W7

Name: Mohammed AL Shuaili

Date: 26/2/2025

Consider the following frequency response for a transmission line.

A graph with blue lines

AI-generated content may be incorrect.

Figure 1 : shows the frequency response of the exact solution.

This is modelled over 100 frequency points using the following parameters:

Generating rational approximation using the first 15 points will result in with :

A graph of a function

AI-generated content may be incorrect.

Figure 2 compares the exact responses to AWE and Y Rational approximation.

The poles are :

Observations:

For any approximation, when using AWE at , the poles are always the same with different sign.

For Example, consider a second model where the exact values are used from (15:25) as in the following code:

[H2,num,deno] = generate\_yp2(real(vo(15:25)),imag(vo(15:25)),w(15:25));

[A,B,C,D] = create\_state\_space(num,deno);

[poles2, moments2] = AWE\_poles(A,B,C,D,w(1));

where the poles are,

This is without evaluating the second approximation as the difference between the exact and the first model at (15:25).

With :

AWE adjusted to be evaluated about

function [h\_impulse, h\_s, y\_step, t] = AWE2(A, B, C, D, w, input, time)

t = linspace(0, time, 250);

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(:));d

% Shift poles to s-plane

poles = poles\_unshifted + s0;

% Impulse response using shifted poles

h\_impulse = zeros(size(t));

for i = 1:approx\_order

h\_impulse = h\_impulse + residues(i) \* exp(poles(i) \* t);

end

% Transfer function in s-domain

h\_s = @(s) sum(residues ./ (s - poles), 1);

% Step response using recursive convolution

y\_step = zeros(size(t));

y = zeros(length(poles), 1);

for n = 2:length(t)

dt = t(n) - t(n-1);

exp\_term = exp(poles \* dt);

for i = 1:length(poles)

y(i) = residues(i) \* (1 - exp\_term(i))/(-poles(i)) \* input + exp\_term(i) \* y(i);

end

y\_step(n) = sum(y);

end

end

in this code the poles are shifted to the origin after calculating moments and residues which gives much accurate results depending on the number of models and the transmission line parameters.

* Complex frequency hoping

Analyse the poles at different frequency and find the dominant poles. Consider 4 hopes.

|  |  |
| --- | --- |
| w | Poles |
| AWE\_, Y\_W(1:15) |  |
| AWE\_, Y\_W(16:25) | -0.4033 + 1.2386i  0.0798 + 1.0928i |
| AWE\_, Y\_W(26:35) | -0.0939 + 2.2961i  -0.0458 + 1.5426i |
| AWE\_, Y\_W(36:45) | -0.0222 + 2.5013i  -0.0035 + 2.2141i |

Chose the dominant poles.

1. **Dominance Condition**:
   * Dominant poles are those with:
     + **Smallest magnitude of real part** (slow decay).
     + **Largest residues** (significant contribution to *H*(*s*)).

One pole from each model is used following the above conditions.

Changing the number of moments as suggested in the paper to obtain 20-30 moments to capture more poles and residues near ​