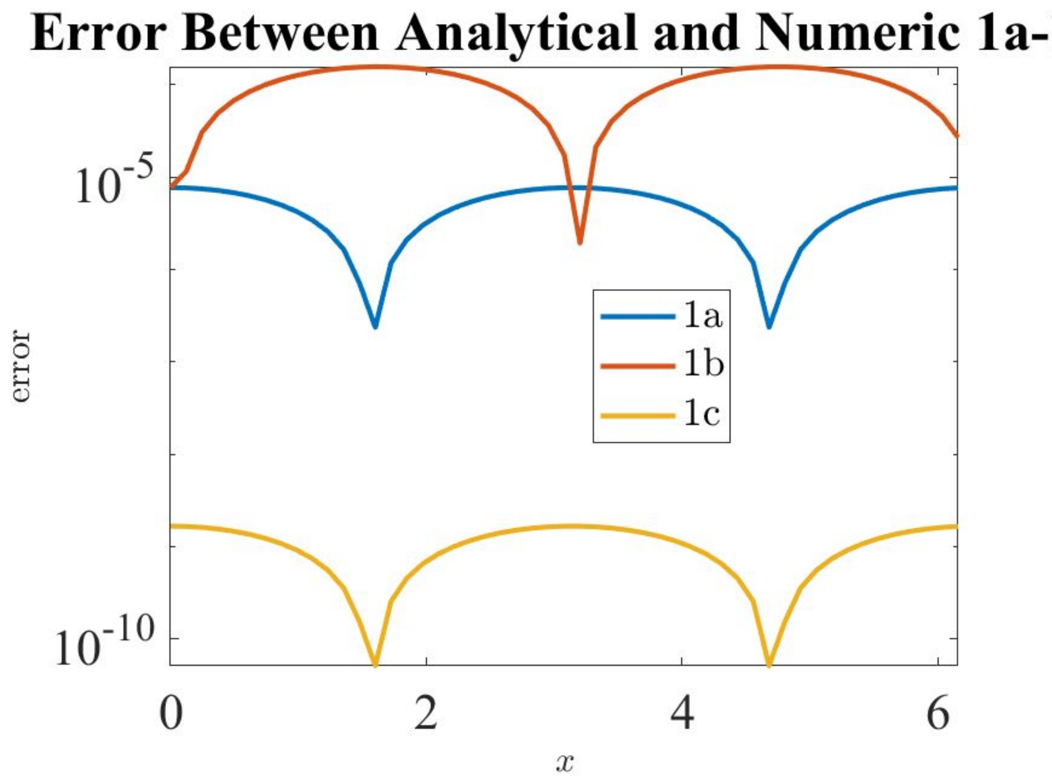
Christopher Endres
AE410
3/4/2020

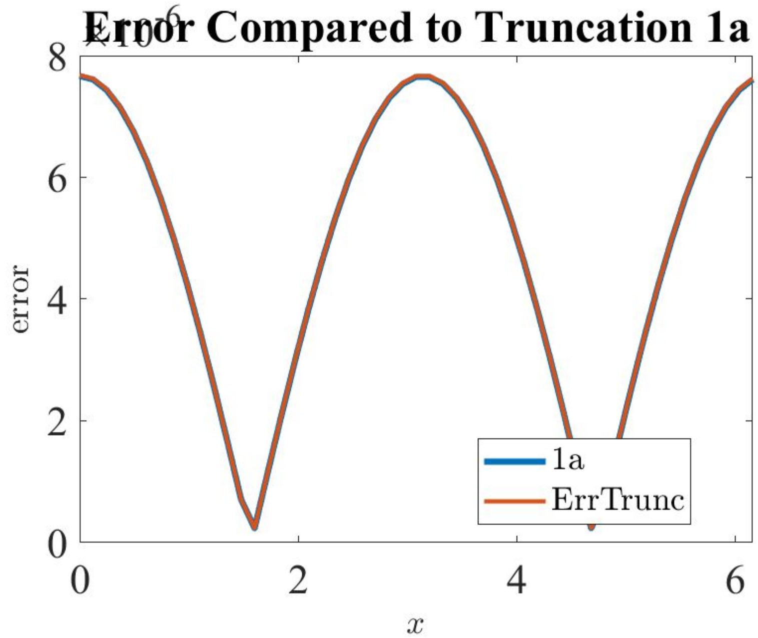
Homework 3

1. See code
2. See plots:



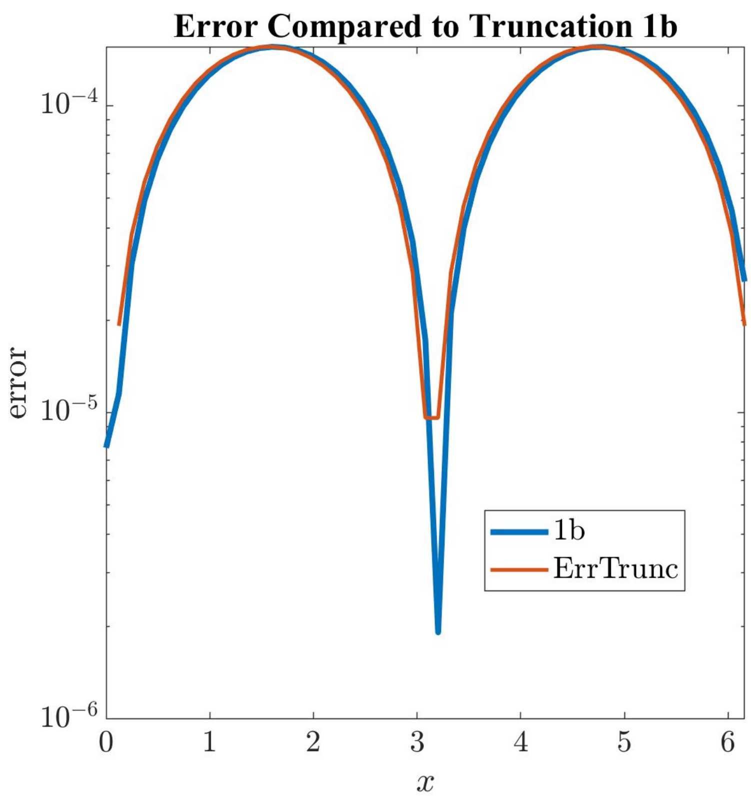
a.

