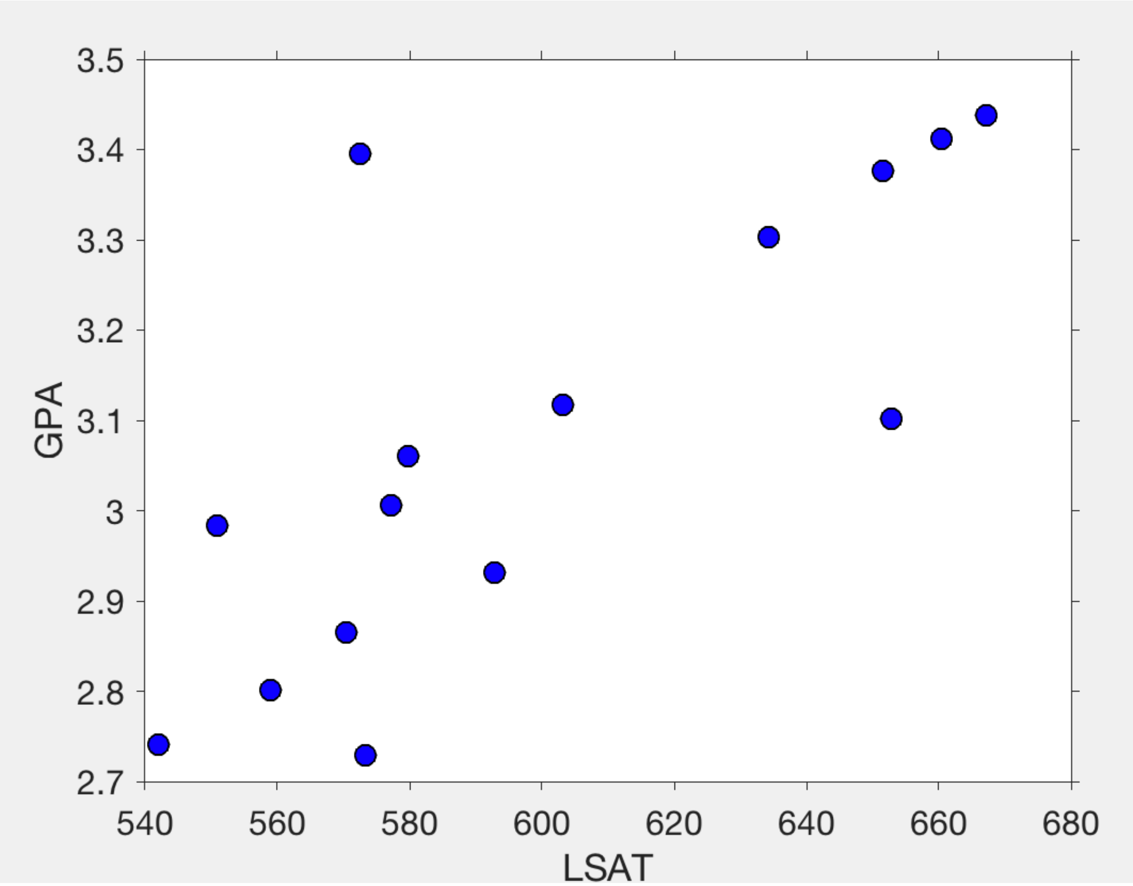
A. The file “GPAvsLSAT.csv” (in “Files/Homework” on CourseWorks) contains the GPA and LSAT score pairs digitized from the figure on p. 118 of Diaconis and Efron 1983 (see “Files/Readings” on CourseWorks). Read the data using the “csvread” function of Matlab and plot GPA versus LSAT. Make sure that your plot compares closely to the figure in the paper.

