

Problem 1 - Cut-on Frequencies in Ducts and Pipes

The Matlab code for this problem is listed in Appendix [1](#).

Problem 1a

The lowest cut-on frequency for a rectangular duct with air flow is given by equation,

$$f_{\text{cut-on}} = 0.5 \cdot \frac{c}{L} \quad (1)$$

where c is the speed of sound in air, $343 \frac{\text{m}}{\text{s}}$, and L is the largest side of the rectangular cross-section.

With cross-sectional dimensions of $L_x = 12 \text{ cm}$ and $L_y = 20 \text{ cm}$, the lowest cut-on frequency for this rectangular duct is,

$$f_{\text{cut-on}} = 0.5 \cdot \frac{343 \frac{\text{m}}{\text{s}}}{0.20 \text{ m}} = \mathbf{857.5 \text{ Hz}}$$

Problem 1b

The lowest cut-on frequency for a circular duct with air flow with the same cross-sectional area as the rectangular duct in part (a.) can be calculated using equation,

$$f_{\text{cut-on}} = 0.568 \cdot \frac{c}{d} \quad (2)$$

where c is the speed of sound in air, $343 \frac{\text{m}}{\text{s}}$, and d is diameter of the circular duct.

The cross-sectional area of the rectangular duct is,

$$\text{Area}_{\text{rectangular duct}} = 0.12 \text{ m} \cdot 0.20 \text{ m} = 0.024 \text{ m}^2$$

The corresponding diameter for this area is,

$$\text{diameter} = \sqrt{\frac{0.024 \text{ m}^2}{\pi}} \cdot 2 = 0.17 \text{ m}$$

Using Eq. [2](#), the lowest cut-on frequency for this circular duct with air flow is,

$$f_{\text{cut-on}} = 0.568 \cdot \frac{343 \frac{\text{m}}{\text{s}}}{0.17 \text{ m}} = \mathbf{1,114.5 \text{ Hz}}$$

Problem 1c

The lowest cut-on frequency for this circular duct with water flow can be calculated using Eq. 2,

$$f_{\text{cut-on}} = 0.568 \cdot \frac{1,500 \frac{\text{m}}{\text{s}}}{0.17 \text{ m}} = \mathbf{4,873.9 \text{ Hz}}$$

The lowest cut-on frequency for water is considerable larger than it is for air flow.

Problem 1d

The speed of sound in air is calculated by,

$$c = \sqrt{\gamma \cdot R \cdot T_K} \quad (3)$$

where $\gamma = 1.4$ is the ratio of specific heats, $R = 287 \frac{\text{J}}{\text{kg} \cdot \text{K}}$ is the gas constant, and T_K is the absolute temperature in Kelvin.

Figure 1 illustrates how the lowest cut-on frequency changes as the air heats from 0° to 500° Celsius.

The square-root relationship between temperature and the speed of sound in air is apparent and governs the behaviour of the cut-on frequency.

