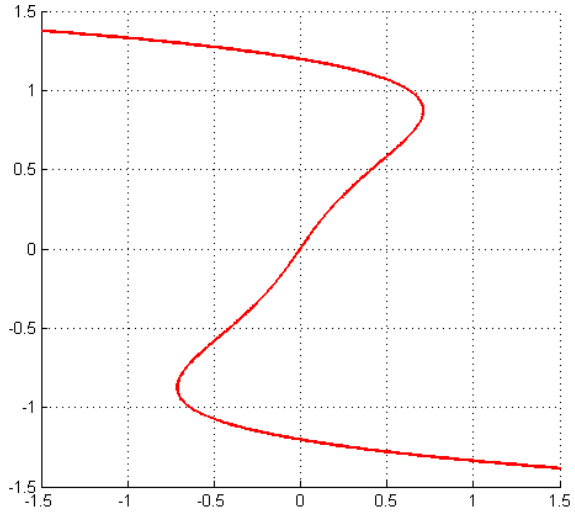
Your task:

First, modify the code above so

- $r = 0.6336$
- It plots five solutions with the initial conditions $y(-1.5) = -1.3,\ -0.2,\ 0,\ 0.1,\ 1.5$ with lines of different style and colour
- The axes of the right graph are switched so that zeroes of the right graph are now on the same level as constant solutions of the left graph.

You'll get the following:





Now analyse the equation. By varying the values of all the parameters and plotting more

solution curves, answer the following questions:

Depending on the value of r, what is the possible number of constant solutions? We have seen one and three above; can you get precisely two, four or five constant solutions? What is the limiting behavior of solutions? Which constant solutions are stable and which are not stable? Why is y^3 in the equation called “the destabilizing term” and why is $-y^5$ called “the stabilizing term”?

There is nothing to submit here. You can just discuss your findings with other students. Make sure that you try positive and negative values of r and find cases with one, three, and five constant solutions. If you don’t know how to interpret you graphs, ask the lab instructor.

[GRADED] Activity 8: Model the fish population.

Objective: use MATLAB for mathematical modelling and write an illustrated report.

We are going to model a fishery. The business model is lazy: we release some fish into the lake, wait for the fish population to grow and then, when the amount of fish is sufficient, start fishing.

Specifically, we assume that the fish population is $P(t)$ (say, measured in hundreds) where t is time (say, measured in weeks). Further, when the fish is left on its own, $P(t)$ satisfies the logistic equation, that is,

$$\frac{dP}{dt} = kP\left(1 - \frac{P}{K}\right),$$

where $K > 0$ is the carrying capacity, and the parameter $k > 0$ shows a relative growth rate (roughly, k is the number of newborn fish per week per existing fish). As we know from Calculus, if the fish was left on its own, then its population would stabilize at K.

However, once the population of fish P_0 is sufficient to support business, we start catching C fish per week. We denote this moment by $t = 0$. It means that now the differential equation of the fish population becomes

$$\frac{dP}{dt} = kP\left(1 - \frac{P}{K}\right) - C, \quad P(0) = P_0$$

The values of the constants K and k are specific to each student (to make sure that it's hardly possible to submit someone else's report as your own). Look at your matriculation number. It has a character, 7 digits, and another character. Your values of K and k are as follows:

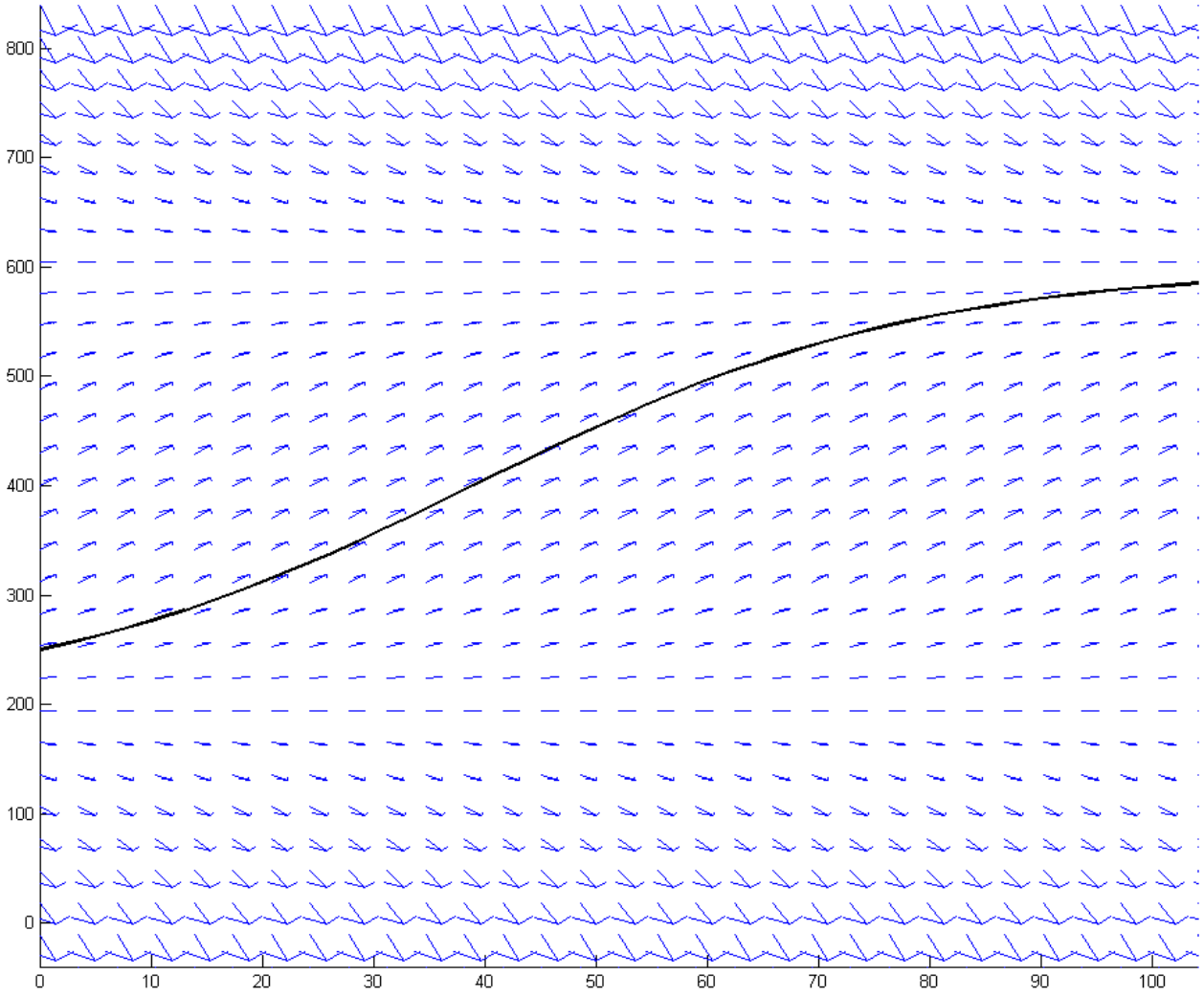
K=1000 + 100×[6th digit], k=0.1+0.01×[7th digit].

