MATLAB Programming

Midterm Two

Instructor: 黃世強 (Sai-Keung Wong)

Content

There are four problems. You must answer all of them.

There are templates for some problems. Please use the templates **if you want**.

About demo video and demo programs.

A demo video or a demo program may have bugs. They show or illustrate roughly the results. These results may not be exactly the same as the requirements. They are for your own reference. You must follow the instruction to finish your programs.

Program file name format

Write all your programs in a folder. The folder name is mat_midterm_02_student_ID. For example, if your ID is 12345678, the folder name is mat_midterm_02_12345678.

Zip the folder and upload it.

Write **a program for each problem** in **one file**.

The file name is m02_X_yourStudentID.m, where X is the problem number.

For example, if your student ID is 12345678 and the problem number is 3, then the file name must be **m02_3_12345678.m**.

You must not output all the intermediate results.

Output the results that are required only.

File content header

At the top of the file, write down your name, ID, email address, department, and date. If you do not do so, you receive a score of zero in the corresponding problem.

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Midterm Number: …
% Problem number: …
% Student Name:  …
% Student ID: …
% Email address: …
% Department:
% Date: ….
%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

File content

For each problem, display the problem number before showing the results.

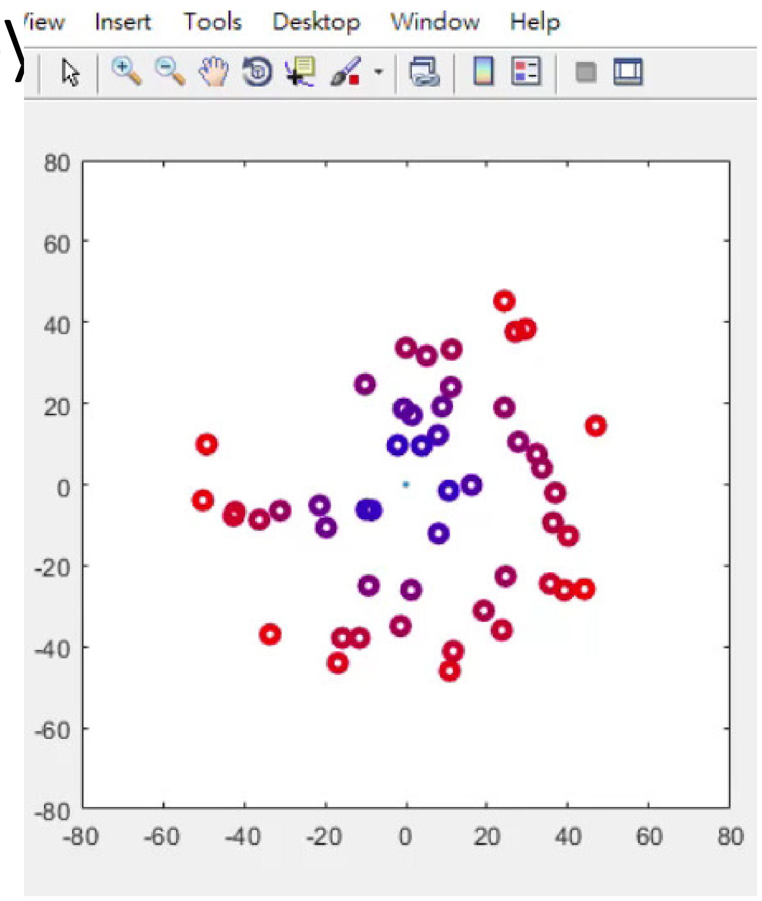
```
close all; clear; clc;          % close all windows
                                % clear variables, and clear screen

disp('Midterm Problem 2.1')     % show Midterm Problem 2.1
```

(25%) Problem 2.1. “Galaxy” sy

There are a particle fixed at center [0 0].

There are (free) particles moving around the center particle.

Color interpolation is adopted to color the colors of the free particles.



Problem 2.1. “Galaxy” system.

- Visualization of the projectile motion of particles . The update rules for a particle p is as follows:

R1) $F = -\frac{p}{||p||}\frac{mM}{1+p^2}$ % force

R2) a = F/m % acceleration

R3) v $\leftarrow$ v + a Δt % velocity

R4) p $\leftarrow$ p + v Δt % position

R5) t $\leftarrow$ t + Δt % time

Initial condition: t = 0. The particle position is randomly generated within a region in a uniform manner. A point p is inside the region iff (if only if), 50>=||p|| >= 10. Here, ||p|| is the distance between p and the origin [0 0].

Initialize the velocity of each particle so that it is orthogonal to the vector (p-[0 0]).

Δt = 0.025. m = 1. M = 10000. Use the same scale along the x- and y-axes. Use pbaspect. **Apply color interpolation** to set the color of a particle based on distance.

Problem 2.1

Velocity normalization. Make the velocity of the particle p orthogonal to the vector p – [0 0].

