*Yunus Emre Soysal*
*200101065*

# HOMEWORK 3 REPORT

## 1)

Aşağıda görmüş olduğunuz ilk görselde yazmış olduğum kod yer almaktadır. Bu kod bir veri seti oluşturur ve bu veri setinin ortalamasını ve standart sapmasını hesaplar. İlk satır, randn fonksiyonunu kullanarak ortalama değeri 2 ve standart sapması 5 olan bir normal dağılıma sahip rasgele sayılar içeren bir 500x1 boyutunda bir vektör oluşturur. İkinci ve üçüncü satırlar, oluşturulan vektörün ortalamasını ve standart sapmasını hesaplar. mean fonksiyonu vektörün ortalamasını bulurken, std fonksiyonu standart sapmayı hesaplar. Son olarak, disp fonksiyonu ile hesaplanan örneklem ortalaması ve standart sapması ekrana yazdırılır.

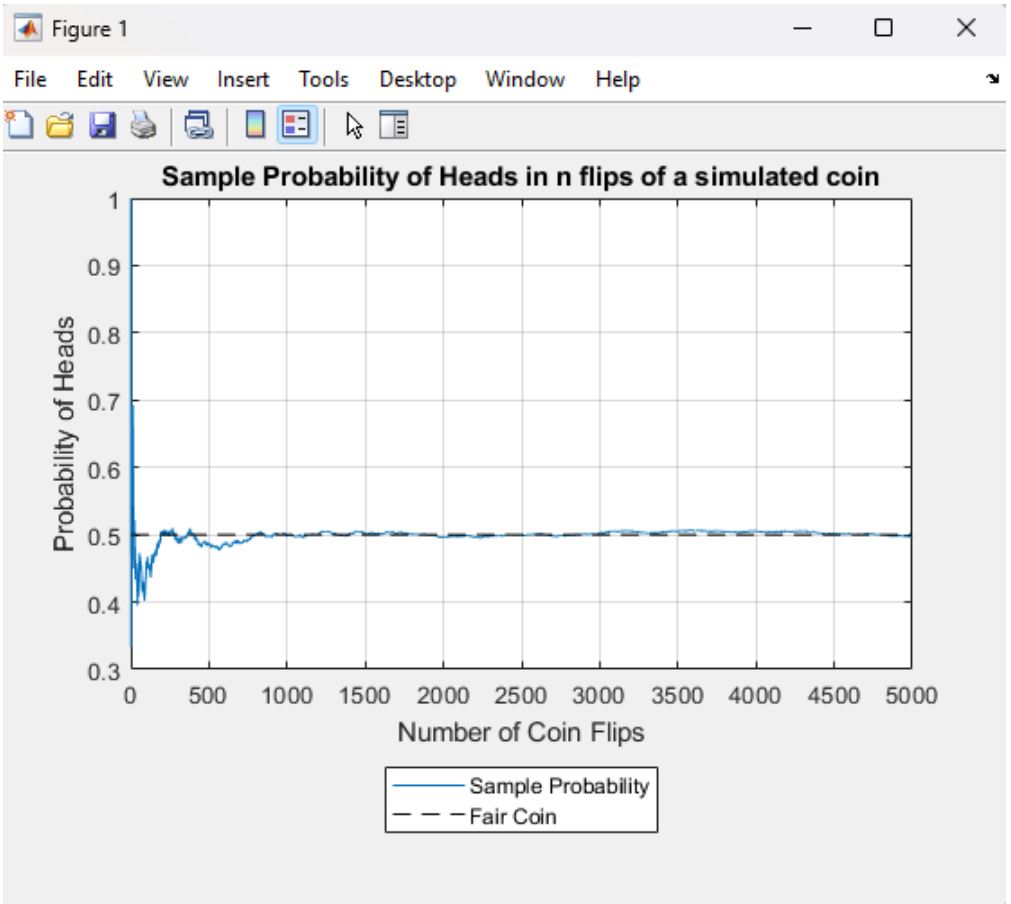
```
vec = 5 * randn(500, 1) + 2;

sample_mean = mean(vec);
sample_std = std(vec);

disp(['Sample Mean: ', num2str(sample_mean)]);
disp(['Sample Standard Deviation: ', num2str(sample_std)]);
```

```
Command Window
>> vec = 5 * randn(500, 1) + 2;

sample_mean = mean(vec);
sample_std = std(vec);

disp(['Sample Mean: ', num2str(sample_mean)]);
disp(['Sample Standard Deviation: ', num2str(sample_std)]);
Sample Mean: 2.0832
Sample Standard Deviation: 5.0617
fx >>
```

**2)**

Bu kod, bir madeni para atışlarının sonuçlarını simüle eder ve her bir atış sonrasında başların gelme olasılığını hesaplar. Sonrasında, atış sayısına bağlı olarak başların gelme olasılığının nasıl değiştiğini görselleştirir.