The plot:

```
clearvars;

% HW 3 - A
% read the data
D=csvread('GPAvsLSAT.csv');
x = D(:,1);
y = D(:,2);

% plot the data
figure(1);
plot(x,y,'ko','MarkerFaceColor','b','MarkerSize',10);
set(gca,'FontSize',16,'TickDir','out');
xlabel('LSAT');
ylabel('GPA');
```



B. Compute and report the correlation coefficient r and the p-value (use the “corrcoef” function in Matlab). You should get a value of r close to that mentioned in the paper (0.776) and a low p-value, meaning that r is unlikely to be zero.

```
% HW 3 - B
% corrcoef:
https://www.mathworks.com/help/matlab/ref/corrcoef.html

[R,P,RL,RU] = corrcoef(x,y);

r = R(1,2);
p = P(1,2);

fprintf('r = %g, p = %g\n',r,p);
```

The output is:

```
r = 0.764199, p = 0.000909279
```

C. Obtain 1,000 bootstrap samples of GPA and LSAT pairs using the “samplebootstrap.m” function in “Files/Homework” on CourseWorks. Make sure you understand what this function does by giving it as input something like `x=[1 2 3 4 5]` and `y=[10 20 30 40 50]`. Call the function several times with the same input and observe the output vectors.

```
% HW 3 - C
x = [1 2 3 4 5];
y = [10 20 30 40 50];

[X,Y] = samplebootstrap(x,y);
```

Tried for several times and get

```
X =


     1     3     5     4     5


Y =


    10    30    50    40    50
```

```
X =

     4     1     5     5     4


Y =

    40    10    50    50    40
```

The 1000 bootstrap samples of GPA and LSAT pairs are in the (D) part below.

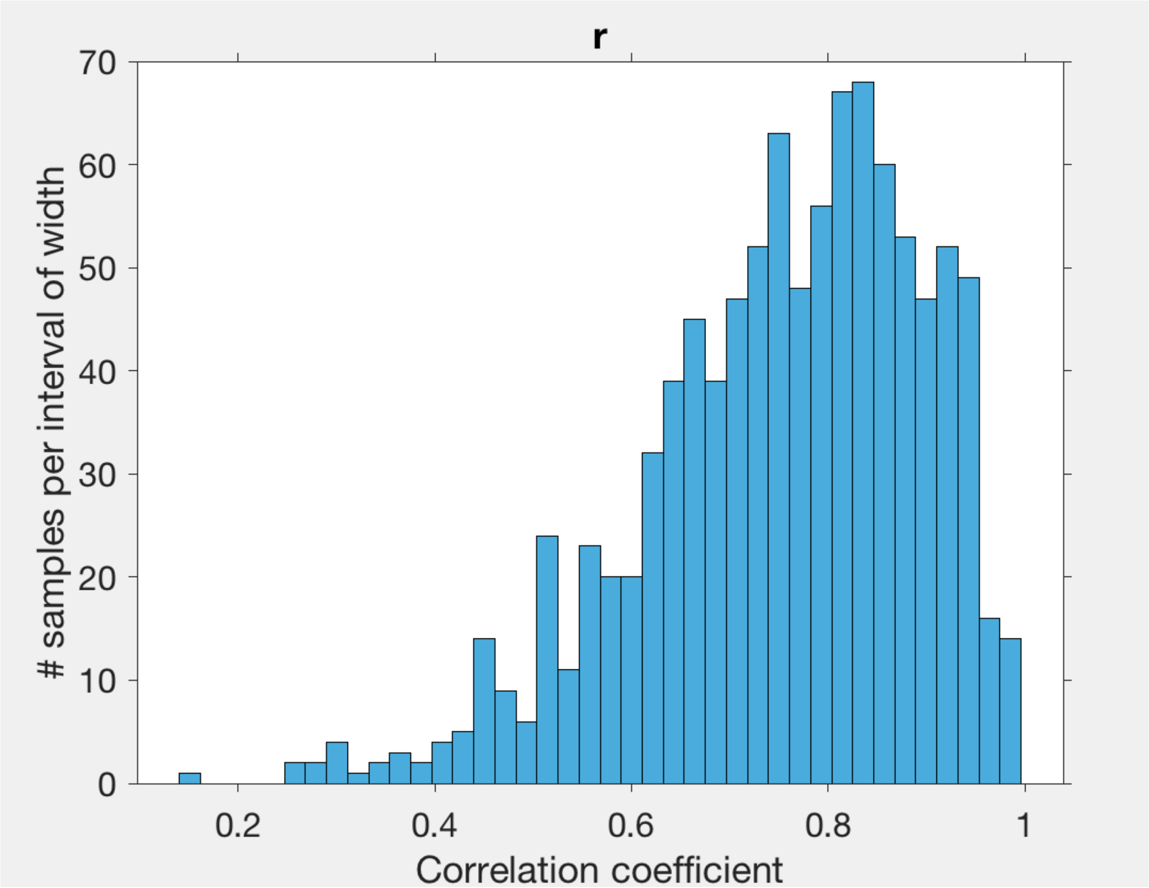
D. Compute the correlation coefficient r for each bootstrap sample in a 1,000-element vector. Plot a histogram of the sampled values of r and compare it to the histogram on p. 120 of Diaconis and Efron 1983. Are you getting similar results?

```
% HW 3 - C & D

N = 1000;
rlist = zeros(N,1);
plist = rlist;
for i=1:N
    [X,Y] = samplebootstrap(x,y);
    [R,P,RL,RU] = corrcoef(X,Y);
    rlist(i) = R(1,2);
    plist(i) = P(1,2);
end

% plot r
figure(2);
histogram(rlist,40);
title("r");
```

```
set(gca,'FontSize',16,'TickDir','out');
```



