ENSC 180 - Assignment 5

Shengcong Zhou
Lab group 9
sza111@sfu.ca

1.
Script

```
A=input('Enter a 3x3 matrix\n');
[rows,cols]=size(A);
if rows~=3 || cols~=3
    error('The size of matrix is not 3x3');
end

a=-1;
b=A(1,1)+A(2,2)+A(3,3);
c=A(1,3)*A(3,1)+A(2,3)*A(3,2)+A(1,2)*A(2,1)-A(1,1)*A(3,3)-A(1,1)*A(2,2)-A(2,2)*A(3,3);
d=A(1,1)*A(2,2)*A(3,3)+A(1,2)*A(2,3)*A(3,1)+A(1,3)*A(2,1)*A(3,2)-A(1,3)*A(2,2)*A(3,1)-
A(1,1)*A(2,3)*A(3,2)-A(1,2)*A(2,1)*A(3,3);
P=[a b c d];
eigVal=roots(P);

disp('eigenvalues'); disp(eigVal');

eigVec=zeros(3);
a=A(1,1);e=A(2,2);b=A(1,2);c=A(1,3);f=A(2,3);
for j=1:3
   l=eigVal(j);
   C=[a-l,b;b,e-l];
   B=[-c;-f];
   eigVec(:,j)=[C\B;1];
end

disp('eigenvectors'); disp(eigVec);
```

Output using A=[4 3 1;3 7 -1; 1 -1 9]

```
Enter a 3x3 matrix
[4 3 1;3 7 -1; 1 -1 9]
eigenvalues
    9.4399    8.6808    1.8793

eigenvectors
   -0.1311    1.1647   -4.3670
   -0.5710    1.4840    2.7537
    1.0000    1.0000    1.0000
```

```
%Comparing to the result of [V,D]=eig(A)
V =
   -0.8304    0.5455   -0.1131
    0.5237    0.6950   -0.4927
    0.1902    0.4684    0.8628
```

```
D =

    1.8793         0         0
         0    8.6808         0
         0         0    9.4399

%Eigenvalues obtained using built-in function eigenvalues are exactly the same.
%Eigenvectors are not normalized with eig(A), but they are the same if normalized.
```

2.
(a)
Script

```
A=[4 3 1;3 7 -1;1 -1 9];
M=[1 0 0;0 2 0;0 0 3];

syms w2
determinant=det(A-w2*M);
coefficient=double(fliplr(coeffs(determinant)));
eigVal=sqrt(roots(coefficient));
eigVal=[eigVal;-(eigVal)]; %accounting the negative values
disp('eigenvalues'); disp(eigVal');

eigVec=zeros(3);
a=A(1,1);e=A(2,2);b=A(1,2);c=A(1,3);f=A(2,3);
for i=1:3 %use only the positive eigenvalues since +- makes no difference after getting squared
    w_2=eigVal(i)^2;
    C=[a-w_2,b;b,e-2*w_2];
    B=[-c;-f];
    eigVec(:,i)=[C\B;1];
end
disp('eigenvectors'); disp(eigVec);
```

Output

```
eigenvalues
    2.4280    1.8099    1.1529   -2.4280   -1.8099   -1.1529

eigenvectors
   22.6805    0.3976   -2.8281
   13.9949   -0.4293    2.1845
    1.0000    1.0000    1.0000
```

(b)
Script

```
M=magic(6);

SumRows=sum(M,2)
SumCols=sum(M)
```

```
SumDiagonals=[trace(M),trace(fliplr(M))] %[M(1,1)toM(6,6) M(1,6)toM(6,1)]
```

Output

```
SumRows =

   111
   111
   111
   111
   111
   111


SumCols =

   111   111   111   111   111   111


SumDiagonal =

   111   111
```

```
%All the sums are equally 111
```

(c)

Script

```
A=magic(4);
B=[A 2*A;A^2 A+2];

SumRows=sum(B,2)
SumCols=sum(B)
SumDiagonals=[trace(B),trace(fliplr(B))]
```