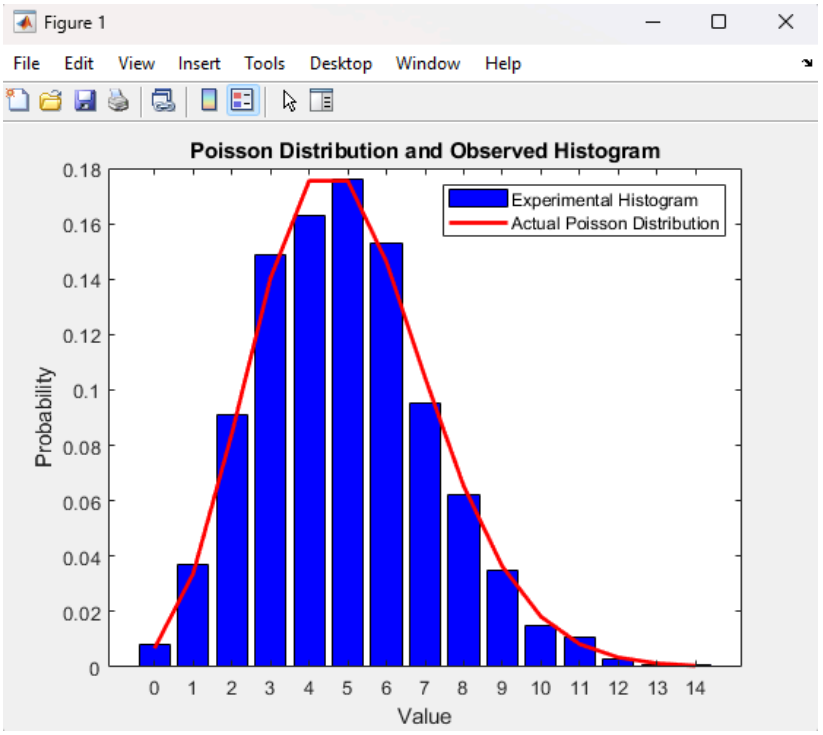
```
num_flips = 5000;

coin_flips = rand(1, num_flips) < 0.5;

running_prob_heads = cumsum(coin_flips) ./ (1:num_flips);

plot(1:num_flips, running_prob_heads);
hold on;

plot([1, num_flips], [0.5, 0.5], 'k--');

xlabel('Number of Coin Flips');
ylabel('Probability of Heads');
title('Sample Probability of Heads in n flips of a simulated coin');

legend('Sample Probability', 'Fair Coin', 'Location', 'southoutside');

grid on;

hold off;
```



**3)**

Bu kod, Poisson dağılımını simüle eder ve bu dağılımın gözlemlenen histogramını çizer. İlk üç satır, Poisson dağılımının olasılık yoğunluk fonksiyonunu (poisspdf 2) ve kümülatif dağılım fonksiyonunu (poiss_cdf) tanımlar. Bu fonksiyonlar lambda parametresini alır ve x değerlerine göre hesaplamalar yapar. Sonraki satırlarda, Poisson dağılımından rastgele örnekler alınır. poissrnd 2 fonksiyonu, lambda parametresini alır ve bu parametre için bir Poisson dağılımından rastgele bir değer döndürür.

```
poisspdf2 = @(x, lambda) (lambda.^x)*exp(-lambda)./factorial(x);
x = 0:100;
x = x';
poiss_cdf = @(lambda) [0; cumsum(poisspdf2(x, lambda))];
poissrnd2 = @(lambda)  find(poiss_cdf(lambda) < rand(), 1, 'last')-1;

num_samples = 1000;
lambda = 5;
samples = zeros(num_samples, 1);
for ii = 1:num_samples
    samples(ii) = poissrnd2(lambda);
end

[N, X] = hist(samples, 0:max(samples));
prob_mass = N / sum(N);

bar(X, prob_mass, 'b');
hold on;

pmf = poisspdf2(X, lambda);
plot(X, pmf, 'r', 'LineWidth', 2);

xlabel('Value');
ylabel('Probability');
title('Poisson Distribution and Observed Histogram');
legend('Experimental Histogram', 'Actual Poisson Distribution');
hold off;
```



