CFH Test

Name: Mohammed Alshuaili

Date : 3/12/2025

Consider the following poles:

$$-1 \pm j, -2 \pm 2j$$

The transfer function is then,

$$\frac{1}{s+1-j} + \frac{1}{s+1+j} + \frac{2}{s+2-2j} + \frac{2}{s+2+2j}$$

$$\frac{2s+2}{s^2+2s+2} + \frac{4s+8}{s^2+4s+8}$$

$$TF = \frac{6s^3+26s^2+48s+32}{s^4+6s^3+18s^2+24s+16}$$

Testing AWE with w=0 and w = 2.5j, we get the correct poles and residues:

$$-1 \pm j, -2 \pm 2j$$

Code

```
clear
clc
num=[6,26,48,32] ; %results from the given exxample
deno = [1 6 18 24 16];
[A,B,C,D] = create_state_space(num,deno);
w = 2.5i;% expnation point
[p,~,r] = AWE_poles(A,B,C,D,w);
 p
```

looking at the transmission line with the following parameters:

$$R = 0.1, l = 400m, L = 2.5\,x10^{-7}, C = 1x10^{-10}, v_{in} = 30\,, f_{max} = 9x10^5$$

2 models:

w	Poles	Residues
AWE_$W(1) = 0$, Y_W(1:10)	-1.5223 + 7.3975i -1.5223 - 7.3975ix10^5	-0.7784 - 4.0332i -0.7784 + 4.0332ix10^5
AWE_$W = 150$, Y_W(11:15)	-0.3748 + 7.7129i -0.3762 - 7.7037i$x10^5$	2.5440 - 0.2574i 2.5410 + 0.2553i$x10^4$

The results are almost the same as the first model

Code:

```
clear
clc
% Generate frequency points
f = linspace(0, 9e5, 100);
w = 2*pi*f;
s = 1i * w;
t = 50e-6;
```

```
first_idx = 1:10;
vo =1./(cosh(400.*(1e-10.*s).^(1/2).*(0.1+2.5e-7.*s).^(1/2)));
[H1a,num,deno] = generate_yp2(real(vo(first_idx)),imag(vo(first_idx)),w(first_idx));
[A,B,C,D] = create_state_space(num,deno);
[~,H1,~,~,p1,m1]=AWE2(A,B,C,D,0,30,t);% to compare it with the results
[p1,np1,r1,m1] = AWE_poles(A,B,C,D,w(first_idx(1)));
%second model -------------------------------------------------------
idx = 11:15; % f = 9.09e4 : 1.2727e+05
H_diff = vo(idx)-H1a(s(idx));
[H2a,num,deno] = generate_yp2(real(H_diff),imag(H_diff),w(idx));
%[H2a,num,deno] = generate_yp2(real(vo(idx)),imag(vo(idx)),w(idx));
[A,B,C,D] = create_state_space(num,deno);
[p2,np2,r2,m2] = AWE_poles(A,B,C,D,150);
test_poles = [p1',p2'];
% remove unstable poles
for i=1:length(test_poles)
if real(test_poles(i))<0
    ptest(i) = test_poles(i);
end
end
moments = m1; % moments from the first model has 1 value and zeros
[hs,r]= generate_hs(ptest,length(ptest),moments);
RMSE_idx = 1:25;
R1 = RMSE(hs(s(RMSE_idx)),vo(RMSE_idx));
R2 = RMSE(H1a(s(RMSE_idx)),vo(RMSE_idx));
plot(f,hs(s),f,vo,f,H1a(s),'ro');
legend('result','exact','first model');
xlabel('f (Hz)')
```

AWE_poles :

```
function [poles,poles_unshifted,residues,moments]= AWE_poles(A, B, C, D, w)
    q = length(B);
    num_moments = 2 * q;
    s0 = 1i * w;
    moments = zeros(1, num_moments);
    [r, c] = size(C);
    if r ~= 1
        C = C';
    end
    for k = 1:num_moments
        moments(k) = (-1)^(k-1)*C*(s0*eye(size(A))- A)^-(k)*B;
    end
    moments(1) = moments(1) + D;  % Include D in the zeroth moment

    approx_order = q;

    % Construct moment matrix and vector for denominator coefficients
    moment_matrix = zeros(approx_order);
    Vector_c = -moments(approx_order+1 : 2*approx_order)';
    for i = 1:approx_order
        moment_matrix(i, :) = moments(i : i + approx_order - 1);
    end

    % Solve for denominator coefficients
    b_matrix = moment_matrix \ Vector_c;
    poles_unshifted = roots([b_matrix; 1]);  % Unshifted poles (s' = s - s0)

    % Compute residues using unshifted poles
    V = zeros(approx_order);
    for i = 1:approx_order
```

```
        for j = 1:approx_order
            V(i, j) = 1 / (poles_unshifted(j))^(i-1);
        end
    end
    A_diag = diag(1 ./ poles_unshifted);
    r_moments = moments(1:approx_order);
    residues = -A_diag \ (V \ r_moments(:));

    % Shift poles to s-plane
    poles = poles_unshifted+s0;
end
```

generate hs

```
function [h_s,residues]= generate_hs(poles,q,moments)
    approx_order =q;

    % Compute residues using given poles and moments
    V = zeros(approx_order);
    for i = 1:approx_order
        for j = 1:approx_order
            V(i, j) = 1 / (poles(j))^(i-1);
        end
    end
    A_diag = diag(1 ./ poles);
    r_moments = moments(1:approx_order);
    residues = -A_diag \ (V \ r_moments(:));

    % Transfer function in s-domain
    h_s = @(s)0;
    for i =1:length(poles)
        h_s = @(s) h_s(s)+residues(i) ./ (s - poles(i));
    end
end
```