Assume p = [x y]. Then the velocity is

A) s * [y -x] /||p|| or

B) s * [-y x]/||p||

Set s to 20. If you adopt the same rule (A or B) to set the velocities of all the particles, all the particles rotate in the same direction.

Color interpolation. The color of a particle depends on the distance, d, of the particle to the origin [0 0].

The color is computed as follows:

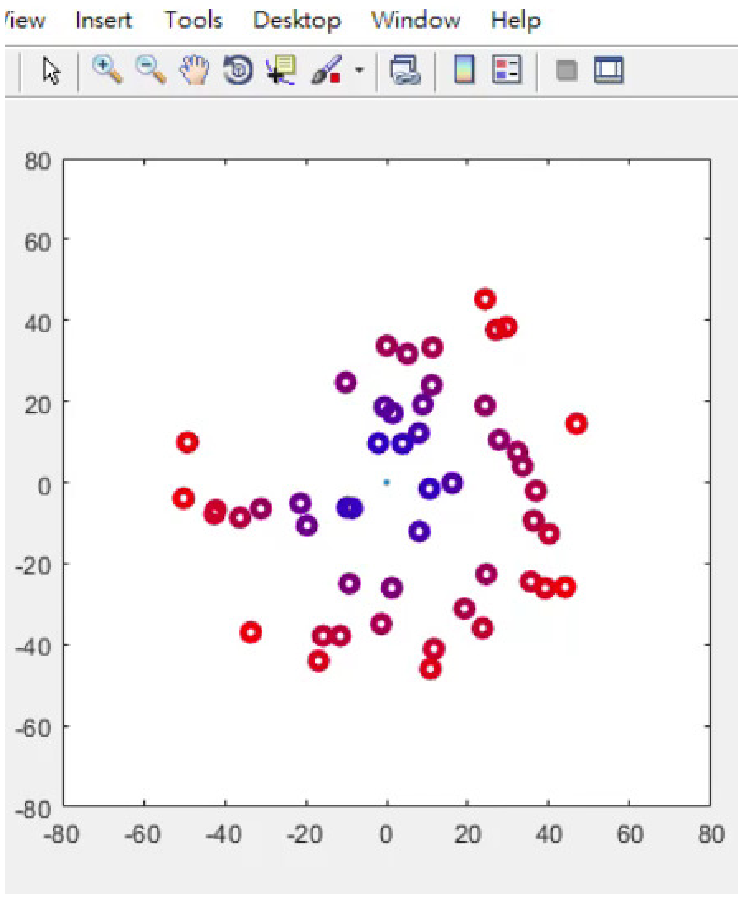
cp0 = [0 0 1]; %blue

cp1 = [1 0 0]; % red

vD = min(1, d/50);

particle_color = cp0 + vD*(cp1-cp0);

Play it to see the animation



Problem 2.1

The major steps:

1. Ask to input the number, n, of particles: [0, 300].
2. If n is zero, show the student ID and quit the program.
3. Show a game play message as follows:

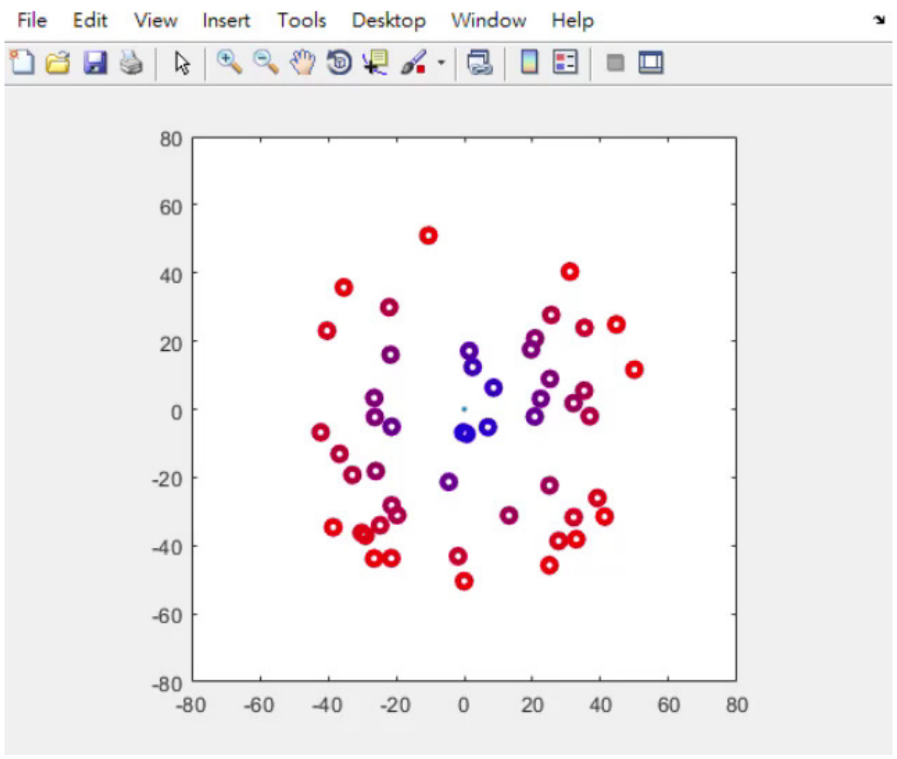
   This system generates n particles. Enjoy the simulation.