Output

```
SumRows =

         102
         102
         102
         102
        1198
        1198
        1198
        1198


SumCols =

        1190        1190        1190        1190         110         110         110         110
```

SumDiagonals =

```
    76    1224
```

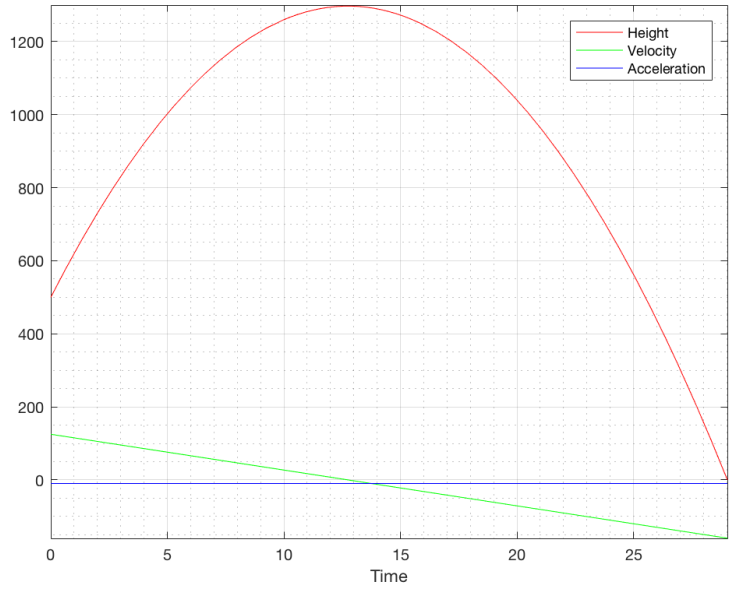
%Each rows either adds up to 102 or 1198. Each Columns either adds up to 1190 or 110 Diagonal top right to bottom left adds up to76 and top left to bottom right adds up tp 1124

3.
Script

```
syms t positive %allow only positive numbers for time
h=-4.9.*t.^2+125*t+500;
v=-9.8*t+125; %1st derivative of h
a=-9.8; %2nd derivative of h

land=double(solve(h,t)); %land: landing time
fprintf('Landing time: %0.2fs\n',land);

fplot(h,'r',[0,land])
hold on
fplot(v,'g',[0,land])
fplot(a,'b',[0,land])
hold off
legend('Height','Velocity','Acceleration')
xlabel('Time')
grid on
grid minor
```

