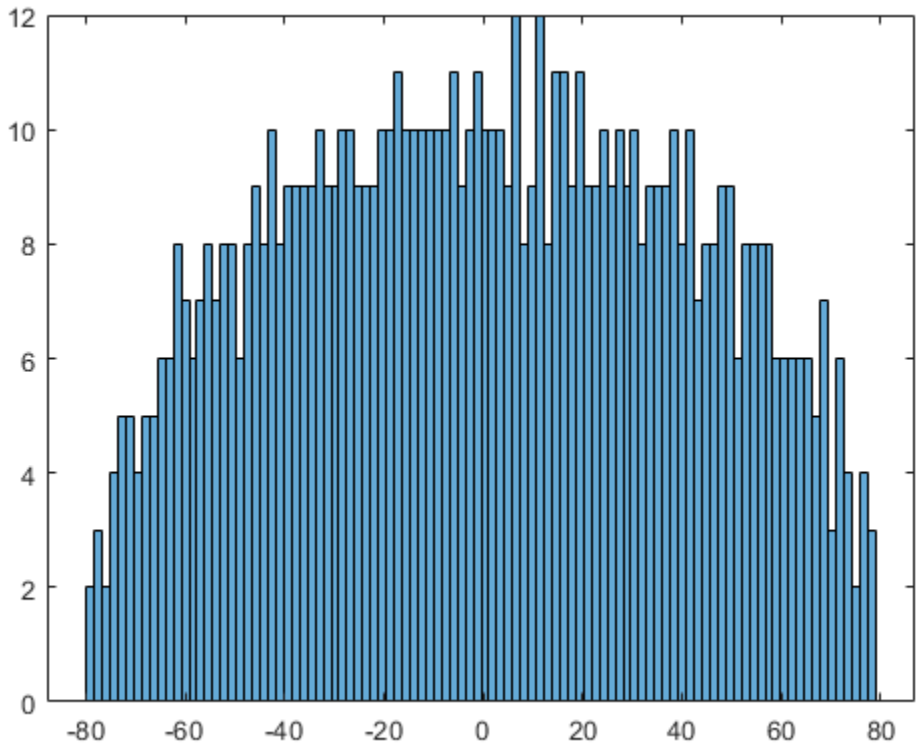
Table of Contents

Exercise C1

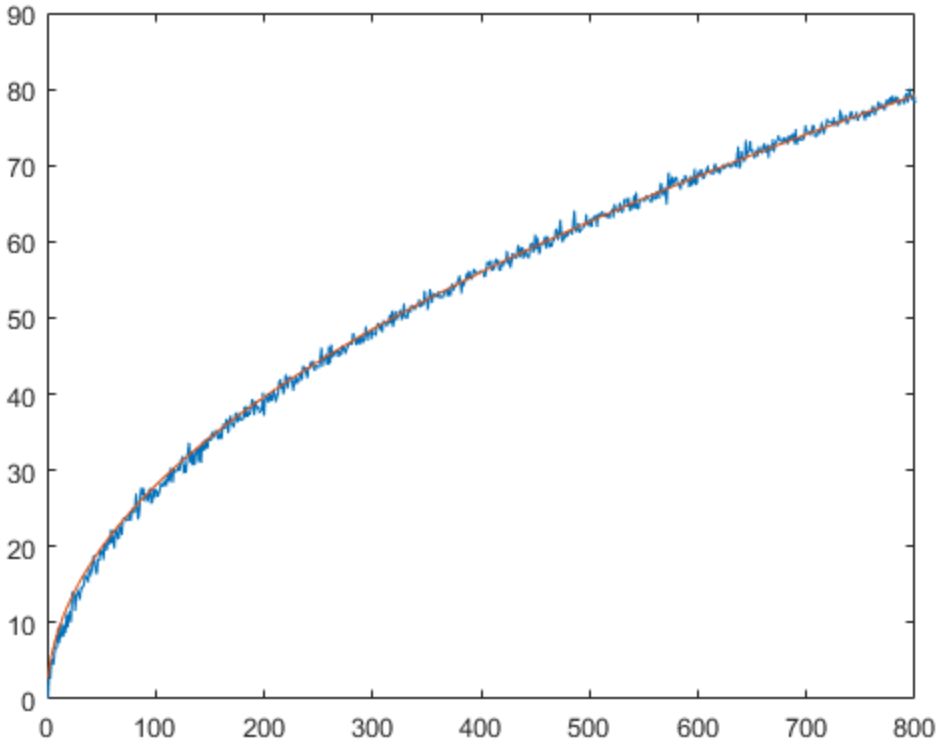
```
%Obtain Eigenvalues
EigenValues = RandSpec(800);
%Plot them on a histrogram with 100 bins
hold off;
histogram(EigenValues, 100)
```



