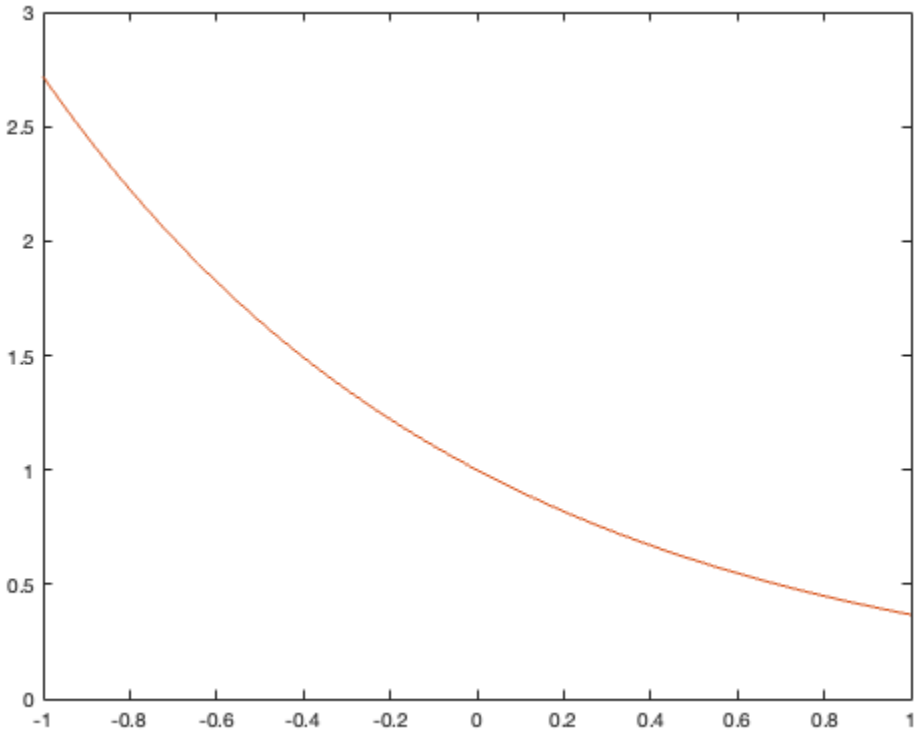
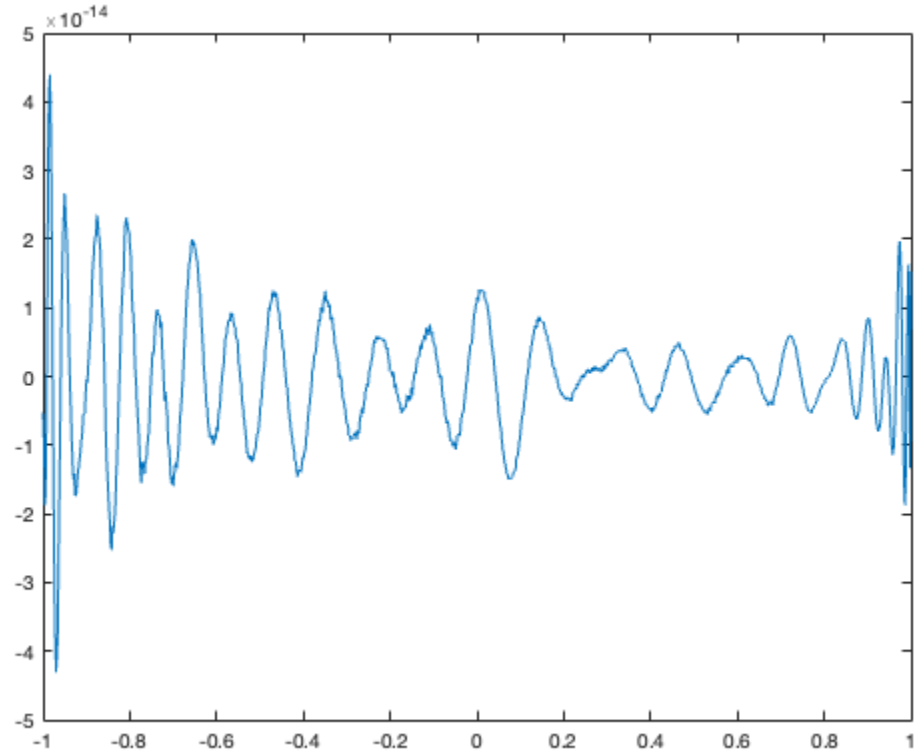
```
clear

% Chebyshev rational approximant parameters
n = 60;
m = 0;

% Choice of function:
f = @(x) exp(-x);

% Number of Gauss-Chebyshev quadrature points
ng = 100;

% Plotting interval is [-b,b]
b = 1;

% Construct Chebyshev rational approximation
[p,q] = my_chebrat(f,n,m,ng);

% Plot results on [-b,b]
nx = 1000;
x_eval = linspace(-b,b,nx)';
r_eval = chebrat_eval(p,q,x_eval); % approximant
y_eval = f(x_eval); % ground truth

% Plot approximation and ground truth
figure(1);clf;plot(x_eval,r_eval);hold on;plot(x_eval,y_eval);

% Plot difference of approximant and ground truth
figure(2);clf;plot(x_eval,r_eval-y_eval);


% Function that calculates coefficients a_0, a_1, ... a_{size} for use in
% Chebyshev approximations.

function a = cheb_coeffs(f, size, ng)
a = zeros(1, 1+size);
a(1) = (1/pi)*gauss_chebyshev(f, ng);
for k = 2:1+size
    prod = @(x) (f(x)*cheb_eval(k-1, x));
    a(k) = (2/pi)*gauss_chebyshev(prod, ng);
end
return
end

% Function that constructs Chebyshev rational approximation
% Inputs:
% - f is function handle for function to approximate
% - n and m are numerator and denominator degrees
% Outputs: p and q are coefficients for numerator and denominator polys


function [p,q] = my_chebrat(f,n,m,ng)
```

```
    % YOUR CODE GOES HERE

    % Define order
    N = n+m;

    % Get 0, ..., N+m Chebyshev coefficients
    a = cheb_coeffs(f, N+m, ng);

    % Construct E matrix.
    E = [eye(n+1);zeros(N-n,n+1)];

    % Construct B matrices.
    B1 = zeros(N+1, m+1);
    for k = 0:N
        for i = 0:m
            if i <= k
                B1(k+1, i+1) = a(k-i+1);
            end
        end