Output

```
Landing time: 29.03s
```

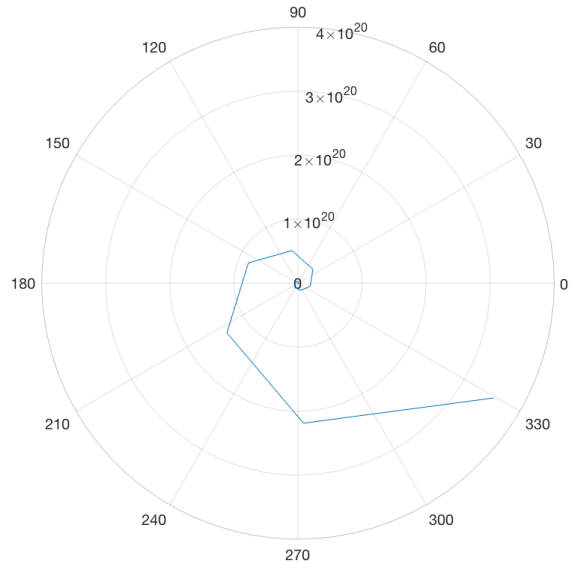


4.
Script

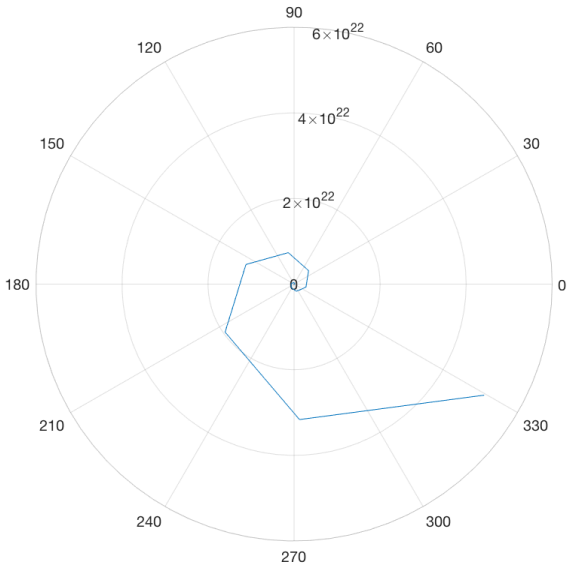
```
function fiboseq
%prompt
NUM1=input('enter the first number in the sequence\n');
NUM2=input('enter the second number in the sequence\n');
index=input('enter the total number of elements in the sequence\n');

if NUM2==0
    error('The second number cannot be 0');
elseif index<=3
    error('Please allow more than 3 numbers to be in the sequence');
end

f=zeros(1,index); %initialize sequence array
%assign the first two numbers
f(1)=NUM1;
f(2)=NUM2;

%calculate the sequence from the 3rd number
for i=3:index
    f(i)=f(i-1)+f(i-2);
end

x=1:index; %element number array
polarplot(x,f(x))

fprintf('Function paused\nPress any key to continue to ratios\n');
pause

g=zeros(1,index-1); %initialize ratio array

%calculate the ratio
for i=1:index-1
    g(i)=f(i+1)/f(i);
end

y=1:index-1; %element number array
plot(y,g)
grid on
grid minor
ylabel(‘ratio')
```

Output using [1 1 100], [0 234 100], [9 9 100]
Fourth graph shows the ratio of adjacent numbers

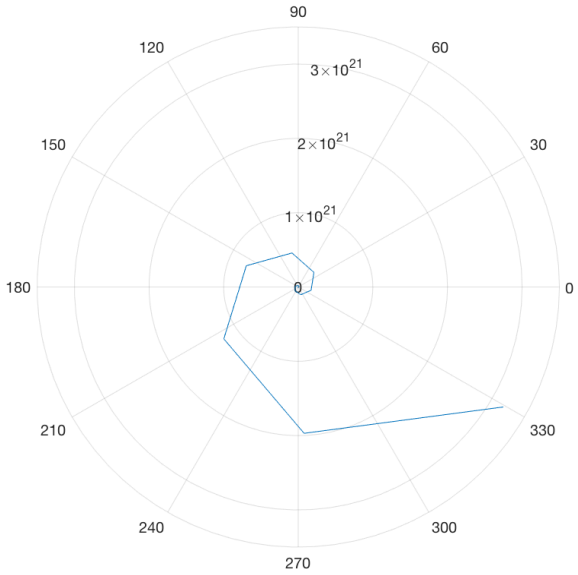
[1 1 100]



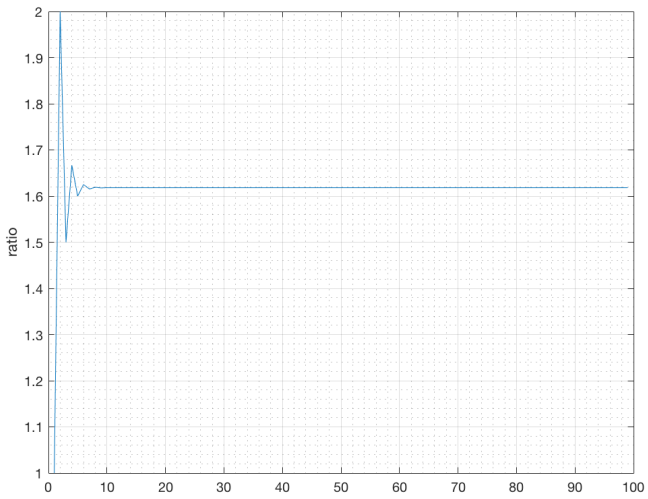
[0 234 100]



[9 9 100]



Ratio



As the sequence grows, the ratio of adjacent numbers tends to reach about 1.618