For example, if my matriculation number is U1340158F, then K=1000+100×5=1500, and k=0.1+0.01×8=0.18. The values of C and P_0 are to be experimented with.

Your task:

Write a MATLAB code that plots the direction field of the differential equation $\frac{dP}{dt} = kP\left(1 - \frac{P}{K}\right) - C$ and the solution satisfying the initial condition. As a start, you can use the two-year interval for t and the interval $[-0.1K, 1.1K]$ for P, but probably you'll have to change those. The values of K and k are fixed for you and to be found from your matriculation number as described above.

For your reference, here is an example with $K = 800$, $k = 0.1$, $C = 15$, $P_0 = 250$.



Answer the following questions:

1. Depending on the value of $C \geq 0$, what are the equilibrium solutions of the equation $\frac{dP}{dt} = kP\left(1 - \frac{P}{K}\right) - C$? How many constant solutions can we get? You don't need a plot here - please just include calculations.
2. How does the limiting behaviour of the fish population depend on P_0 and C? What is the physical meaning of the equilibrium solutions? You need to support your findings by plots. You'll see that there are a few essentially different cases, so you'll need just a few plots.
3. Propose a valid business plan. Specifically, what is the maximum amount of fish we can weekly get from our lake without the fish eventually dying out? What is

the minimal initial fish population that is able to support commercial fishing? You don't need a plot here as the business plan just depends on your answers to questions 1 and 2.

[Graded Activity for Submission]

Write a report, where you should include your name, your matric number, your values of the constants K and k and answers to **all** the questions above supported by **calculations** and **diagrams** prepared in MATLAB. Save your report as a **PDF** and submit it to your lab instructor.

Please **don't** submit text files, word documents, photographed handwritten reports etc. Equations and diagrams should be very clear and readable. You can use MS Word, Open Office, Google Drive or LaTeX to prepare your report and save it as a **PDF**.

Of course, you can discuss your report with the instructor during the lab session to make sure that it includes everything and doesn't contain obvious errors.

Activity 9: draw the phase portrait.

Objective: apply all the commands we learned so far to draw the phase portrait of a given system of differential equations. We'll have to create two different pieces of MATLAB scripts that are supposed to work at the same time.

Solving systems of differential equations in MATLAB is done by the same commands as solving individual differential equations, but is, unfortunately, a little bit trickier because the input of the "ode45" command must be of a very specific format. It cannot be done by just one piece of MATLAB code. Consider the following system of differential equations:

$$\frac{dx}{dt} = Gxy - kx,\ \frac{dy}{dt} = -Gxy - fy + p$$

that arises in quantum mechanics. Here, x represents the number of laser photons, y represents the number of excited atoms, G is the gain coefficient for stimulated emission, k is the decay rate due to loss of photons, f is the decay rate due to spontaneous emission, and p is the pump strength.

The functions $x(t)$ and $y(t)$ are unknowns and G, k, f, and p are parameters. We are going to set the following values for our demo: $G = 0.1$, $k = 2$, $f = 0.02$, and $p = 3$ (in fact, we don't know if it makes physical sense, but it doesn't matter).

