**Introduction to Scientific Computing, Homework #2**

**Problem 1 Solution**

clear;clc;

figure('units','normalized','outerposition',[0 0 1 1]); % Full screen

subplot(2,2,1);seedsPlot(137.513);

title('Sunflower');set(gca, 'RTick', []);

% To make it a bit more accurate: 137.51(3) (deg)

subplot(2,2,2);seedsPlot(137.45);

title('Spokes1');set(gca, 'RTick', []);

subplot(2,2,3);seedsPlot(137.65);

title('Spokes2');set(gca, 'RTick', []);

subplot(2,2,4);seedsPlot(137.92);

title('Catherine wheels');set(gca, 'RTick', []);

set(gcf, 'Color', 'w'); % white figure background

function seedsPlot(d)

N = 1:500; % Sufficient to show the tendency

theta = pi\*d/180\*N;

rho = sqrt(N);

polarplot(theta, rho, 'ro', 'MarkerSize', 3);

% NOTE: function "polar" is being deprecated. So I use "polarplot" instead

end

**图表, 散点图

描述已自动生成Plots:**

**(Next Page)**

(Next Page)

**Problem 2 Solution**

clear;clc;

fprintf("%d ", The\_date\_difference(20210101,20210105));

fprintf("%d ", The\_date\_difference(20210105,20210101));

fprintf("%d ", The\_date\_difference(20210105,20210106)); % Consecutive days

fprintf("%d ", The\_date\_difference(20120228,20120301)); % Leap year

fprintf("%d ", The\_date\_difference(20110228,20110301)); % Not leap year

fprintf("%d ", The\_date\_difference(20000228,20000301)); % Leap year

fprintf("%d ", The\_date\_difference(19000228,19000301)); % Not leap year

fprintf("%d ", The\_date\_difference(18000128,19050630));

function difference=The\_date\_difference(time1,time2)

% Manually extract the Y/M/D from integers in the form of YYYYMMDD

m2d = [31, 28, 31, 30, 31, 30, 31, 31, 30, 31, 30, 31];

if time1>time2 % inplace swap

time1=bitxor(time1, time2);

time2=bitxor(time2, time1);

time1=bitxor(time1, time2);

end

year1 = fix(time1/10000);year2 = fix(time2/10000);

month1 = rem(fix(time1/100), 100);month2 = rem(fix(time2/100), 100);

day1 = rem(time1, 100);day2 = rem(time2, 100);

isLeapYear = @(n)((rem(n,4)==0&rem(n,100))|(rem(n,400)==0));

difference = 365\*(year2-year1) + sum(isLeapYear(year1:year2-1));

% Whole years between dates

m2d(2) = 28 + isLeapYear(year1);

for i=1:month1-1 % Delete remain days in year1

difference = difference - m2d(i);

end

difference = difference - day1;

m2d(2) = 28 + isLeapYear(year2);

for i=1:month2-1 % Add remain days in year2

difference = difference + m2d(i);

end

difference = difference + day2 + 1;

end

**Example Output:**

5 5 2 3 2 3 2 38504

(Next Page)

**Problem 3 Solution**

clear;clc;

disp(Scores\_ranking(2));

function rank=Scores\_ranking(n)

%% Initialize for File Operation

filename = input('Please input the file name:', 's');

% Must include suffix, like 'score.txt'

% Note: the file MUST under the current working directory.

fid = fopen(filename, 'rt');% Read as a text(ASCII) file

%% Local(in Room) Stuffs

students = fscanf(fid, '%d %d', [2, inf])';% Store all data in a 5n\*2 array

rank = zeros(5\*n, 5);

for i=1:n

room = zeros(5, 4);

room(:,1:2) = sortrows(students(5\*i-4:5\*i,:), 2, 'descend');

% Sort by scores, high-to-low order

% No need to consider the ID column, since all the room will be mixed

% and re-calculate later.

for j=1:5 % Dispatch local rankings

if (j==1 || room(j,2)~=room(j-1,2))

room(j,4) = j;

else

room(j,4) = room(j-1,4);

% Same score, same ranking.

end

end

room(:,3) = repelem(i, 5)'; % Add the column of room number

rank(5\*i-4:5\*i,1:4) = room;

end

%% Generate Global and Final Result

rank = sortrows(rank, [2, 1], {'descend', 'ascend'});

% Sort the result table by both score and ID column.