b. As one can see the error between the first and second scheme is order of magnitudes higher than the error of the scheme in HW2 1c.

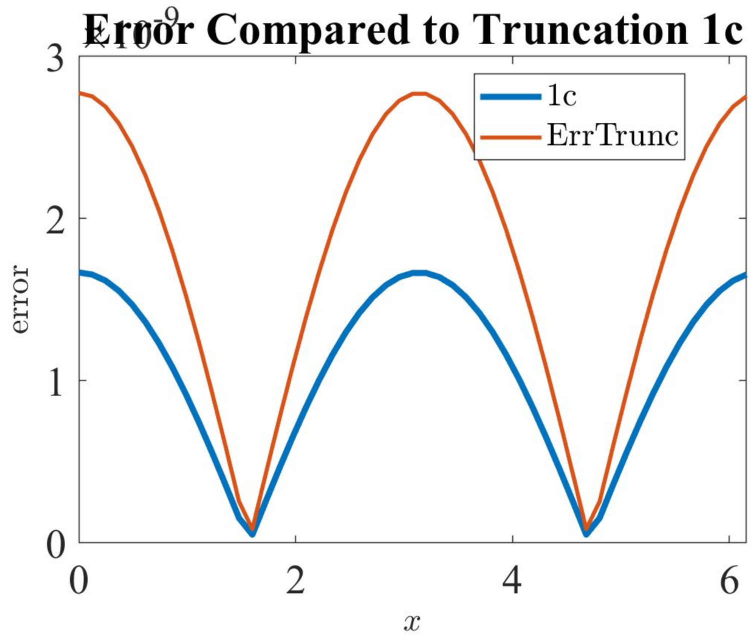


Error Compared to Truncation 1a
$\times 10^{-6}$
8
6
4
2
0
error
0
2
4
6
x
1a
ErrTrunc

c.