Lowest Cut-on Frequency for a Circular Pipe with Air Flow Versus Air Temperature

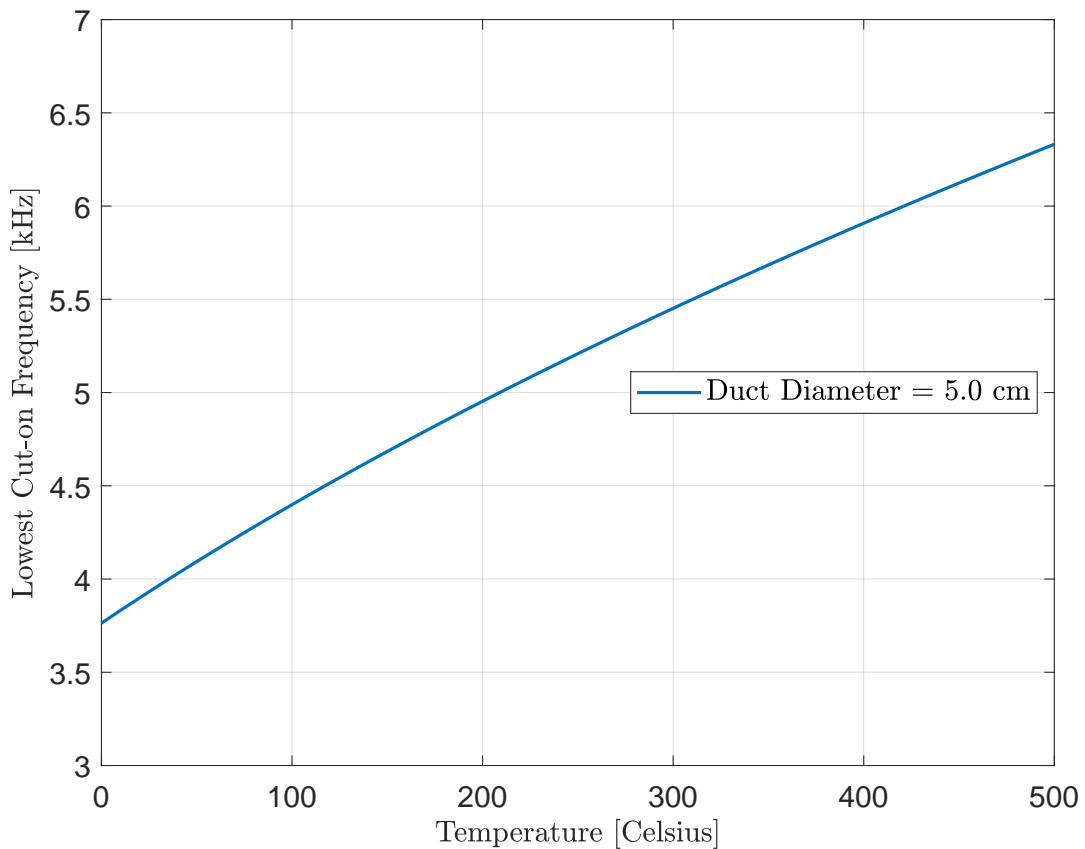


Figure 1: Lowest cut-on frequency for a circular 5 cm diameter duct versus air temperature.

Problem 1e

Question: Are cut-on frequencies higher for a circular or rectangular duct for a given cross-sectional area?

The lowest cut-on frequency is higher for a circular duct than for a rectangular duct for a given cross-sectional area.

For the dimensions given in class, the rectangular duct is not square. This produces a larger dimension and thus a smaller, lowest cut-on frequency. If the rectangular duct is square dimensions on the order of the circular duct diameter with the same cross-sectional area, the cut-on frequencies are approximately equal.

Question: What about in air versus water?

The lowest cut-on frequency is larger for water than for air. The cut-on frequency is proportional to the speed of sound and the speed of sound in water is greater than the speed of sound in air.

Question: What about cold versus hot air?

The lowest cut-on frequency is higher for warm air than it is for cold air.

Problem 2 - Muffler Design Comparison

The Matlab code for this problem is listed in Appendix 2.

For these muffler comparisons, it is assumed that there is no resistive terms and no flow. Since transmission losses are considered, the end corrections (i.e., load impedance at outlet of the system) have no physical meaning and are not accounted for in the computations.

Problems 2a, 2b, and 2c

Figure 2 shows the transmission loss profiles for a simple expansion chamber, a double-tuned expansion chamber, and a cascaded double-tuned expansion chamber muffler.

The peaks for the simple expansion chamber (red, dashed line) are approximately 22 dB and occur at frequencies with a wavelength that is a quarter of the length of the expansion chamber. Minimal loss occurs at half wavelength multiples.

The addition of the extension tube inside the muffler produces a quarter wavelength resonator. The side branch of Ji (2005; Slide 11, Lecture 3 notes) was used to calculate L_0 . For the cascaded double-tuned expansion chamber, the extension tubes produce a secondary quarter wavelength resonator.

As noted in the office hours session, there is no damping which produces artificially high resonances.