4. Perform the simulation until CTRL-C is pressed.

Note: In Step 3, the actual number of n must be shown. For example, if you enter 135 for n. Then the message must be:

This system generates 135 particles. Enjoy the simulation.

Problem 2.1



This demo shows that the particles do not rotate about the origin in the same direction.

You can set them in any way.

But the initial velocities of the particles must orthogonal to (p – [0 0])

Problem 2.1. Marking scheme

[1%] Ask to input n.

[1%] Show student ID before the program is quit.

[1%] Show the game play message.

[7%] The particles are generated correctly. The particle position is randomly generated within a region in a uniform manner. A point p is inside the region iff (if only if), 50>=||p|| >= 10. Here, ||p|| is the distance between p and the origin [0 0].

[5%] Velocity normalization. Make the velocity of the particle p orthogonal to the vector p – [0 0].

[5%] The animation of particles is correct.

[5%] The color interpolation is correct.

(25%) Problem 2.2. Image processing

This program shifts the image, tmp.png, in four directions and modifies the intensity of the pixels in a circular region at the center of the image. Resize the image to [640 640]. If the image is shifted, 10 pixels are shifted at each step. **The program should be implemented using a loop-structure. Don’t quit the program until option 1 is selected.**

Show the options:

1) Show student name and ID. Then quit the program.
2) Shift the image to left
3) Shift the image to right
4) Shift the image to top
5) Shift the image to bottom
6) Turn on or off a spot light on the center. Raise or reduce the intensities of pixels in a circular disk (i.e., filled circle) at the center of the image. The radius of the disk is a half of the width of the image.

Problem 2.2. Marking scheme

[1%] 1) Show student name and ID. Then quit the program.

[4%] 2) Shift the image to left

[4%] 3) Shift the image to right

[4%] 4) Shift the image to top

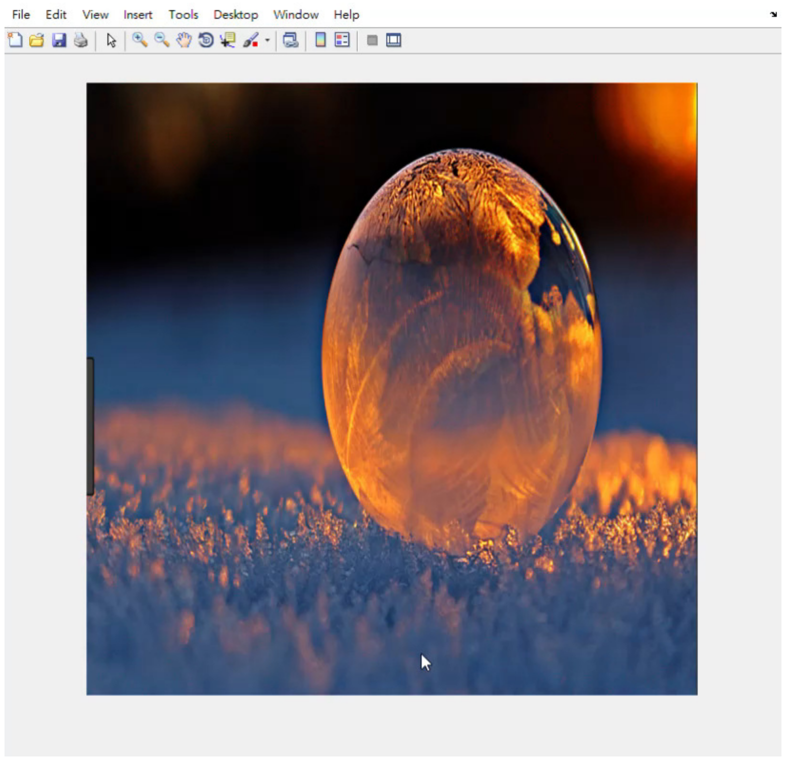
[4%] 5) Shift the image to bottom

[4%] 6) Turn on or off a spot light on the center. Raise or reduce the intensities of pixels in a disk (i.e., filled circle) at the center of the image. The radius of the disk is a half of the width of the image.

[2%] Show the six options on the screen.

[2%] The program does not quit until option 1 is selected.

Problem 2.2. Demo video.



(25%) Problem 2.3.

There are 4 pairs of functions. The two functions of each pair are as follows:

$y_{i,1}(x) = \cos(2*i*x)$, $y_{i,2}(x) = 2\ x \sin(i^2*x)/(x^2+1) - 2$,

for i =1, 2, 3, and 4.