% If the score is the same, then sort by ID column in non-decreasing order.

for j=1:5\*n % Calculate the final/global ranking for every student

if (j==1 || rank(j,2)~=rank(j-1,2))

rank(j,5) = j;

else

rank(j,5) = rank(j-1,5);

end

end

end

(Next Page)

**User Input:**

Please input the file name:**score.txt**

**Output(match√):**

**22024005 100 1 1 1**

**22024014 100 2 1 1**

**22024001 95 1 2 3**

**22024003 95 1 2 3**

**22024004 85 1 4 5**

**22024012 85 2 2 5**

**22024002 77 1 5 7**

**22024013 65 2 3 8**

**22024011 25 2 4 9**

**22024010 0 2 5 10**

图表, 折线图

描述已自动生成**Plot:**

**Problem 4 Solution (data saved in test.mat)**

clear;clc;

saveSineWave('test');

plotSineWave('test');

% Make sure that the 'test.mat' file is under current working directory.

function saveSineWave(fileName)

x = -2\*pi:pi/100:2\*pi;

y = 5\*sin(x) + 2;

save([fileName, '.mat'], 'x', 'y', '-mat'); % Manually add '.mat' suffix

end

function plotSineWave(fileName)

load([fileName, '.mat'], 'x', 'y', '-mat');

% Specific the variables to load to avoid possible conflict with functions

% or variables that already exist.

plot(x, y, '-k', 'LineWidth', 2);

xlabel('x');ylabel('y'); % Required: axis label

axis auto;

ylim([-4 8]); % Breathing room

set(gca,'XTick',-2\*pi:pi/2:2\*pi)

set(gca,'XTickLabel', ...

{'-2\*pi', '-3\*pi/2', '-pi', '-pi/2', '0','pi/2','pi','3\*pi/2','2\*pi'});

set(gca,'FontSize', 12, 'FontName', 'Consolas'); % Bigger font

grid on;

legend('sin(x)', 'Location','northeast');

end

(Next Page)

**Problem 5 Solution (Test script: glider gun)**

clear;clc;

%%

% Game of Life

% Based on the script teacher provided, the work is to implement 3 core

% functions(at the end of this file).

block = [1 1; 1 1];

boat = [1 1 0; 1 0 1; 0 1 0];

blinker = [1 1 1];

toad = [0 1 1 1; 1 1 1 0];

glider = [1 1 1; 1 0 0; 0 1 0];

LWSS = [0 1 0 0 1; 1 0 0 0 0; 1 0 0 0 1; 1 1 1 1 0];

gun\_left = [

0 0 1 1 0 0 0 0;

0 1 0 0 0 1 0 0;

1 0 0 0 0 0 1 0;

1 0 0 0 1 0 1 1;

1 0 0 0 0 0 1 0;

0 1 0 0 0 1 0 0;

0 0 1 1 0 0 0 0];% For glider gun

gun\_right = [

0 0 0 0 1;

0 0 1 0 1;

1 1 0 0 0;

1 1 0 0 0;

1 1 0 0 0;

0 0 1 0 1;

0 0 0 0 1];

%% Initialization of the basic patterns

% My shapes inserted here.

% Create a gosper glider gun

in = zeros(100);

in (6:7, 2:3) = block;

in (4:10, 12:19) = gun\_left;

in (2:8, 22:26) = gun\_right;

in (4:5, 36:37) = block;

%% Simulation

iterations = 1000;

for i = 1:iterations

image(logical(in));

colormap ([1 1 1; 0 0 0]);

grid off; % Get rid of the grid, get it? A pun.

in = updateCells(in);

pause(0.001);

end

clear;clc;

%% Function implemented

function out = getCell(in, row, col)

%% Get an element from matrix, but with tolerance of array overrun

[maxrow, maxcol] = size(in);

if row<1 || row>maxrow || col<1 || col>maxcol

% Special judge for illegal positions

out = 0;

else

out = (in(row, col)==1);

end

end

function out = countNeighbors(in)

[maxrow, maxcol] = size(in);

out = zeros(maxrow, maxcol);

for x=1:maxrow % Question about this methodology:

% It takes O(n^4) for each period, but maybe it can reduced to O(n^2),

% just by shifting and overlapping the matrix itself? Codes adhered

% after this segment.

for y=1:maxcol

out(x, y) = getCell(in, x-1, y-1) + getCell(in, x-1, y) + ...

getCell(in, x-1, y+1) + getCell(in, x, y-1) + ...

getCell(in, x, y+1) + getCell(in, x+1, y-1) + ...

getCell(in, x+1, y) + getCell(in, x+1, y+1);

% 8 neighbors in 8 directions, not 4

end

end

%% Alternative Method

% Maybe faster.