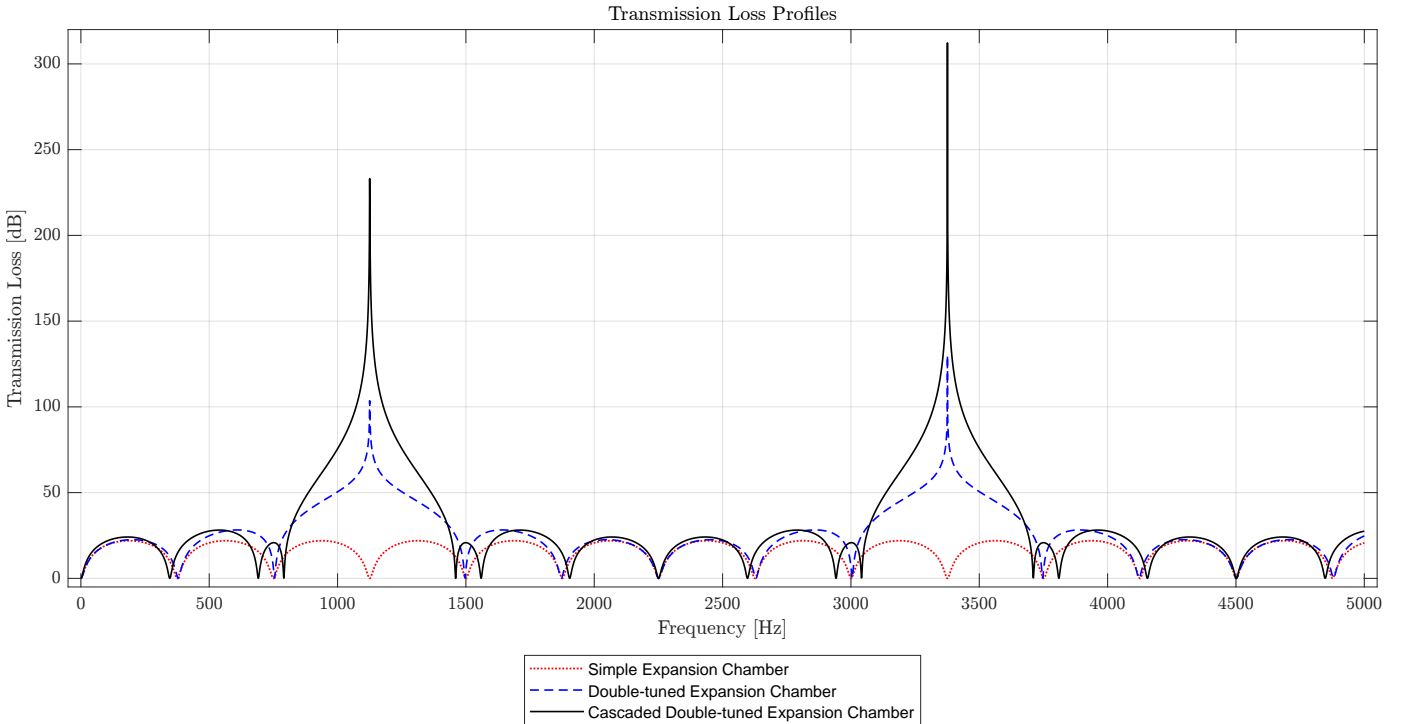


Figure 2: Transmission loss profiles for a simple expansion chamber, a double-tuned expansion chamber, and a cascaded double-tuned expansion chamber mufflers.

Problem 2d

Figure 3 shows the transmission loss profiles for a cascaded double-tuned expansion chamber, and a modified cascaded double-tuned expansion chamber muffler.

Two modifications were made to the original system:

1. The left 3" extension tube in the left chamber was shortened to 2" inches, making the respective muffler section 1" longer.

2. The left 3" extension tube in the right chamber was lengthened to 4", making the respective muffler section 1" shorter.