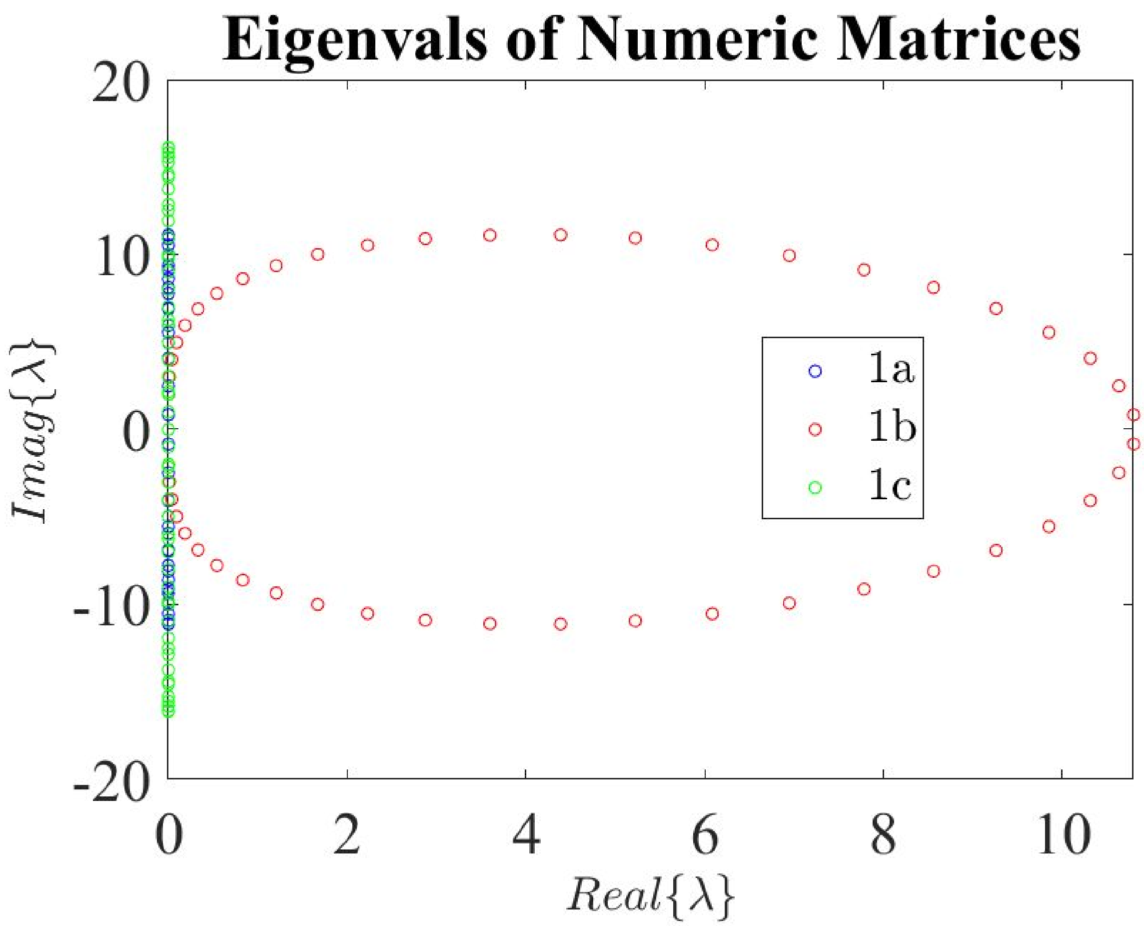


Error Compared to Truncation 1b
10^{-4}
10^{-5}
10^{-6}
error
0
1
2
3
4
5
6
x
1b
ErrTrunc



d. The errors between the truncation from 1a-1c is also relatively small. There seems to be certain values of x where error is minimal

3.