In order to do it in MATLAB properly, we have to create two different MATLAB files. First, please create a new MATLAB script, name it "RHS.m" and copy the following code into it:

```
function ANSWER=RHS(t,X)

%Creating the RHS for a system of differential equations with two unknowns,
%say x and y. The system is dx/dt=A(x,y) and dy/dt=B(x,y). This MATLAB code
%calculates the RHS for any given values of x and y.
%In MATLAB, the proper format for differential equations is one that
%operates a vector input. Here, X=[x; y] is a two-component vector and
%hence X(1)=x and X(2)=y.

%Getting the values of the constants from the global workspace. Without
%this command, MATLAB would look for values of these variables inside the
%function RHS and they are not defined here.
global G k f p;

%Now we construct the function itself:
A=G.*X(1).*X(2)-k.*X(1);
```

```
B=-G.*X(1).*X(2)-f.*X(2)+p;
ANSWER=[A; B];
```

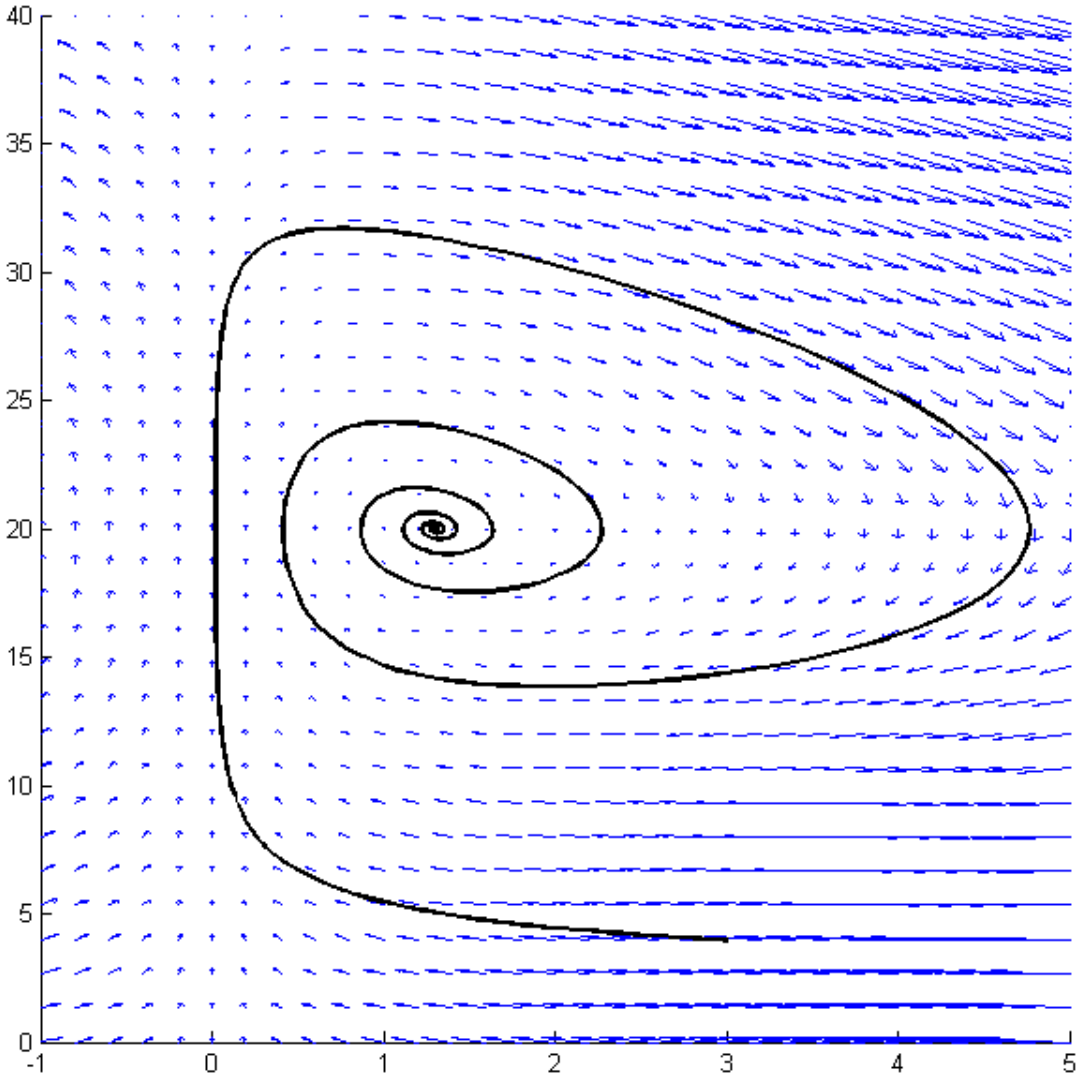
Notice that t is used as the first input of this MATLAB function but doesn't occur in the body of the function. Although the RHS of the system is independent of t, we still have to introduce it into MATLAB - otherwise, the differential equation solver will not work. Now the "RHS.m" defined the function called "RHS". It will compute the RHS of our system of differential equations for any specific values of x and y. Its input is a single variable t and a vector variable X. The vector X has two components, so it is, actually, $X = [x; y]$ and MATLAB will store it as a column vector. The output of the function "RHS" is also a vector $[A(x,y); B(x,y)]$, where A and B represent the right-hand side of the two equations.