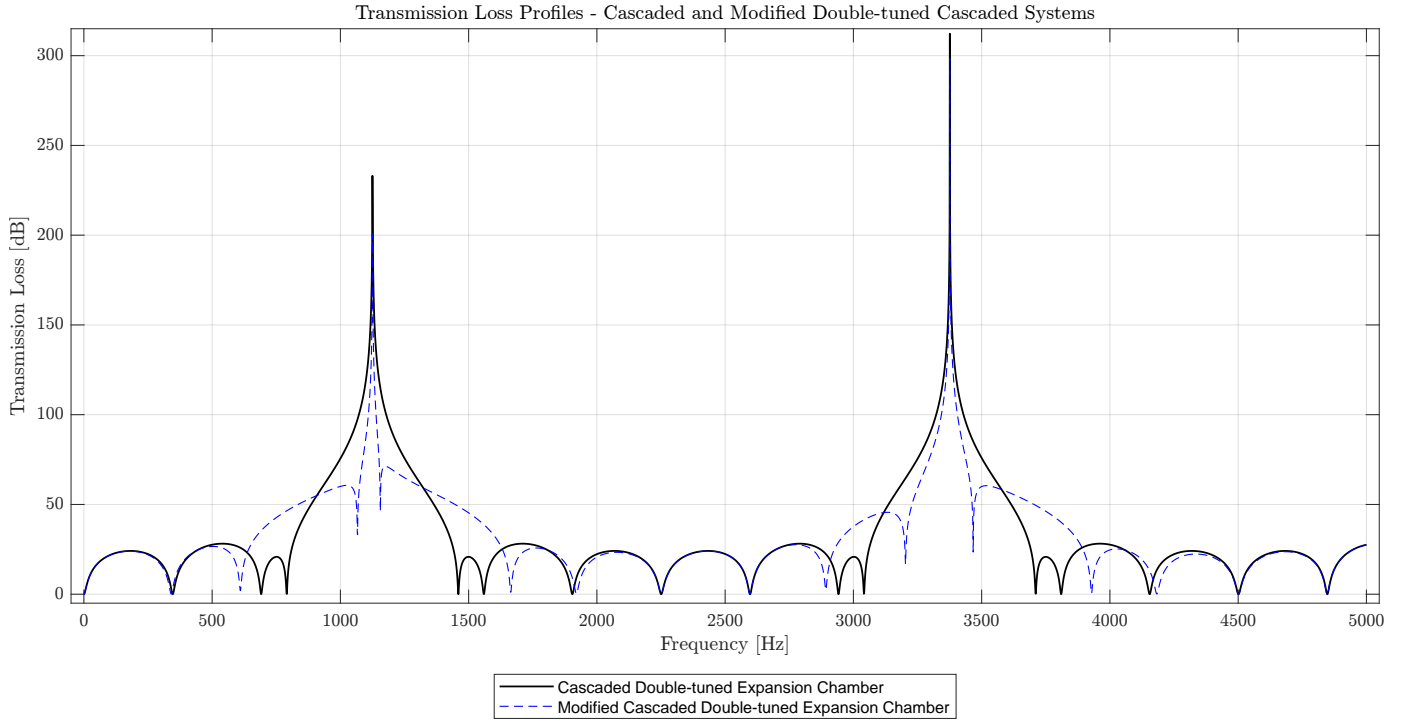


Figure 3: Transmission loss profiles for a cascaded double-tuned expansion chamber muffler and a modified version of this muffler.

These modifications change the symmetry of the cascaded system, and allow the resonate frequencies to be independently changed.

Problem 3 - Bugle Recorder

Diameters of holes should be smaller than a wavelength.

R_A is neglected (energy loss).