Interpolate linearly the two functions of each pair i by:

$y_i(x) = k\ y_{i,1}(x) + (1-k)\ y_{i,2}(x)$.

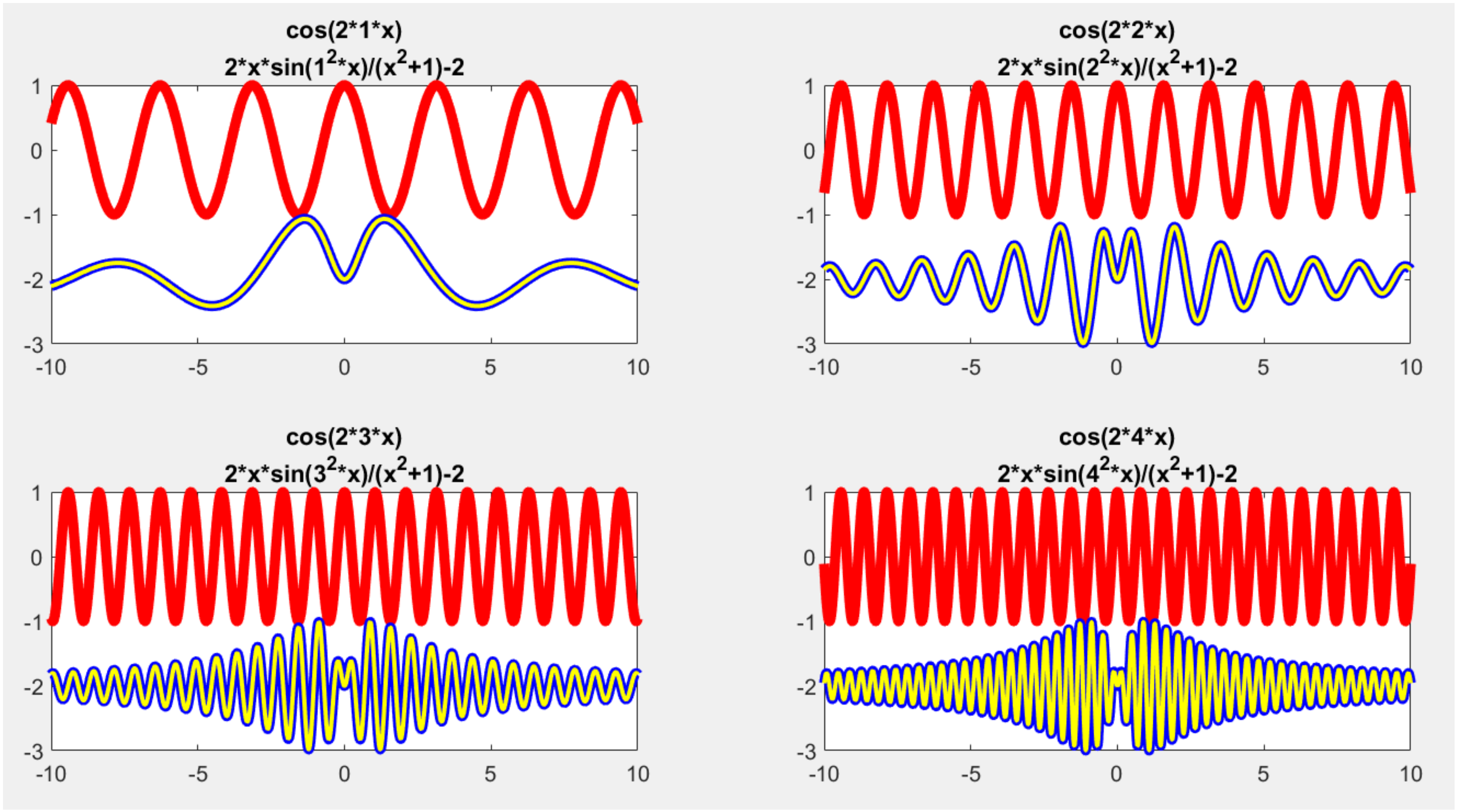
[4%x4] Animate the interpolation process by changing k from 1 to 0, with decrement 0.01. Pause for 1/30 second per step. Show the result on the same figure. Partition the figure into a 2x2 grid. Each pair of functions is shown in a region. The original functions must also be plotted. x in [-10, 10]. The two original curves are plotted in red and blue, respectively.