% out = zeros(maxrow+2, maxcol+2);

% out(1:maxrow,1:maxcol) = in;

% out(1:maxrow,2:maxcol+1) = out(1:maxrow,2:maxcol+1) + in;

% out(1:maxrow,3:maxcol+2) = out(1:maxrow,3:maxcol+2) + in;

% out(2:maxrow+1,1:maxcol) = out(2:maxrow+1,1:maxcol) + in;

% out(2:maxrow+1,3:maxcol+2) = out(2:maxrow+1,3:maxcol+2) + in;

% out(3:maxrow+2,1:maxcol) = out(3:maxrow+2,1:maxcol) + in;

% out(3:maxrow+2,2:maxcol+1) = out(3:maxrow+2,2:maxcol+1) + in;

% out(3:maxrow+2,3:maxcol+2) = out(3:maxrow+2,3:maxcol+2) + in;

% out = out(2:maxrow+1,2:maxcol+1);

end

function out = updateCells(in)

%% Calculate the next frame

neig = countNeighbors(in);

out = (neig==3) | (neig==2 & in);

% Simplified logic.

% Using logical matrix operation, short, easy to understand and fast.

End

(Next Page)

图表, 散点图

描述已自动生成**Problem 5 Runtime Screenshot:**

(Next Page)

**Problem 6 Solution**

clear;clc;

load('AFMdata0001.mat', 'current', 'friction', 'height', '-mat');

plotAFMdata(height, friction, current, 200, 'AFM\_Visualize');

function plotAFMdata(H, F, I, ss, saveName)

%% Arguments Checking

[row, col] = size(H);

assert(row==col);

assert(all(size(H)==size(F)));

assert(all(size(I)==size(F)));

Xcor = linspace(0, ss, row);

Ycor = linspace(0, ss, col); % X-Y surface division

%% Figure Initialization

figure;

set(gcf, 'PaperPositionMode', 'auto');

set(gcf, 'PaperUnits', 'points');

set(gcf, 'Color', 'w'); % Required: white figure background

screen\_rect = get(0, 'ScreenSize');

screen\_width = screen\_rect(3);

screen\_height = screen\_rect(4);

figure\_width = 1000;

figure\_height = 460; % Required figure size

set(gcf, 'position', [ ...

screen\_width/2-figure\_width/2, screen\_height/2-figure\_height/2, ...

figure\_width, figure\_height]);

% Required: Figure in the center of the screen

%% Axes1: F(x, y)

plotFxy = subplot(2, 2, 3);

[X, Y] = meshgrid(Xcor, Ycor);

surf(X, Y, F);

set(gca,'FontSize', 9, 'FontName', 'Consolas');

xlabel('nm');ylabel('nm');zlabel('nm');

title('Friction overlaid height');

shading interp; % Smooth color shifting when hegiht changes

cb1 = colorbar;

title(cb1, 'Friction (mV)');

colormap(plotFxy, "jet"); % Respectively dispatch styles

view(-40, 60); % Required: specific azimuth and elevation

grid off; % Required: no grid

box on; ax = gca;ax.BoxStyle = "full"; % Required: fully bounding box

axis tight; % Required: axis extend no further than the data range

%% Axes2: I(x, y)

plotIxy = subplot(2, 2, 4);

[X, Y] = meshgrid(Xcor, Ycor);

surf(X, Y, I);

set(gca,'FontSize', 9, 'FontName', 'Consolas');

xlabel('nm');ylabel('nm');zlabel('nm');

title('Current overlaid height');

shading interp;

cb2 = colorbar;

title(cb2, 'Current Response (V)');

colormap(plotIxy, "parula");

view(-40, 50); % Required: specific azimuth and elevation

grid off; % Required: no grid

box on; ax = gca;ax.BoxStyle = "full"; % Required: fully bounding box

axis tight; % Required: axis extend no further than the data range

%% Axes3: I(F)

subplot(2, 2, 1);

scatter(F, I, 1, 'black', '.');

set(gca,'FontSize', 9, 'FontName', 'Consolas');

% Required: specific font size&name for \*ALL\* text elements

% Same afterwards

% Note: This must be set after plotting since the plot/scatter reset axes

xlabel('Friction (mV)');

ylabel('Current response (V)');

title('Current response as a function of friction');

axis auto; % Required: Auto axis

%% Axes4: Current histogram

subplot(2, 2, 2);

histogram(I);

set(gca,'FontSize', 9, 'FontName', 'Consolas');

xlabel('Current response (V)');

ylabel('Counts');

title('Current histogram');

xlim([-1 7]); % Required: specific domain of -1 to 7

% no specific range

%% Save&Exit

% Required: save image to 24-bit jpeg file with resolution of

% 300pixels/inch, then close the figure window

print(gcf, '-djpeg', '-r300', saveName);

% -djpeg stands for 24-bit color depth implicitly

% Note: MATLAB for Windows: 1/96 inch/pixel

% Therefore expected image size is 3125 x 1437

% 1000/96\*300 = 3125

% 460/96\*300 = 1437.5

%% Alternative

% sgtitle('Figure saved as image!', ...