Problem 3a

Problem 3b

Problem 4 - Intake Duct

Problem 4a

Problem 4b

Problem 4c

Problem 4d

Problem 5 - Intake Duct Silencer

Problem 5a

Problem 5b

Problem 5c

Problem 5d

Problem 5e

1 Appendix - Matlab Code for Problem 1

```
1
2
3 %% Synopsis
4
5 % Question 1 – Cut-on Frequencies in Ducts and Pipes
6
7
8
9 %% Environment
10
11 close all; clear; clc;
12 % restoredefaultpath;
13
14 % addpath( genpath( '' ), '-begin' );
15 addpath( genpath( '../40 Assignments/00 Support' ), '-begin' );
16
17 % set( 0, 'DefaultFigurePosition', [ 400 400 900 400 ] ); % [ left bottom width height ]
18 set( 0, 'DefaultFigurePaperPositionMode', 'manual' );
19 set( 0, 'DefaultFigureWindowStyle', 'normal' );
20 set( 0, 'DefaultLineLineWidth', 1.5 );
21 set( 0, 'DefaultTextInterpreter', 'Latex' );
22
23 format ShortG;
24
25 pause( 1 );
26
27 PRINT_FIGURES = 0;
28
29
30
31 %% Define Constants and Anonymous Functions
32
33 c_air = 343; % The speed of sound in air (meters per second).
34 c_water = 1500; % The speed of sound in water (meters per second).
35
36 gamma = 1.4; % The ratio of specific heats [unitless].
37 R = 287; % The gas constant [Joules per ( kilogram * Kelvin)].
38
39
40 h_f_cut_on_rectangular_duct = @( c, L ) 0.5 .* c ./ L;
41 %
42 % c – The speed of sound.
43 % L – The largest cross-section dimension of the rectangular duct.
44
45
46 h_f_cut_on_circular_duct = @( c, d ) 0.568 .* c ./ d;
47 %
48 % c – The speed of sound.
49 % L – The diameter of the circular duct.
50
51
52 h_speed_of_sound_in_air = @( gamma, R, temperature_Kelvin ) sqrt( gamma .* R .*
    temperature_Kelvin );
53
54
55
56 %% Problem 1a
57
58 % The cross-sectional dimensions for the rectangular duct are: Lx = 12 cm and Ly = 20 cm.
59
60 % The largest dimension is Ly = 20 cm or 0.2 m.
61
62 % The cut-on frequency is,
63 h_f_cut_on_rectangular_duct( c_air, 0.2 ); % 857.5 Hz (shown in class 858 Hz)
64 fprintf( 1, '\n Problem 1a: The lowest cut-on frequency for the rectangular pipe with air is
    %3.1f Hz.\n', h_f_cut_on_rectangular_duct( c_air, 0.2 ) );
65
66
67
68 %% Problem 1b
69
70 % The cross-sectional dimensions for the rectangular duct are: Lx = 12 cm and Ly = 20 cm.
71
72 % The cross-sectional area of the rectangular duct is 12 cm * 20 cm = 240 cm^2 or 0.024 m^2.
73 rectangular_duct_cross_sectional_area = 0.12 * 0.20; % 0.024 m^2
```

```

74
75 % The diameter of the circular pipe is,
76 circular_duct_diameter = sqrt( 0.024 / pi ) * 2; % 0.17481 meters
77 %
78 % Check:
79 % pi * ( circular_duct_diameter / 2 )^2 CHECKED
80
81
82 % The cut-on frequency for the circular duct is,
83 h_f_cut_on_circular_duct( c_air, circular_duct_diameter ); % 1,114.5 Hz