[5%] Show the title of each figure. The title format must be correct. Set the font size properly

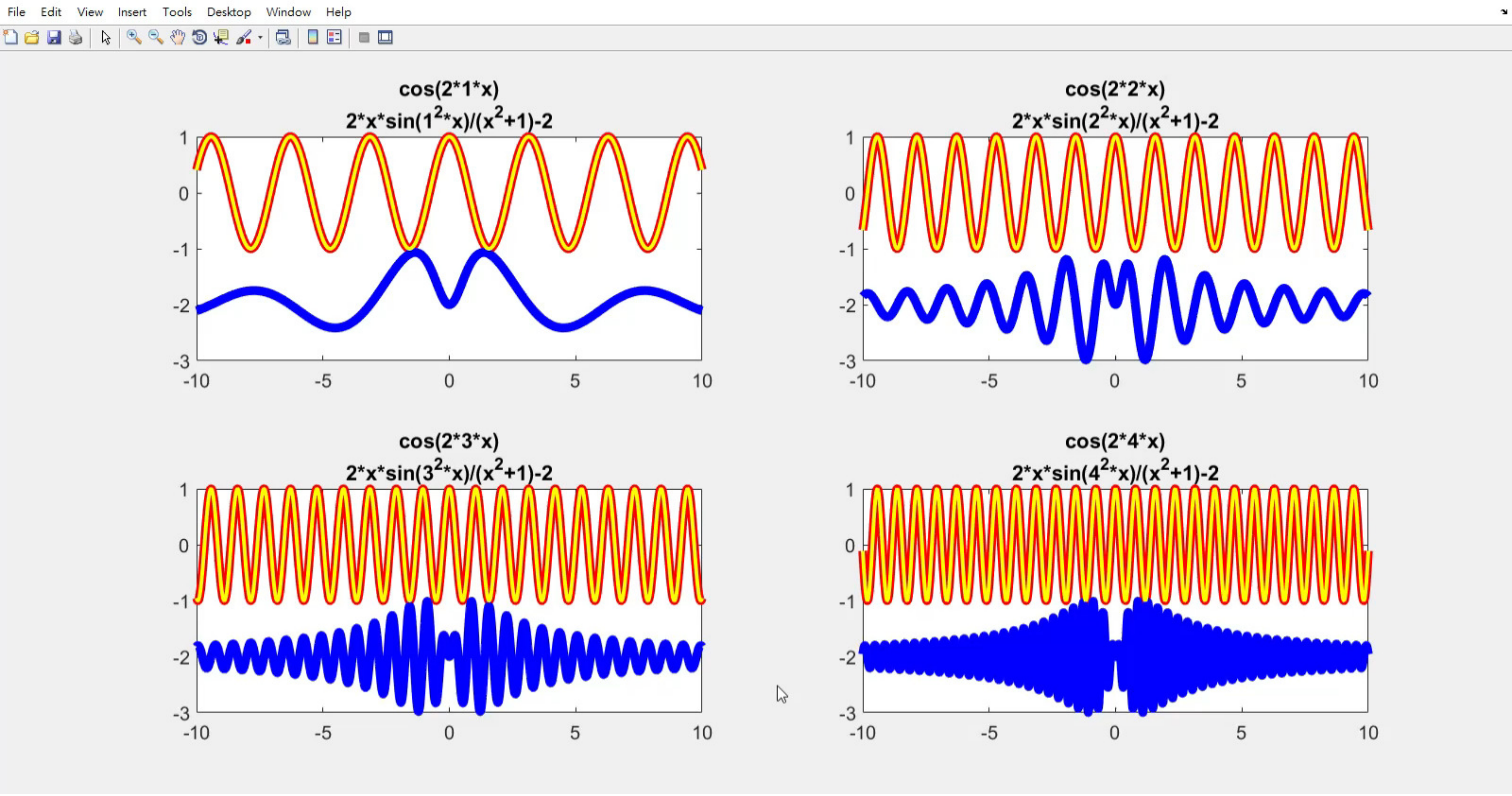
[**4%**] Set the linewidth of the curves properly so that similar results are obtained (see demo). The original curves are thicker than the interpolated curves.

Problem 2.3.

$y_{i,1}(x) = \cos(2*i*x)$, $y_{i,2}(x) = 2\, x \sin(i^2*x)/(x^2+1) - 2$,
for i =1, 2, 3, and 4.



Problem 2.3. Demo video



(25%) Problem 2.4

There are four pairs of functions. Each function can be plotted as a curve.
Randomly generate a set of sample points in a uniform manner inside a rectangle region **X** x **Y**,
where X = [-5, 5], and Y = [-15, 15]. **If a sample point is between two curves**, draw it with the style '.' with green color; **otherwise, it is not shown.**
The four function pairs are as follows:

1. $y_{1,1}$ = sin(x)*sqrt(|x|); $y_{1,2}$ = 2*cos(x)*e^(cos(x^2)).
2. $y_{2,1}$ = sin(x)+x^2cos(x); $y_{2,2}$ = 2*sin(x)*e^(sin(x^2)).
3. $y_{3,1}$ = sin(x)+$\sin^2$(x); $y_{3,2}$ = cos(x)+$\cos^2$(x).
4. $y_{4,1}$ = x^5 cos(x)$e^{-|x|}$; $y_{4,2}$ = x^5 sin(x)$e^{-|x|}$.

The ranges of the axes must be set so that every pair of curves are exactly fitted inside each subfigure.

