Mehak Gupta
301311972

COMPUTING ASSIGNMENT 5

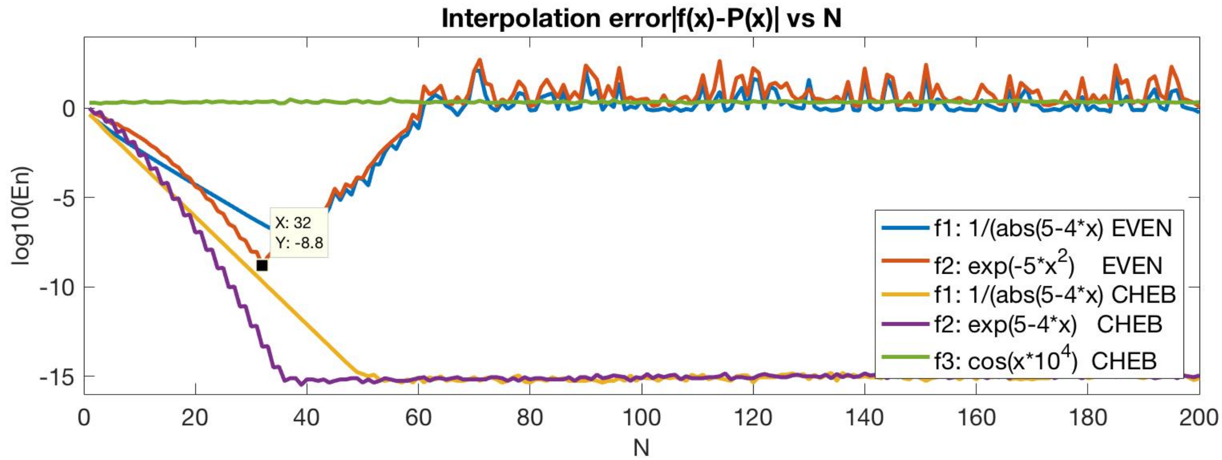

FIG 1

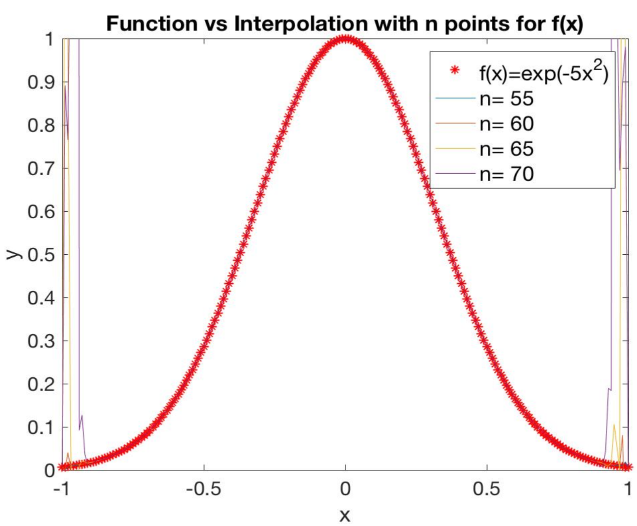

FIG 2

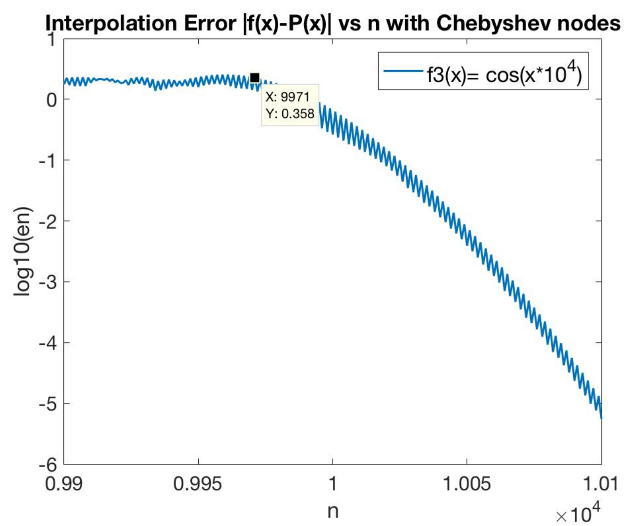

FIG 3

With equal spacing, using n+1 nodes on the interval [-1, 1], Fig. 1 computes the error of interpolating polynomial versus the increase of n. For the function f1(x)-even , the error drops to 10^-8 as n grows but around n = 35 approx, the error begins to grow. Almost the same trend can be seen in f2(x)-even , where the error decreases to 10^-8 and after the point n = 32 approx, the error grows. The accuracy of polynomial interpolation depends on the distribution of the points and the derivative of function. Both functions shows that the polynomial interpolation at equally-spaced nodes is inaccurate and not robust. The reason behind the inaccuracy is the oscillation of the interpolating polynomial near the endpoints of the interval as shown in fig 2. Both the f1 and f2 (even) show this behavior which is called the Runge's phenomenon.

When plotting the same functions with Chebyshev nodes, we observed that the error decreases as n gets larger and after reaching a fixed point, it goes flat which shows high accuracy and robustness (Fig.1).Therefore, the function does not show Runge's phenomenon. However, the interpolation with Chebyshev nodes does not always turns out to be efficient as seen in the case of f3(x). f3(x) = cos(x*10^4) is a function where the y values have large changes over small intervals of x and thus the error does not drop rapidly with small n.

To find the smallest value of n to make the error en <= 10^-5, we observe that the error begins to drop at n= 9971 approx and ultimately reaches en<= 10^-5 at n= 10096 approx as shown in fig (3).