Notice that the script "RHS.m" is not executable - you cannot run it by pressing F9. However, once it is saved as a file, you can run commands like "RHS(0.2,[-1; 4])" in MATLAB. Now create a new script again - it will be your working file for this activity. Copy the following code into it:

```
% Activity 9: phase portrait of a system of differential equations
clc
clf
hold on
grid off

%The following are values of constants that could be modified
%The command global is needed to make the values of these constants
%available inside the function RHS.m
global G k f p;
G=0.1
k=2;
f=0.02;
p=3

%Setting the initial conditions x(0)=x0 and y(0)=y0 and the time interval
%to be [t1 t2]. Again, these could be modified.
x0=3; y0=4; t1=0; t2=100;

%The following values affect the plot: we draw the curve [x(t) y(t)] where
%x is in the interval [a,b] and y is in the interval [c d], using the grid
%(m+1) by (n+1) for the vector field.
a=-1; b=5; c=0; d=40; m=30; n=30;

%The following command initializes the vector field:
A=@(x,y)G.*x.*y-k.*x;
B=@(x,y)-G.*x.*y-f.*y+p;
```

```
%Now plotting the vector field using meshgrid and quiver commands:
[x y]=meshgrid(a:(b-a)/m:b,c:(d-c)/n:d);
quiver(x,y,A(x,y),B(x,y),1,'linewidth',1,'color','blue','linewidth',1)

%Now we'll solve the system of differential equations. It's a bit tricky
%because it requires vector input. Here, X=[x; y] is a vector with two
%components. Besides, the function "ode45" with default settings might not
%be precise enough to produce a good graph. We'll make it more precise by
%the additional command "odeset". If you're interested how it works, select
%the word "odeset" and click F1.
options = odeset('RelTol',1e-5);
[t X]=ode45('RHS',[t1 t2],[x0 y0],options);
x=X(:,1);
y=X(:,2);
plot(x,y,'linewidth',2,'color','black')

axis([a b c d])
```

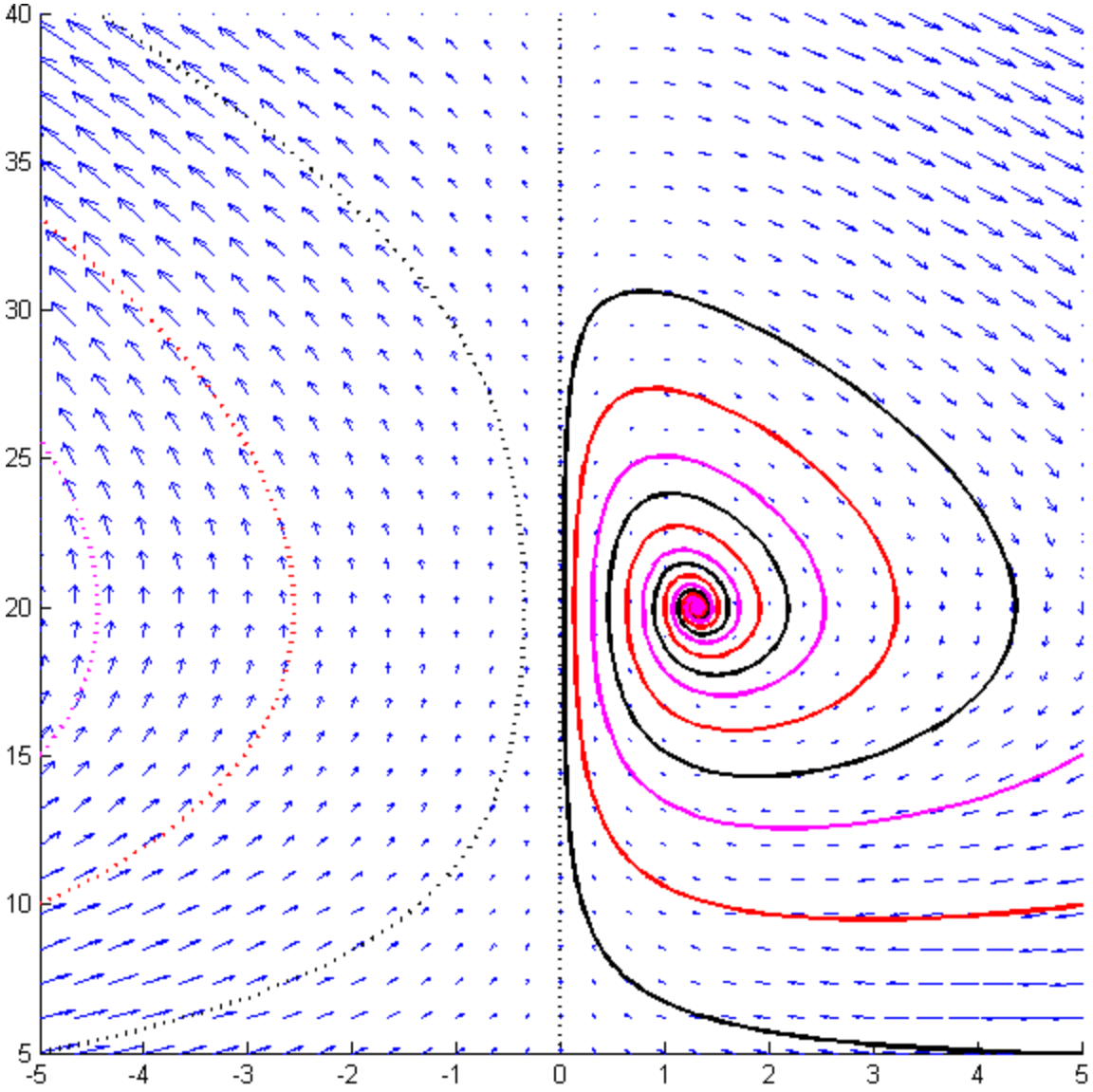
Run this code by pressing F9. You will get the following picture:



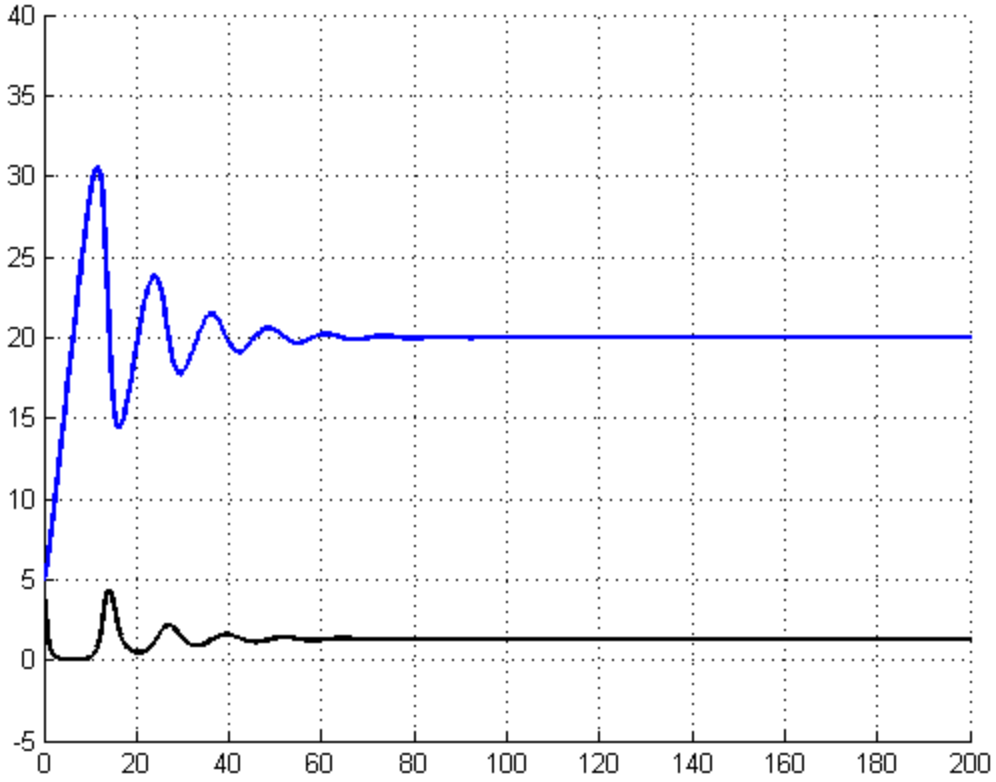
Your task:

1) Modify this code so it plots seven trajectories for the following values of the vector (x_0, y_0):

(0,0), (5,5), (5,10), (5,15), (-5,5), (-5,10), (-5,15). Now you've got the *phase portrait* of the system. Observe that there is a constant solution (X_0, Y_0) with the property that any solution within its neighbourhood is going to converge to (X_0, Y_0). On the other hand, solutions with negative initial values of x tend to infinity.



1) Modify this code to plot x and y against the independent variable t for the initial condition (5,5). How much time does it take until the solution stabilizes? You need to get something like the following picture:



It shows $x(t)$ and $y(t)$ in different colour and we see that the solutions look like constants for $x > 80$ (they are not actually constants, of course, but oscillations become very small). It is also clear that $X_0 \approx 1.3$ and $Y_0 \approx 20$.

There is nothing to submit for this task.

Activity 10: the Lotka-Volterra model of an interspecific competition.

Objective: apply the MATLAB skilled we learned to a particular predator-prey model.

We are going to model the population of two species competing for the same limited resource. For example, you might think of wolves and foxes living in the same forest. We'll use a modification of the Lotka-Volterra model called "Competitive Lotka-Volterra equations":

$$\frac{dx}{dt} = r_1 x\left(1 - \frac{x + C_{12} y}{K_1}\right), \quad \frac{dy}{dt} = r_2 y\left(1 - \frac{y + C_{21} x}{K_2}\right),$$

where $x(t)$ and $y(t)$ are the two populations. Further, we have the following positive constant coefficients:

r_1 and r_2 are the natural growth rates of the respective species,

K_1 and K_2 are the carrying capacities,

C_{12} and C_{21} are the competition coefficients that show the negative impact that the first species has on the second one and, respectively, the negative impact that the second species has on the first one. For instance, if C_{12} is much bigger than C_{21}, then we could think of a scenario when x is a small predator and y is a big predator and the big predator can always win a direct fight with the small one while the harm caused by the small predator is just that they eat the same food. In particular, if $C_{12} = C_{21} = 0$, then we would get two independent logistic equations.

The values of the constants r_1, r_2, K_1, K_2 are specific to each student (to make sure that it's hardly possible to submit someone else's report as your own). Look at your matriculation number. It has a character, 7 digits, and another character. Then

K_1=1000 + 100×[7th digit], K_2=1000 + 100×[6th digit],

r_1=0.1+0.01×[5th digit], r_2=0.1+0.01×[4th digit].