84 fprintf( 1, '\n Problem 1b: The lowest cut-on frequency for the circular pipe (of equal area
    ) with air is %3.1f Hz.\n', h_f_cut_on_circular_duct( c_air, circular_duct_diameter ) );
85
86
87
88 %% Problem 1c
89
90 % The cut-on frequency for the circular duct with water is,
91 h_f_cut_on_circular_duct( c_water, circular_duct_diameter ); % 4,873.9 Hz
92 fprintf( 1, '\n Problem 1c: The lowest cut-on frequency for the circular pipe (of equal area
    ) with water is %3.1f Hz.\n', h_f_cut_on_circular_duct( c_water, circular_duct_diameter ) );
93
94 % The cut-on frequency should be higher because it is proportional to the
95 % speed of sound in a given medium.
96
97
98
99 %% Problem 1d
100
101 fprintf( 1, '\n Problem 1d: See the figure.\n' );
102
103 temperature_range_celsius = 0:0.1:500; % Celsius
104 temperature_range_kelvin = temperature_range_celsius + 273.15; % Kelvin
105
106
107 FONT_SIZE = 14;
108
109 figure( ); ...
110 plot( temperature_range_celsius, h_f_cut_on_circular_duct( h_speed_of_sound_in_air( gamma, R,
    temperature_range_kelvin ), 0.05 ) ./ 1e3 ); grid on;
111 legend( 'Duct Diameter = 5.0 cm', 'Location', 'East', 'FontSize', FONT_SIZE, 'Interpreter
    ', 'Latex' );
112 set( gca, 'FontSize', FONT_SIZE );
113 %
114 xlabel( 'Temperature [Celsius]', 'FontSize', FONT_SIZE );
115 % xl = get( gca, 'xlabel' ); pxl = get( xl, 'position' ); pxl( 2 ) = 1.1 * pxl( 2 );
116 % set( xl, 'position', pxl );
117 %
118 ylabel( 'Lowest Cut-on Frequency [kHz]', 'FontSize', FONT_SIZE );
119 % yl = get( gca, 'ylabel' ); pyl = get( yl, 'position' ); pyl( 1 ) = 1.2 * pyl( 1 );
120 % set( yl, 'position', pyl );
121 %
122 caption = sprintf( 'Lowest Cut-on Frequency for a Circular Pipe with Air Flow Versus Air
    Temperature\n' );
123 title( caption, 'FontSize', FONT_SIZE );
124 %
125 ylim( [ 3 7 ] );
126
127
128
129 %% Problem 1e
130
131 fprintf( 1, '\n Problem 1e: See Section Problem 1e of the Matlab script for the answers.\n\n' );
132
133
134 % Question: Are cut-on frequencies higher for a circular or rectangular duct for a given cross-
    sectional area?
135
136 % The lowest cut-on frequency is higher for a circular duct than for a
137 % rectangular duct for a given cross-sectional area.
138
139 % For the dimensions given in class, the rectangular duct is not square.
140 % This produces a larger dimension and thus a smaller, lowest cut-on
141 % frequency.
142
143 % If the rectangular duct is square dimensions on the order of the circular
144 % duct diameter with the same cross-sectional area, the the cut-on
145 % frequencies are approximately equal.