**4)**

İlk kısım, bir hücre dizisi oluşturur. Her bir satırda bir kişinin adı, soyadı ve maaşı bulunmaktadır. İkinci kısım, 'Sarah' isminin indeksini bulur ve ardından 'Brown' soyadını 'Meyers' olarak günceller. Üçüncü kısım, 'Pat' isminin indeksini bulur ve ardından maaşını 50000 artırır.

```
data = {'Joe', 'Smith', 30000;
        'Sarah', 'Brown', 150000;
        'Pat', 'Jackson', 120000};

disp('Original cell array:');
disp(data);

sarah_index = find(strcmp(data(:,1), 'Sarah'));

data{sarah_index, 2} = 'Meyers';

disp('Cell array after Sarah changes her last name:');
disp(data);

pat_index = find(strcmp(data(:,1), 'Pat'));

data{pat_index, 3} = data{pat_index, 3} + 50000;

disp('Cell array after Pat gets a raise:');
disp(data);
```

Command Window

```
Original cell array:
    {'Joe'  }    {'Smith'  }    {[ 30000]}
    {'Sarah'}    {'Brown'  }    {[150000]}
    {'Pat'  }    {'Jackson'}    {[120000]}

Cell array after Sarah changes her last name:
    {'Joe'  }    {'Smith'  }    {[ 30000]}
    {'Sarah'}    {'Meyers' }    {[150000]}
    {'Pat'  }    {'Jackson'}    {[120000]}

Cell array after Pat gets a raise:
    {'Joe'  }    {'Smith'  }    {[ 30000]}
    {'Sarah'}    {'Meyers' }    {[150000]}
    {'Pat'  }    {'Jackson'}    {[170000]}

fx >>
```

**5)**

Bu kod, mevcut dizindeki dosyaların listesini alır ve her dosyanın adını ve boyutunu ekrana yazdırır.

İlk kısım, dir fonksiyonunu kullanarak mevcut dizindeki dosyaların bir listesini oluşturur ve bu listeyi a değişkenine atar.

size fonksiyonu, a değişkeninin boyutunu hesaplar ve size_a değişkenine atar.

fieldnames fonksiyonu, a yapısının alan adlarını (name, bytes, isdir, vb.) alır ve field_names değişkenine atar.

for döngüsü, a değişkenindeki her bir öğe için döner. Eğer öğe bir dizin değil ise (~a(i).isdir), dosyanın adını ve boyutunu ekrana yazdırır.

```
a = dir;

size_a = size(a);

field_names = fieldnames(a);

for i = 1:numel(a)
    if ~a(i).isdir
        fprintf('File %s contains %d bytes\n', a(i).name, a(i).bytes);
    end
end
```