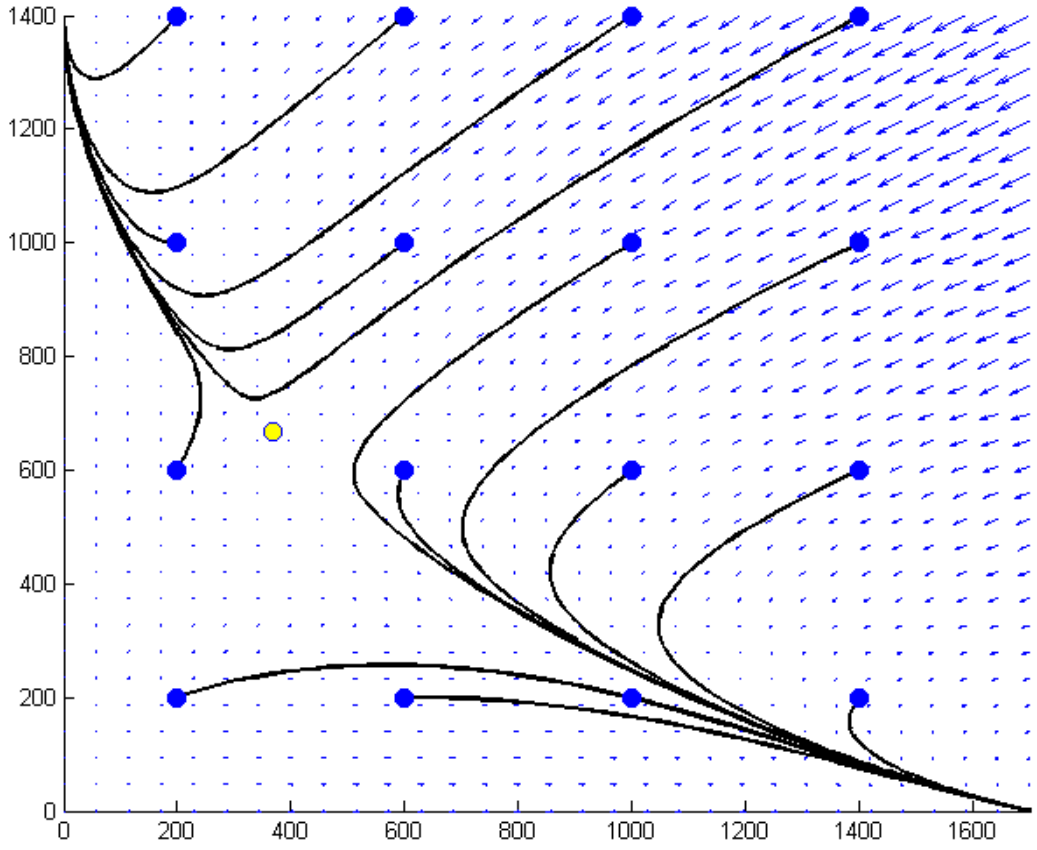
For example, if my matriculation number is U1340158F, then K_1=1000+100×8=1800, K_2=1000+100×5=1500, r_1=0.1+0.01×1=0.11, r_2=0.1+0.01×0=0.1. The values of C_{12} and C_{21} are to be experimented with.

Your task:

Write a MATLAB code that plots the phase portrait of the system. The values of the parameters r_1, r_2, K_1, K_2 are fixed and to be extracted from your matriculation number and you'll need to write a report that describes the behaviour of the system under different values of the coefficients C_{12} and C_{21}. Your are supposed to

1. Find equilibrium solutions analytically. You will notice there are three significantly different cases: $1 - C_{12}C_{21} > 0$, $1 - C_{12}C_{21} = 0$, and $1 - C_{12}C_{21} < 0$.
2. Sketch possible phase portraits in each of the three cases described above. You will notice that there are two-subcases in each case, so you'll need to draw 6 diagrams overall. Please plot a fat dot that represents the nonzero equilibrium.
3. Write a conclusion: what are possible types of behaviour of this system depending on the values of C_{12} and C_{21}? Under which values of the coefficients C_{12} and C_{21} do we observe one of the species dying out? Is it possible that both species die out? When do the both species survive? Are the populations going to approach a particular limit or we'll see cycles like in the classical predator-prey model? Is the limiting behaviour of the system affected by the initial conditions?

For instance, here is an example of a phase portrait that shows that the equilibrium solution is unstable and that depending on the initial conditions, one of the species will die out.



The initial conditions are shown by blue dots and the equilibrium (found analytically) is

shown by the yellow dot.

Remark The graph above shows only one of the possibilities, with some particular values of the parameters.

There is nothing to submit for this task.

Activity 11: phase portrait of a 2nd order differential question.

Objective: apply all the commands we learned so far to draw the phase portrait of a simple harmonic oscillator.

A second-order differential equation is one of the form

$$\frac{d^2x}{dt^2} = f\left(t, x, \frac{dx}{dt}\right).$$

Here, the unknown function is $x(t)$ and t is the independent variable. The LHS is the second derivative of the unknown function and the RHS is some expression containing the independent variable, the unknown function, and its first derivative. Second-order differential equations are very common in mechanics because of Newton's Law

$$F = am.$$

Usually we know how the force depends on the position and the velocity, which gives us F. Then a is the acceleration, i.e., the second derivative of the position.