CODE

```
%GIVEN CODE NOT INCLUDED
%
%
function assign5()
clear;
format long;
T_Size = 15;
n = 200; %n>0
step = 0.01;
grid = [-1:step:1]';
f1x = @(x) 1/abs(5 - 4*x);
f2x = @(x) exp(-5*x^2);
f3x= @(x) cos(x*10^4);
f1Error = [];
f2Error = [];
f3Error = [];
f1_EC = [];
f2_EC = [];
f3_EC = [];
for i=1:n
%even data points for interpolation
evenip = getEvenip(i);
barycentricW = baryweights(evenip);
f1yip = ev(evenip,f1x);
f1y = baryinterp(evenip,barycentricW,f1yip,grid);
f2yip = ev(evenip,f2x);
f2y = baryinterp(evenip,barycentricW,f2yip,grid);
%cheb data points for interpolation
CS_Nodes = getCS_Nodes(i);
chebyW = getChebyShevWeights(i);
f1ychebip = ev(CS_Nodes,f1x);
f1yCheb = baryinterp(CS_Nodes,chebyW,f1ychebip,grid);
f2ychebip = ev(CS_Nodes,f2x);
f2yCheb = baryinterp(CS_Nodes,chebyW,f2ychebip,grid);
f3ychebip = ev(CS_Nodes,f3x);
f3yCheb = baryinterp(CS_Nodes,chebyW,f3ychebip,grid);
%Max Errors
error = getMaxError(f1y,f1x,grid);
f1Error = [f1Error getMaxError(f1y,f1x,grid)];
f2Error = [f2Error getMaxError(f2y,f2x,grid)];
f1_EC = [f1_EC getMaxError(f1yCheb,f1x,grid)];
f2_EC = [f2_EC getMaxError(f2yCheb,f2x,grid)];
f3_EC = [f3_EC getMaxError(f3yCheb,f3x,grid)];
end
figure;
plot(1:n,log10(f1Error),'DisplayName', 'f1: 1/(abs(5-4*x) EVEN', 'LineWidth',2.5);
hold on
plot(1:n,log10(f2Error),'DisplayName', 'f2: exp(-5*x^2)    EVEN', 'LineWidth',2.5);
hold on
plot(1:n,log10(f1_EC),'DisplayName', 'f1: 1/(abs(5-4*x) CHEB', 'LineWidth',2.5);
hold on
plot(1:n,log10(f2_EC),'DisplayName', 'f2: exp(5-4*x)   CHEB', 'LineWidth',2.5);
hold on
plot(1:n,log10(f3_EC),'DisplayName', 'f3: cos(x*10^4)  CHEB', 'LineWidth',2.5);
hold off
title('Interpolation error|f(x)-P(x)| vs N','fontsize',20);
xt = get(gca, 'XTick');
set(gca, 'FontSize', T_Size)
xlabel('N','fontsize',15);
ylabel('log10(En)','fontsize',15);
lgd = legend('Location','southeast');
lgd.FontSize = 15;
ylim([-16 4]);
legend('show');
figure; %next figure shows runge's phenomenon
maxn = 70;
st = 5;
plot(grid,ev(grid,f2x),'r*','DisplayName', 'f(x)=exp(-5x^2)');
hold on
```

```
for i=55:st:maxn