Problem 2.4

The main process is as follows.

1. Ask to input the number of sample points, n, inside [0, 10000]. If n is not inside the interval, repeat step 1.
2. If n is zero, show the student ID. Then quit the program.
3. Ask to input the option:

   Please input an option

   1) Plot the curves of y(1,1) and y(1,2) in one figure.

   2) Plot two pairs of curves, (y(1,1) and y(1,2)) and (y(2,1) and y(2,2)).

   3) Plot the four pairs of curves.

After an option is finished, go back to step 1.

In option 1, use a 1x1 grid.

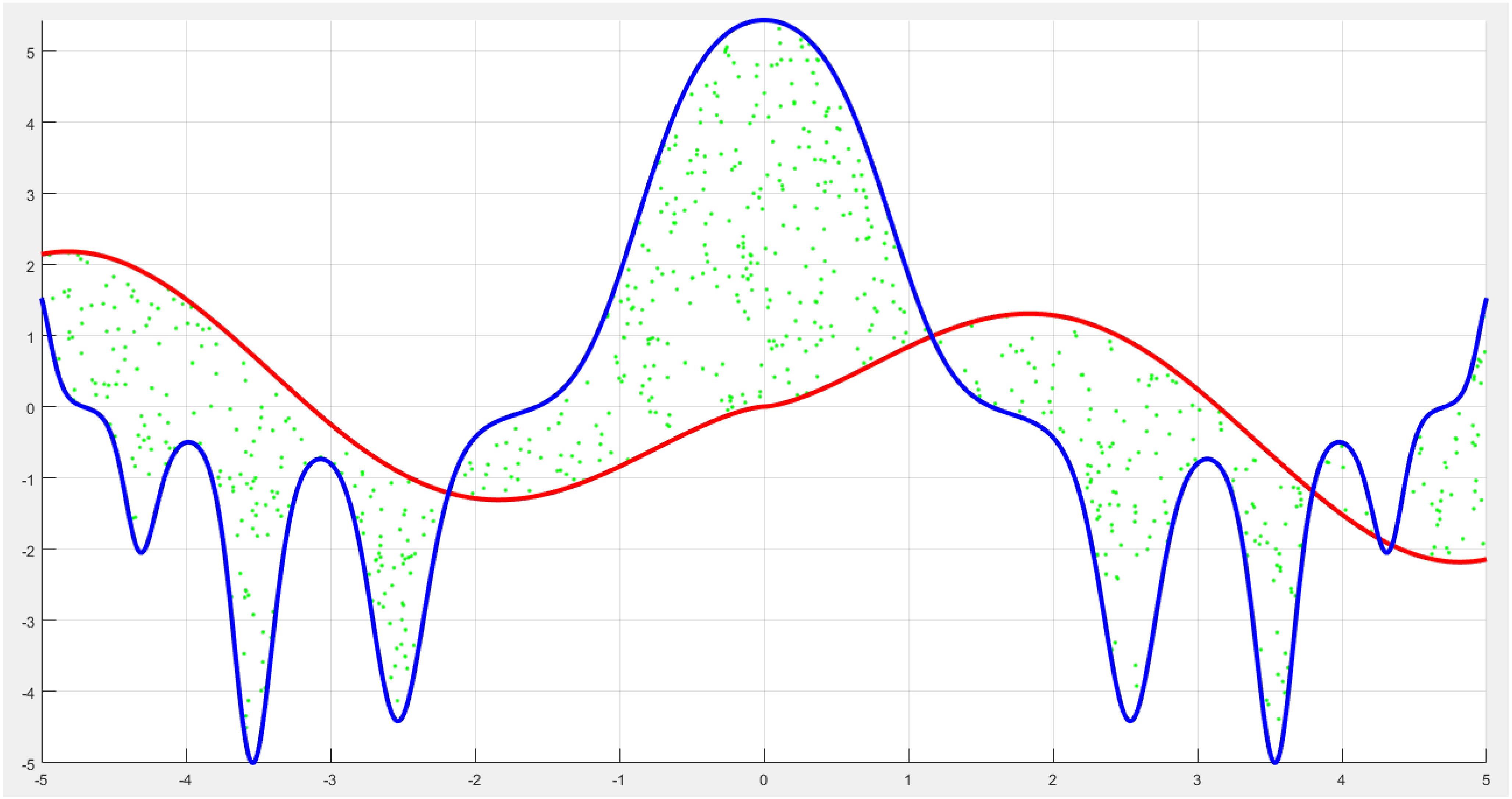
In option 2, use a 2x1 grid. Each pair of curves are shown in one sub-figure in the same order as the function pair index.

In option 3, use a 2x2 grid. Each pair of curves are shown in one sub-figure.

An option must be completed within 10 seconds.