% 'Color','b', 'FontSize', 12, 'FontName', 'Consolas');

% !Only available in version MATLAB 2018b or later

pause(3);

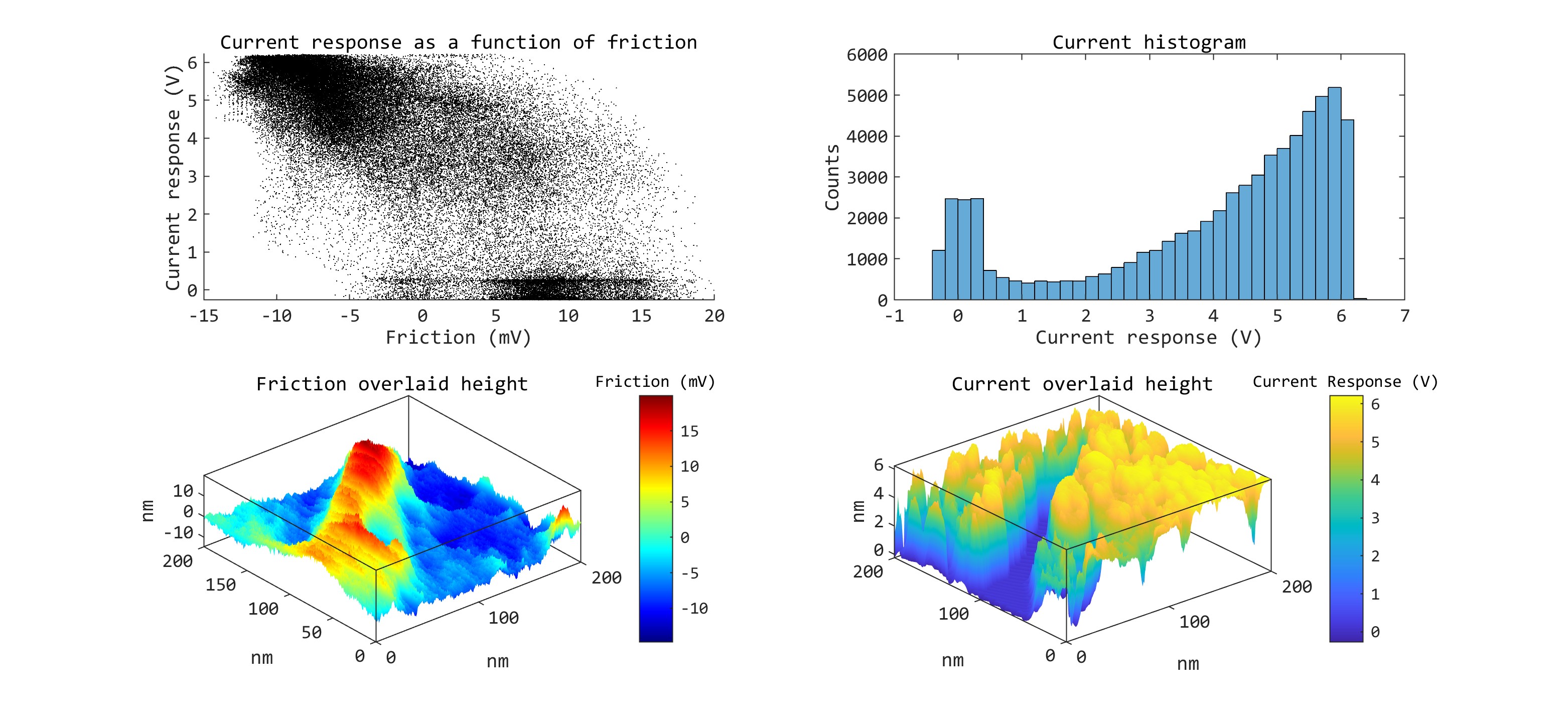
% in case when your computer is too fast to have a glimpse of the figure!

close(gcf);

end

(Next Page)

**Saved image (AFM\_Visualize.jpg, 3125x1437)**



(The End)