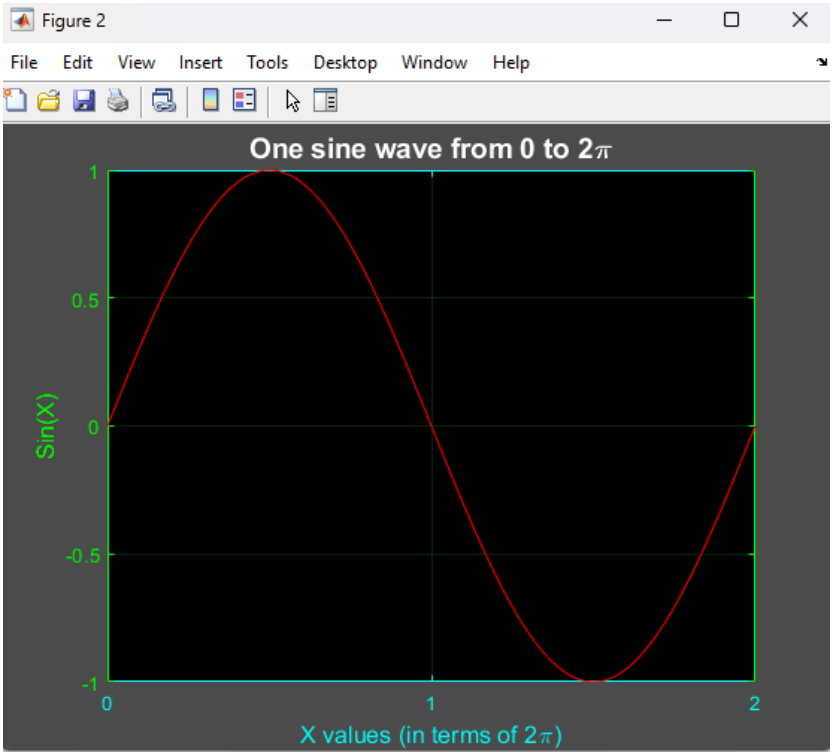
```
function displayDir()
    % Get the contents of the current directory
    a = dir;

    % Loop through all elements of 'a'
    for i = 1:numel(a)
        % Check if the element is not a directory
        if ~a(i).isdir
            fprintf('File %s contains %d bytes\n', a(i).name, a(i).bytes);
        end
    end
end
```

```
File AverageGrades.m contains 832 bytes
File Main.m contains 1006 bytes
File displayDir.m contains 330 bytes
File poisson_workaround.m contains 1330 bytes
fx >>
```

**6)**

Bu kod, 0 ile 2π arasında bir sinüs dalgası oluşturur ve çizimini özelleştirir. Çizimin üzerindeki ana değişiklikler arasında renklerin, eksenlerin ve arka planın özelleştirilmesi bulunmaktadır. Başlık, eksen etiketleri ve ızgara da eklenir ve metin biçimlendirmesi ile düzenlenir. Bu kod, bir grafik çizimini daha belirgin ve estetik hale getirmek için MATLAB'ın çeşitli grafik özelliklerini kullanır.

```
x = linspace(0, 2*pi, 100);
y = sin(x);

figure;
plot(x, y, 'r');

xlim([0, 2*pi]);

set(gca, 'xtick', [0 pi 2*pi], 'xticklabel', {'0', '1', '2'});

set(gca, 'ytick', -1:0.5:1);

grid on;

set(gca, 'xcolor', 'cyan', 'ycolor', 'green', 'color', 'black');

set(gcf, 'color', [.3 .3 .3]);

title('\fontsize{14}\bf\color{white}One sine wave from 0 to 2\pi');

xlabel('\fontsize{12}\color{cyan}X values (in terms of 2\pi)');
ylabel('\fontsize{12}\color{green}Sin(X)');

editmenu(gcf, 'CopyOptions', 'figure background color', 'UseFigureColor');
```



7)

Bu kod, belirli bir JPEG görüntü dosyasını alır ve öncelikle görüntüyü yeniden boyutlandırır, ardından kırmızı, yeşil ve mavi renk kanallarını ayrıştırır. Daha sonra, orijinal görüntüyü, kırmızı kanalı sağ üstte, yeşil kanalı sol altta ve mavi kanalı sağ altta olacak şekilde birleştirerek bir kompozit görüntü oluşturur. Son olarak, bu kompozit görüntüyü özel bir ölçekte ekrana çizer.

```
function im = displayRGB(filename)

    img = imread(filename);

    [h, w, ~] = size(img);
    if h > w
        new_h = 800;
        new_w = round(w * (new_h / h));
    else
        new_w = 800;
        new_h = round(h * (new_w / w));
    end
    img_resized = imresize(img, [new_h, new_w], 'bicubic');

    red_layer = img_resized(:, :, 1);
    green_layer = img_resized(:, :, 2);
    blue_layer = img_resized(:, :, 3);

    composite_img = zeros(2 * new_h, 2 * new_w, 3, 'uint8');
    composite_img(1:new_h, 1:new_w, :) = img_resized;
    composite_img(1:new_h, new_w+1:end, 1) = red_layer;
    composite_img(new_h+1:end, 1:new_w, 2) = green_layer;
    composite_img(new_h+1:end, new_w+1:end, 3) = blue_layer;

    figure, imshow(composite_img, 'InitialMagnification', 'fit'), title('Composite Image');
    im = composite_img;
end
```

```
im = displayRGB('anıtkabir.jpg');
```