Suppose the independent variable is not actually present in the equation, which means the equation is just $\frac{d^2x}{dt^2} = f\left(x, \frac{dx}{dt}\right)$. Then we can draw a phase portrait by changing one second-order equation to the system of two first-order equations

$$\frac{dx}{dt} = y,\ \frac{dy}{dt} = f(x,y).$$

Obviously, this system contains the same information as the original equation and the new variable y represents the derivative of the unknown function.

A simple harmonic motion is one satisfying the equation $F = -kx = am$. We can think of an object of mass m attached to a spring moving according to Hooke's Law and without friction. Then k is the constant representing the stiffness of the spring.

You'll need two pieces of MATLAB code to work with this equation. First, create a new script "simple_harmonic.m" and copy the following code into it:

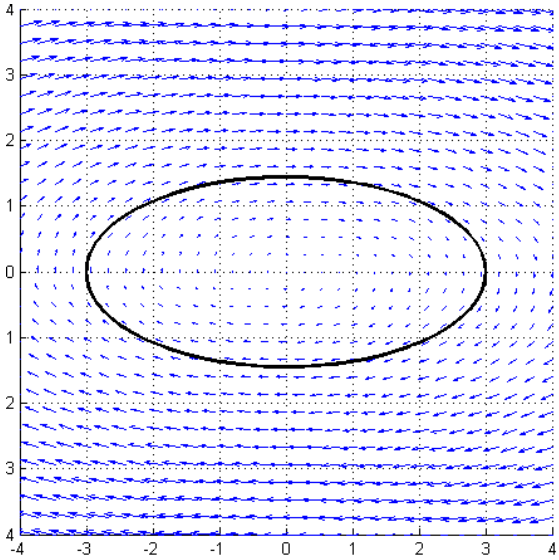
```
%RHS for the simple harmonic oscillator
function dy = simple_harmonic(t,y)
global k M;

dy = zeros(2,1);
dy(1) = y(2);
dy(2) = -k*y(1)/M;
```

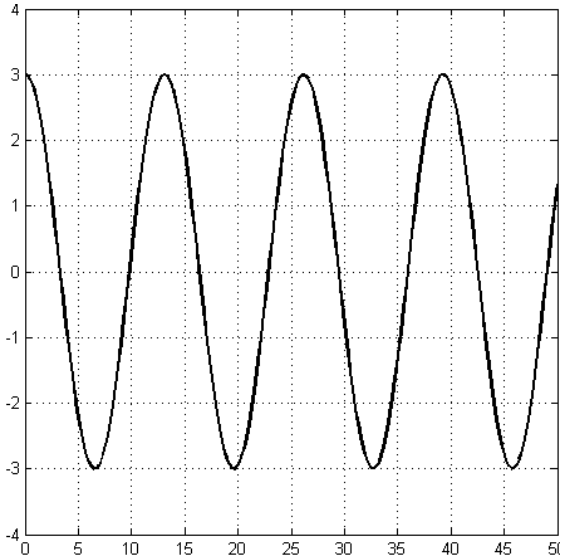
Now create your main script for activity 11 and copy the following code into it:

```
% Activity 11: simple harmonic oscillator
```

```
% You can think of a weight on a spring, where M is the mass of the object
% and k is the coefficient for Hooke's Law.
clc
clf

global k M;
k=1.2; M=5.2;

%The following is the period of motion predicted by the theory:
2*pi*sqrt(M/k)
% Does it match the graph?

%Setting the initial conditions:
x0=3; y0=0;
t1=0; t2=50;

%We draw the curve [x(t) y(t)] for x,y in the rectangle [a,b] by [c,d].
a=-4; b=4; c=-4; d=4; m=30; n=30;

%The following command initializes the vector field:
A=@(x,y)y;
B=@(x,y)-k.*x./M;

subplot(1,2,1)
hold on
grid on

%Now plotting the vector field using meshgrid and quiver commands:
[x y]=meshgrid(a:(b-a)/m:b,c:(d-c)/n:d);
quiver(x,y,A(x,y),B(x,y),1,'linewidth',1,'color','blue','linewidth',1)

options = odeset('RelTol',1e-4);
[t X]=ode45('simple_harmonic',[t1 t2],[x0 y0],options);
plot(X(:,1),X(:,2),'linewidth',2,'color','black')
axis([a b c d])

subplot(1,2,2)
plot(t,X(:,1),'linewidth',2,'color','black')
grid on
axis([t1 t2 c d])
```

This code is supposed to produce the following graph:





The left diagram is the plot of x and y on top of the vector field, that is, the phase portrait. We could sketch more solutions, of course, but they are all going to be ellipses, just like the one plotted. The right diagram is the graph of the function $x(t)$. We see that it's periodic and looks like the cosine. It is, in fact, the cosine up to translation and sketching. Moreover, the period of this function is predicted by the theory and equals $2\pi\sqrt{\frac{m}{k}}$ and we can easily verify it in a this particular case.