a. It can be seen that the first and third schemes have very small or no real parts, where the second scheme has both real and imaginary parts

```
%% Prob 1
clear all;close all; clc
% default plot attributes
set(0,'defaultaxesfontname','times');
set(0,'defaultaxesfontsize',20);
figID = 1;
% Ny = 11;
% L = 2*pi;
% y = linspace(0,L,Ny).';
% dy = y(2)-y(1)

Nx = 51;
L  = 2*pi;
dx = L/Nx;
x  = dx*linspace(0,Nx-1,Nx).';

% construct matrix operator
D1 = Central4P(Nx,dx);% Periodic
D2 = OneBP(Nx,dx);% Periodic
D3 = OneCP(Nx,dx);
D1sparse = sparse(D1);
D2sparse = sparse(D2);
D3sparse = sparse(D3);

%% Prob 2
% equation to differentiate
f = sin(x);

syms a
f1 =  sin(a);
derivative = diff(f1);
DR = subs(derivative,a,x);

% derv(end+1)=derv(1);
f1 = D1sparse * f;
f2 = D2sparse * f;
f3 = D3sparse * f;

% plot
figure(figID); figID = figID + 1; clf;
plot(x,f1,'linewidth',2)
hold on;
plot(x,f2,'linewidth',2)
plot(x,f3,'linewidth',2)
plot(x,DR,'ko')
h = legend( '1a','1b','1c','Analaytical' );
set(h, 'location', 'best', 'interpreter', 'latex', 'fontsize', 16)
xlabel( '$x$', 'interpreter', 'latex', 'fontsize', 16)
ylabel( '$\frac{df}{dx}$', 'interpreter', 'latex', 'fontsize', 14)
title('Numerical Approximation 1a-1c ')
print( '-dpng', 'picture2', '-r150' )
```

```
%%
figure(figID); figID = figID + 1; clf;
plot(x,abs(DR-f1),'linewidth',2)
hold on;
plot(x,abs(DR-f2),'linewidth',2)
plot(x,abs(DR-f3),'linewidth',2)
h = legend( '1a','1b','1c' );
set( h, 'location', 'best', 'interpreter', 'latex', 'fontsize', 16)
xlabel( '$x$', 'interpreter', 'latex', 'fontsize', 14)
ylabel( 'error', 'interpreter', 'latex', 'fontsize', 14)
set(gca, 'YScale', 'log')
title('Error Between Analytical and Numeric 1a-1c')
print( '-dpng', 'picture3', '-r150' )
%% compare to truncaiton 1a
truncation_error_term(1,:) = abs(((-dx^4)/30)*(cos(x)));
figure(figID); figID = figID + 1; clf;
plot(x,abs(DR-f1),'linewidth',3), hold on;
plot(x,truncation_error_term(1,:),'linewidth',2)
h = legend( '1a','ErrTrunc' );
set( h, 'location', 'best', 'interpreter', 'latex', 'fontsize', 16)
xlabel( '$x$', 'interpreter', 'latex', 'fontsize', 16)
ylabel( 'error', 'interpreter', 'latex', 'fontsize', 16)
title('Error Compared to Truncation 1a')
print( '-dpng', 'picture4', '-r150' )
%% compare to truncaiton 1b
truncation_error_term(2,:) = abs(((dx^3)/12)*(sin(x)));
figure(figID); figID = figID + 1; clf;
plot(x,abs(DR-f2),'linewidth',3), hold on;
plot(x,truncation_error_term(2,:),'linewidth',2)
h = legend( '1b','ErrTrunc' );
set( h, 'location', 'best', 'interpreter', 'latex', 'fontsize', 16)
xlabel( '$x$', 'interpreter', 'latex', 'fontsize', 16)
ylabel( 'error', 'interpreter', 'latex', 'fontsize', 16)
set(gca, 'TickLabelInterpreter','latex', 'fontsize', 16 )
set(gcf,'PaperPositionMode', 'manual')
set(gcf,'Color', [1 1 1])
set(gca,'Color', [1 1 1])
set(gcf,'PaperUnits', 'centimeters')
set(gcf,'PaperSize', [15 15])
set(gcf,'Units', 'centimeters' )
set(gcf,'Position', [0 0 15 15])
set(gcf,'PaperPosition', [0 0 15 15])