E.

Compute a 95% confidence interval for r by sorting the vector of sampled values and taking the values bounding the lowest and highest 2.5% of the sampled values.

```
% HW 3 - E

close all;
CI = 0.95;

% 2.5 % * 1000 = 25, so we will have the 26 to 975 values

new_r = sort(rlist);
a = round(N * (1-CI)/2) ;
b = N - a;
new_r = new_r(a + 1:b);

fprintf("The 0.95 confidence interval is (%.3f,
%.3f)\n",new_r(1), new_r(end));
```

The output is:

```
The 0.95 confidence interval is (0.447, 0.961)
```

F. Repeat the bootstrap sampling process for 1,000 samples. The histogram of the sampled values of r and the computed 95% confidence interval will be slightly different every time you run the sampling. Suppose that you wanted to have enough bootstrap samples so that in every run you get the same first two significant digits in the 95% confidence interval (e.g., 0.23 to 0.87). Increase the number of bootstrap samples and find the number you need to get the same first two significant digits when sampling is repeated a few times (say, 5). Report your final 95% confidence interval for r.

```
% HW 3 - F

n_check = 5;
```

```
CI = 0.95;
N = 1000;        % initial N = 1000

syms RL_list RU_list;

while  1
    RL_list = zeros(n_check,1);
    RU_list = RL_list;

    for c = 1:n_check
        rlist = zeros (N,1);

        for i=1:N
            [X,Y] = samplebootstrap(x,y);
            [R,P,RL,RU] = corrcoef(X,Y);
            rlist(i) = R(1,2);
        end

        new_r = sort(rlist);
        a = round(N * (1-CI)/2) ;
        b = N - a;
        new_r = new_r(a + 1:b);

        RL_list(c) = round(new_r(1),2);
        RU_list(c) = round(new_r(end),2);
    end

    if  max(RL_list) == min(RL_list) && max(RU_list) == min(RU_list)
        % make sure they give same two digits and jump out of the while
        break;
    else
        N = N + 1000;
    end
```

```
end

fprintf("The total sampling number to get the 2 digits same
for repeating %g times: %g\n",n_check, N);
fprintf("The 0.95 confidence interval is (%g, %g)\n",
RL_list(1), RU_list(1));
```

The output is:

```
The total sampling number to get the 2 digits same for
repeating 5 times: 20000
The 0.95 confidence interval is (0.43, 0.96)
```