Exercise C2

```
%Define an X range
X = 1:800;
%Obtain values from MaxEval
Y = MaxEval(800);
```

```
%Plot the values of Y(i) and the values of 2.8sqrt(i) on the same plot
plot(X,Y)
hold on;
plot(X,2.8.*X.^(.5))
hold off;
```



Exercise C3

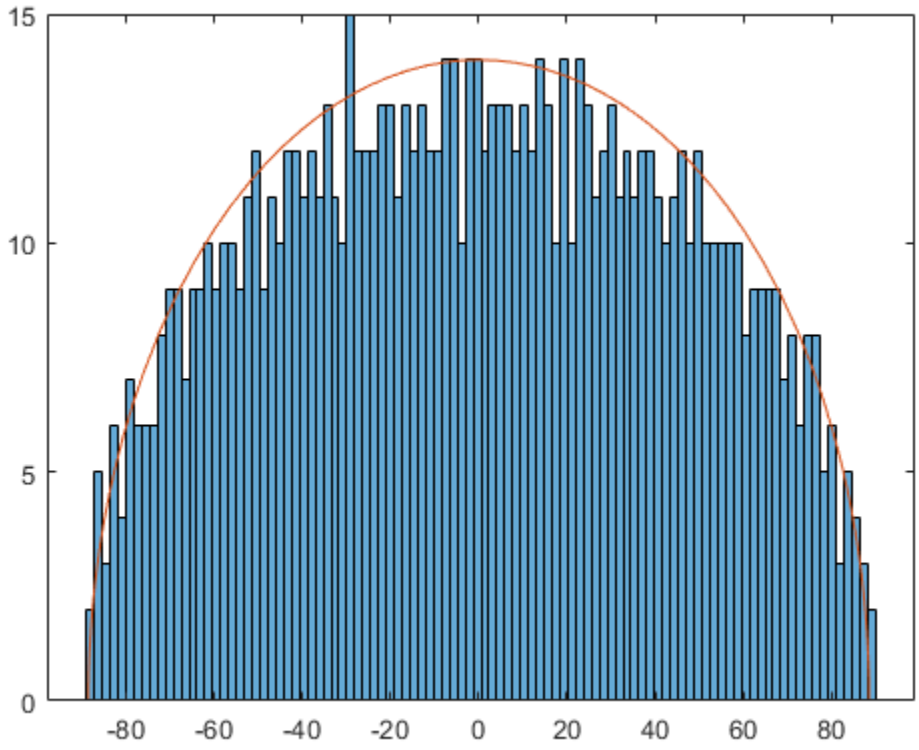
```
%Generate Eigenvalues of a 1000x1000 matrix
EigenValues = RandSpec(1000);
%Plot it's histogram, as in C1
histogram(EigenValues, 100)

%The highest bar appears to be at 14. Initialize a and b
b = 14;
a = 2.8*1000^.5;
%Plot the upper half of the ellipse (x/a)^2 + (y/b)^2 = 1 on the same
%graph, i.e. plotting y = b*sqrt(1 - (x/a)^2) for x in [-a, a]
hold on;
fplot(@(x) b*(1-(x/a)^2)^.5,[-a,a])
%The graph looks better approximated by b=13 but aside from that it
%aligns very closely.
%The final figure for this part was not appearing so has been
%included at the end of the document.
snapnow
```

Function Definitions

```
%Define RandSpec
function e = RandSpec(n)
    %Generate M, an nxn matrix of Gaussian variables
    M = randn(n);
    %Symmetrise it to produce S
    S = M + M.';
    %Output the eigenvalues of S
    e = eig(S);
end

%Define MaxEval
function e = MaxEval(n)
    %Preallocate e's size for reduced memory usage
    e = zeros(n,1);
    %Iterate across i in {1,...,n}
    for i = 1:n
        %Generate Eigenvalues from an ixi matrix
        EigenValues = RandSpec(i);
        %Store the maximum element of EigenValues in e(i)
        e(i) = max(EigenValues);
    end
end
```



Published with MATLAB® R2019b