set(gca, 'YScale', 'log')
title('Error Compared to Truncation 1b')
print( '-dpng', 'picture5', '-r200' )
%% compare to truncaiton 1c
truncation_error_term(3,:) = abs(((4*dx^6)/factorial(7))*(-cos(x)));
figure(figID); figID = figID + 1; clf;
plot(x,abs(DR-f3),'linewidth',3), hold on;
plot(x,truncation_error_term(3,:),'linewidth',2)
h = legend( '1c','ErrTrunc' );
```

```
set( h, 'location', 'best', 'interpreter', 'latex', 'fontsize', 16)
xlabel( '$x$', 'interpreter', 'latex', 'fontsize', 16)
ylabel( 'error', 'interpreter', 'latex', 'fontsize', 16)
title('Error Compared to Truncation 1c')
print( '-dpng', 'picture6', '-r200' )
%%
% plot the eigenvalues of D
[V1,De1,W1] = eig(D1);
[V2,De2,W2] = eig(D2);
[V3,De3,W3] = eig(D3);

figure(figID); figID = figID + 1; clf;
plot(real(diag(De1)),imag(diag(De1)),'bo','markersize',4), hold on;
plot(real(diag(De2)),imag(diag(De2)),'ro','markersize',4)
plot(real(diag(De3)),imag(diag(De3)),'go','markersize',4)
h = legend( '1a','1b','1c' );
set( h, 'location', 'best', 'interpreter', 'latex', 'fontsize', 16)
xlabel( '$Real\{\lambda\}$', 'interpreter', 'latex', 'fontsize', 16)
ylabel( '$Imag\{\lambda\}$', 'interpreter', 'latex', 'fontsize', 16)
title('Eigenvals of Numeric Matrices')
print( '-dpng', 'picture7', '-r150' )


%%
function C = Central4P(Nx,dx)
%
% [C] = Central4P(Nx,dx)
%
% Return the matrix for the 1/12,?2/3,0,2/3,?1/12 Central fourth order FD scheme on a↙
grid
% of Nx points with equal spacing dx.
%
% INPUTS:
%
% OUTPUTS:
% C = B
%
a=zeros([5 1]);
a(1)=1/12;
a(2)=-2/3;
a(3)=0;
a(4)=2/3;
a(5)=-1/12;

vect=[a(3) a(4) a(5) zeros([1 Nx-5]) a(1) a(2) ];

M = zeros(Nx,Nx);

for i=1:Nx
    M(i,:) = vect;
    vect = circshift(vect,1);
```

```
end

M = M * (1/dx);

C = M;

return;

end
%%
function C = OneBP(Nx,dx)
%
% [C] = OneBP(Nx,dx)
%
% Return the matrix for the 1/6,?1,1/2,1/3 Biased Stencil third order FD scheme on a grid
% of Nx points with equal spacing dx.
% INPUTS:
%
% OUTPUTS:
% C = B
%
a=zeros([4 1]);
a(1)=1/6;
a(2)=-1;
a(3)=1/2;
a(4)=1/3;

vect=[a(3) a(4) zeros([1 Nx-4]) a(1) a(2)];


M = zeros(Nx,Nx);

for i=1:Nx
    M(i,:) = vect;
    vect = circshift(vect,1);
end

M = M * (1/dx);

C = M;

return;

end
%%
function C = OneCP(Nx,dx)
%
% [C] = OneCP(Nx,dx)
%
%Basically a pade scheme with 5 points and 3 derivatives on the left side
% Return the matrix for the alpha=1/3 Biased Stencil sixth order compact FD scheme on a↙
```

```
grid
% of Nx points with equal spacing dx.
% INPUTS:
%
% OUTPUTS:
% C = B
% We need two matrices
al = 1/3;

a=zeros([5 1]);
a(1)=-(4*al-1)/12;
a(2)=-(al+2)*4/12;
a(3)=0;
a(4)=(al+2)*4/12;
a(5)=(4*al-1)/12;

vect1=[a(3) a(4) a(5) zeros([1 Nx-5]) a(1) a(2) ];

vect2=[1 al zeros([1 Nx-3]) al ];

A = zeros(Nx,Nx);
B = zeros(Nx,Nx);

for i=1:Nx
    B(i,:) = vect1;
    vect1 = circshift(vect1,1);
    A(i,:) = vect2;
    vect2 = circshift(vect2,1);
end
B = B * (1/dx);

M=inv(A)*B;

C = M;

return;

end
%
% function f = func(w)
%     f = sin(w);
% end
```