evenip = getEvenip(i);
barycentricW = baryweights(evenip);
f2yip = ev(evenip,f2x);
f2y = baryinterp(evenip,barycentricW,f2yip,grid);
plot(grid,f2y,'DisplayName', ['n= ' num2str(i)]');
hold on
end
hold off
title(['Function vs Interpolation with n points for f(x)'] ,'fontsize',18);
xlabel('x','fontsize',15);
ylabel('y','fontsize',15);
xt = get(gca, 'XTick');
set(gca, 'FontSize', T_Size)
ylim([0 1]);
lgd = legend('Location','northeast');
lgd.FontSize = 16;
legend('show');
figure; %next figure displays error going down for f3x
f3_EC = [];
nstart = 9900;
nend = 10100;
found = false;
for i=nstart:nend
CS_Nodes = getCS_Nodes(i);
chebyW = getChebyShevWeights(i);
f3ychebip = ev(CS_Nodes,f3x);
f3yCheb = baryinterp(CS_Nodes,chebyW,f3ychebip,grid)
error = getMaxError(f3yCheb,f3x,grid);
f3_EC = [f3_EC error];
if error <= 10^-5 && found == false
disp(['en <= 10^-5 for f3(x) at n=' num2str(i)])
found = true;
end
end
plot(nstart:nend,log10(f3_EC),'DisplayName', 'f3(x)= cos(x*10^4)', 'LineWidth',1.5);
title(['Interpolation Error |f(x)-P(x)| vs n with Chebyshev nodes'] ,'fontsize',18);
xlabel('n','fontsize',15);
ylabel('log10(en)','fontsize',15);
xt = get(gca, 'XTick');
set(gca, 'FontSize', T_Size)
lgd = legend('Location','northeast');
lgd.FontSize = 16;
legend('show');
hold off
end
function y = ev(points,fun)
yValues = [];
for i=1:length(points)
yValues = [yValues (fun(points(i)))];
end
y = yValues';
return
end
function y = getEvenip(n) %domain: [1,-1]
ip = [];
for i=0:n
ip = [ip (-1 + (2*i)/n)];
end
y = ip';
return
end
function y = getCS_Nodes(n) %domain: [1,-1]
CS_Nodes = [];
for i=0:n
CS_Nodes = [CS_Nodes (cos((i*pi)/n))];
end
y = CS_Nodes';
return
end
function y = getChebyShevWeights(n)
weights = [];
```

```
weights = [weights (1/2)];
for i=1:n-1
weights = [weights ((-1)^i)];
end
weights = [weights ((1/2)*((-1)^n))];
y = weights';
return
end
function y = getMaxError(pyvalues,fun,grid)
maxError = 0;
for i=1:length(grid)
error = abs(pyvalues(i) - fun(grid(i)));
if error > maxError
maxError = error;
end
end
y = maxError;
return
end
```