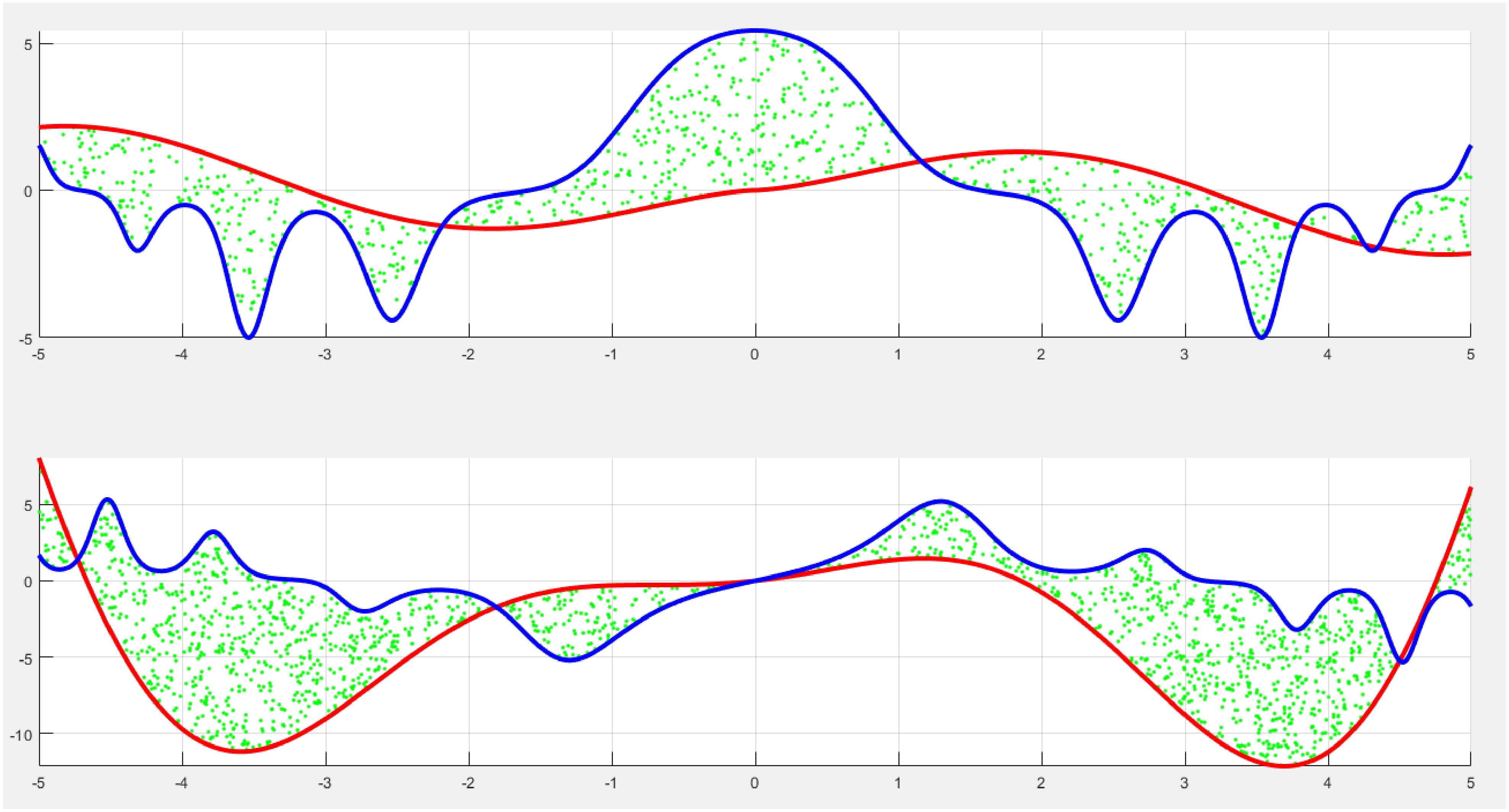
Problem 2.4. Demo

```
Please input n:10000
1) Plot the curves of y(1,1) and y(1,2) in one figure.
2) Plot two pairs of curves, (y(1,1) and y(1,2)) and (y(2,1) and y(2,2)).
3) Plot the four pairs of curves.
Please input an option:1
```


5
4
3
2
1
0
-1
-2
-3
-4
-5
-5
-4
-3
-2
-1
0
1
2
3
4
5

Problem 2.4. Demo

```
Please input n:10000
1) Plot the curves of y(1,1) and y(1,2) in one figure.
2) Plot two pairs of curves, (y(1,1) and y(1,2)) and (y(2,1) and y(2,2)).
3) Plot the four pairs of curves.
Please input an option:2
```