    end

    B2 = zeros(N+1, m+1);
    for k = 0:N
        for i = 0:m
            if i >= k
                B2(k+1, i+1) = a(i-k+1);
            end
        end
    end

    B3 = zeros(N+1, m+1);
    for k = 0:N
        for i = 0:m
            if k >= 1
                B3(k+1, i+1) = a(i+k+1);
            end
        end
    end

    % Construct C matrix.
    C = (1/2).*(B1+B2+B3);
    c = C(:, 1); % Copy first column into c vector.
    C(:, 1) = []; % Remove first column to form C.

    % Construct A and solve.
    A = [E, -C];
    x = A\c;
    p = x(1:n+1);
    q = x(n+2:end);


end
```

```
% Define function that evaluates rational function
% Inputs:
% - p and q are coefficients for numerator and denominator polys
% - x_eval is vector of evaluation points
% Output: r_eval is vector of values at evaluation points

function r_eval = chebrat_eval(p,q,x_eval)

    % YOUR CODE GOES HERE
    % Get vector sizes from input
    n = length(p)-1;
    m = length(q);
    nx = length(x_eval);

    % Build numerator
    num = zeros(nx,1);
    for i=0:n
        num = num + p(i+1)*cheb_eval(i, x_eval);
    end

    % Build denominator
    denom = ones(nx,1);
    for i=1:m
        denom = denom + q(i)*cheb_eval(i, x_eval);
    end

    % Divide
    r_eval = num./denom;

end


% You can use this function to evaluate the n-th Chebyshev polynomial
% on a vector x_eval of evaluation points
% Output: T_eval is vector of values of T_n at evaluation points

function T_eval = cheb_eval(n,x_eval)
    T_eval = cos(n*acos(x_eval));
end
```


3
2.5
2
1.5
1
0.5
0
-1
-0.8
-0.6
-0.4
-0.2
0
0.2
0.4
0.6
0.8
1


x 10^-14
5
4
3
2
1
0
-1
-2
-3
-4
-5
-1
-0.8
-0.6
-0.4
-0.2
0
0.2
0.4
0.6
0.8
1

Published with MATLAB® R2023b