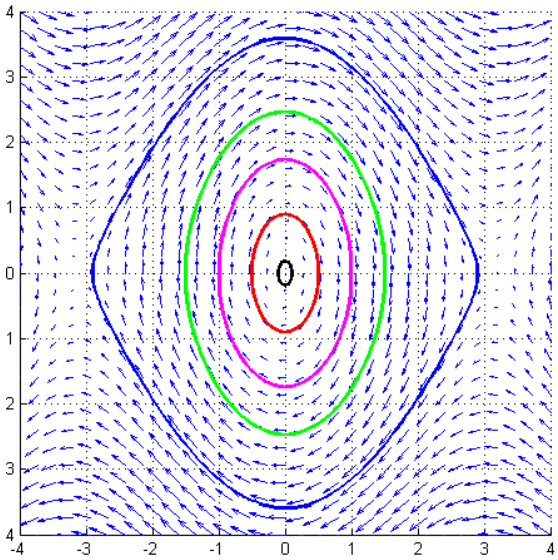
Your task:

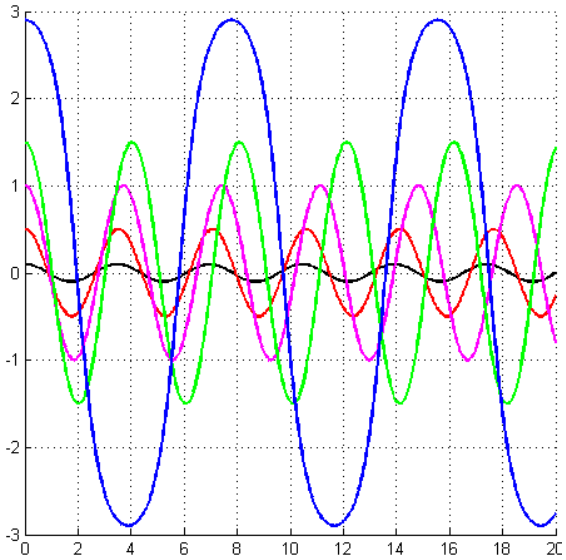
Modify this code so it plots the phase portrait of a simple frictionless pendulum. The differential equation is

$$\frac{d^2x}{dt^2} = -\frac{g}{L}\sin x,$$

where $x(t)$ is the angular displacement, g is the gravitational acceleration and L is the length of the pendulum. Naturally, values of $x(t)$ that have physical meaning are those satisfying $-\pi < x < \pi$. Observe also that for a small value of x, we have $\sin x \approx x$ and hence the equation is very similar to the equation of simple harmonic motion. Our goal is to explore how much the pendulum differs from a simple harmonic motion if the amplitude increases.

Specifically, sketch the phase portrait of this equation. Use the value $l = 3$, though it's not very important (changing l, you just scale the picture). Plot solutions with the following initial values of x: 0.1, 0.5, 1, 1.5, 2.9. Use 0 as the initial value of y (it means that the initial speed is 0). Produce both the phase portrait and the graphs of $x(t)$ against t. Observe that the motion is periodic but the period now depends on $x(0)$ and increases as the amplitude of motion becomes larger. The phase trajectories are not ellipses any more either. You are supposed to get the following diagram:





There is nothing to submit for this task.

[GRADED] Activity 12: model with a 2nd order differential equation.

Objective: apply the MATLAB skill we learned to find out some numerical info about the equation of a damped pendulum.

Recall that the equation $\frac{d^2x}{dt^2}+\frac{g}{L}\sin x=0$ that we used in Activity 11 describes the motion of an idealised pendulum, where the string weighs nothing and there is no friction and no energy loss. The motion is then periodic, where the period depends on the initial conditions (amplitude). Here, L is the length of the string.

Now we are going to introduce some reality by adding damping - an external force (say, due to friction) that causes the motion to stop eventually. The equation becomes

$$\frac{d^2x}{dt^2}+\frac{C}{M}\cdot\frac{dx}{dt}+\frac{g}{L}\sin x=0,$$

where C is the damping coefficient and M is the mass of the object on the string. You can simply google “damped pendulum” to find out how this equation is derived.

The values of the constants C,M,L are specific to each student (to make sure that it’s hardly possible to submit someone else’s report as your own). Look at your matriculation number. It has a character, 7 digits, and another character. Then

C=0.1 +0.01*[7th digit], M=1 + 0.1×[6th digit], L=10+[5th digit],

For example, if my matriculation number is U1340158F, then C=0.1+0.01×8=0.18, M=1+0.1×5=1.5, L=10+1=11.

We are going to use the initial condition $x(0)=\frac{3\pi}{4},\ \frac{dx}{dt}(0)=0$, which means that the initial angular displacement is 135 degrees and the initial speed is zero.

[Graded Activity for Submission]

Your task:

Write a MATLAB code that plots the solution of the damped equation and the solution of the undamped equation together. Answer the following questions:

1. Obviously, damping makes the amplitude of oscillations decrease with time. Does damping affect the period of oscillations too? Does it become smaller or bigger because of damping? Does the length of one full oscillation remain constant, increase, or decrease?
2. How long does it take for the amplitude of oscillations to become smaller than 0.1?

Write a report, where you should include your name, your matric number, your values of the constants and answers to **both** questions above supported by a **diagram** prepared in MATLAB (Yes, this task is simpler than #8 - just one diagram and a few phrases will do). Save your report as a **PDF** and submit it to your lab instructor.

Please **don't** submit text files, word documents, photographed handwritten reports etc. Equations and diagrams should be very clear and readable. You can use MS Word, Open Office, Google Drive or LaTeX to prepare your report and save it as a **PDF**.

Of course, you can discuss your report with the instructor during the lab session to make sure that it includes everything and doesn't contain obvious errors.