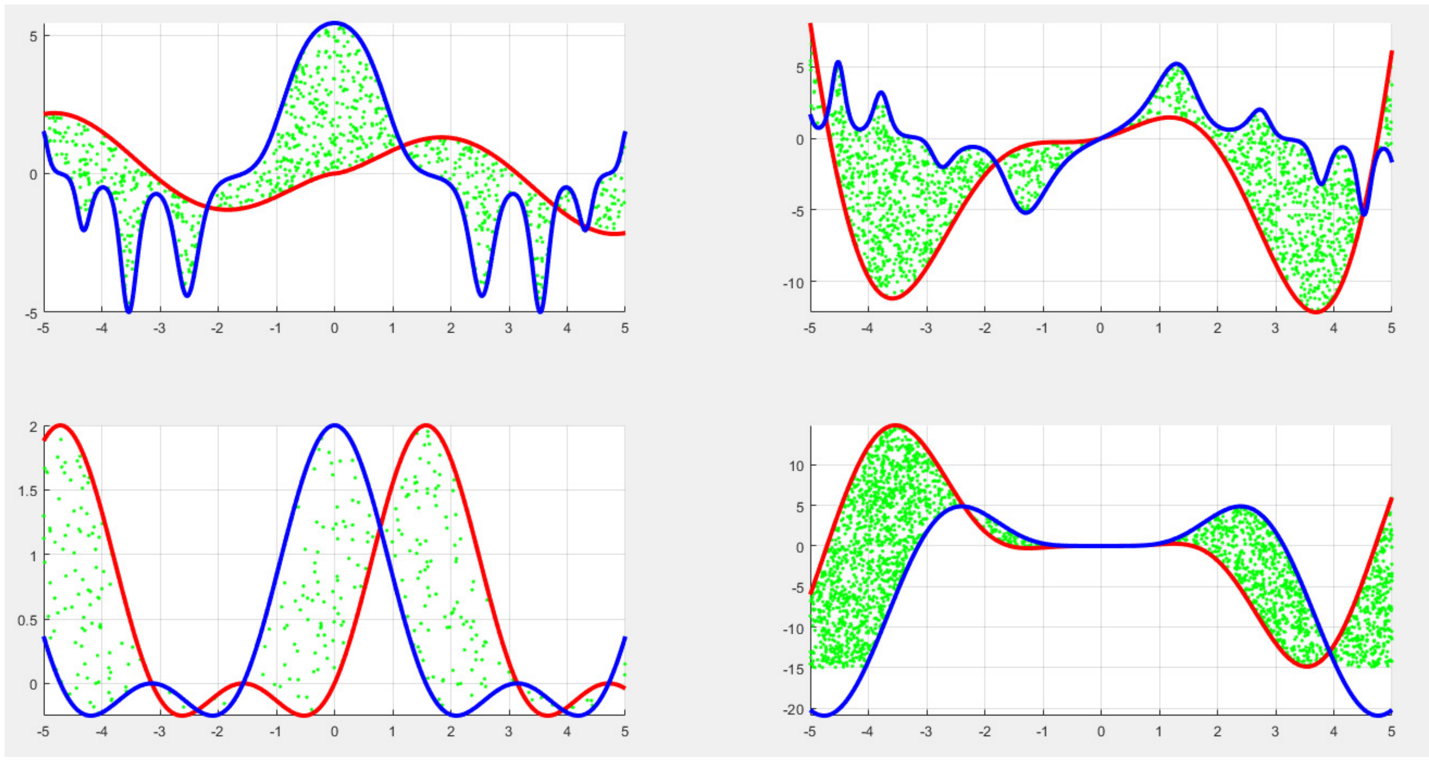

5
0
-5
-5
-4
-3
-2
-1
0
1
2
3
4
5
5
0
-5
-10
-5
-4
-3
-2
-1
0
1
2
3
4
5

Problem 2.4. Demo

```
Please input n:10000
1) Plot the curves of y(1,1) and y(1,2) in one figure.
2) Plot two pairs of curves, (y(1,1) and y(1,2)) and (y(2,1) and y(2,2)).
3) Plot the four pairs of curves.
Please input an option:3
```

The ranges of the axes must be set so that every pair of curves are exactly fitted inside each subfigure.

The min and max y-coordinates of the curves are just fitted inside each subfigure.



Note: there are no samples for y < -15.

Problem 2.4. Marking scheme

The main process is as follows.

1. **[1%]** Ask to input the number of point, n, inside [0, 10000]. If n is not inside the interval, repeat 1.
2. **[1%]** If n is zero, show the student ID. Then quit the program.
3. Ask to input the option for showing the curves:

   [1%] Please input an option

   [4%] 1) Plot the curves of y(1,1) and y(1,2) in one figure.

   [4%] 2) Plot two pairs of curves, (y(1,1) and y(1,2)) and (y(1,1) and y(1,2)).

   [5%] 3) Plot the four pairs of curves.

   [5%] In each option, the sample points inside **`X`** `x` **`Y`** and between two curves are drawn in green color. Use '.' shape.

   [4%] **`The ranges of the axes must be set so that every pair of curves are exactly fitted inside each subfigure.`**

The performance must be completed within 10 seconds. **If not, deduce 10%.**

End

- Enjoy MATLAB Programming.