```

```

146
147
148 % Question:  What about in air versus water?
149
150 % The lowest cut-on frequency is larger with water than air.  This due to
151 % the fact that the cut-on frequency is proportional to the speed of sound
152 % and the speed of sound in water is greater than it is in air.
153
154
155 % Question:  What about cold versus hot air?
156
157 % For a circular pipe, the cut-on frequency is higher in warm air than cold
158 % air.
159
160
161
162 %% Clean-up
163
164 if ( ~isempty( findobj( 'Type', 'figure' ) ) )
165     monitors = get( 0, 'MonitorPositions' );
166     if ( size( monitors, 1 ) == 1 )
167         autoArrangeFigures( 2, 2, 1 );
168     elseif ( 1 < size( monitors, 1 ) )
169         autoArrangeFigures( 2, 2, 1 );
170     end
171 end
172
173 if ( PRINT_FIGURES == 1 )
174     saveas((gcf, 'Cut-on Frequency Versus Temperature - Sunday, January 19, 2025.pdf' );
175 end
176
177 fprintf( 1, '\n\n*** Processing Complete ***\n\n' );

```

2 Appendix - Matlab Code for Problem 2

```
1
2
3
4 %% Synopsis
5
6 % Question 2 – Muffler Design Comparison
7
8
9
10 %% Environment
11
12 close all; clear; clc;
13 % restoredefaultpath;
14
15 % addpath( genpath( '' ), '-begin' );
16 addpath( genpath( '../00 Support' ), '-begin' );
17
18 % set( 0, 'DefaultFigurePosition', [ 400 400 900 400 ] ); % [ left bottom width height ]
19 set( 0, 'DefaultFigurePaperPositionMode', 'manual' );
20 set( 0, 'DefaultFigureWindowStyle', 'normal' );
21 set( 0, 'DefaultLineLineWidth', 1.5 );
22 set( 0, 'DefaultTextInterpreter', 'Latex' );
23
24 format ShortG;
25
26 pause( 1 );
27
28 PRINT_FIGURES = 0;
29
30
31
32 %% Constants
33
34 rho0 = 1.21; % Air density (kg per m^3).
35 c = 343; % Speed of sound in air (meters per second).
36
37 frequency_set = 0:1:5e3; % Hertz
38
39
40
41 %% Dimensions
42
43 convert.inches_to_meters = 0.0254;
44 convert.foot_to_meters = 0.3048;
45
46 dimensions.inlet_diameter_meters = 2 * convert.inches_to_meters; % 0.0508 meters
47 dimensions.inlet_length_meters = 6 * convert.foot_to_meters; % 1.82 meters
48
49 dimensions.muffler_diameter_meters = 10 * convert.inches_to_meters; % 0.254 meters
50 dimensions.muffler_length_meters = 18 * convert.inches_to_meters; % 0.4572 meters
51
52 dimensions.outlet_diameter_meters = 2 * convert.inches_to_meters; % 0.0508 meters
53 dimensions.outlet_length_meters = 1 * convert.foot_to_meters; % 0.3048 meters
54
55 outlet_flanged = false;
56
57 dimensions.overhang = 3 * convert.inches_to_meters; % 0.0762 meters
58
59 segment_diameters = [ ...
60     dimensions.outlet_diameter_meters, ...
61     dimensions.muffler_diameter_meters, ...
62     dimensions.inlet_diameter_meters, ...
63 ].';
64 %
65 h_area_from_diameter = @( d ) pi .* d.^2 ./ 4;
66 %
67 segment_areas = h_area_from_diameter( segment_diameters );
68
69 segment_lengths = [ ...
70     dimensions.outlet_length_meters, ...
71     dimensions.muffler_length_meters, ...
72     dimensions.inlet_length_meters, ...
73     dimensions.overhang, ...
74 ].';
75
```

```

76
77
78 %% Part a - Simple Expansion Chamber
79
80 nFreq = length( frequency_set );
81 TL = zeros( nFreq, 1 );
82
83 for frequency_index = 1:1:nFreq
84
85     f = frequency_set( frequency_index );
86
87     T_total = [ 1 0; 0 1 ];
88
89     T1 = duct_segment_transfer_matrix( f, rho0, c, 0.3048, 0.0020268 ); % Duct - Outlet
90     T2 = duct_segment_transfer_matrix( f, rho0, c, 0.4572, 0.050671 ); % Duct
91     T3 = duct_segment_transfer_matrix( f, rho0, c, 1.8288, 0.0020268 ); % Duct - Inlet
92
93     T_net = T3 * T2 * T1 * T_total;
94     % T_net = T_inlet * T_total; % Zero transmission loss for a straight duct.
95
96     T11 = T_net(1, 1); T12 = T_net(1, 2); T21 = T_net(2, 1); T22 = T_net(2, 2);
97     TL( frequency_index ) = 10 * log10( abs( ( T11 + 0.0020268*T12/(rho0*c) + (rho0*c)*
98         T21/0.0020268 + T22 ) / 2 )^2 );
99     %
100     % The transmission loss calculation does not require a load impedance.
101 end
102
103 TL_parta = TL; % The maximum peak value should be about 22 (21.952) dB.
104 %
105 % Expected behaviour:
106 %
107 % 1.) 0 dB at 0 Hz.
108 % 2.) The transmission loss of a straight duct section is zero; energy out equals energy in.
109
110 max( TL_parta ); % 22 dB
111
112
113
114 %% Part b - Double-tuned Expansion Chamber
115
116 annulus_area_squared_meters = pi/4 * ( 0.254^2 - 0.0508^2 );
117
118 L_o = 0; % Assume that the L_o extension is negligible.
119
120
121 nFreq = length( frequency_set );
122 TL = zeros( nFreq, 1 );
123
124 for frequency_index = 1:1:nFreq
125
126     f = frequency_set( frequency_index );
127
128     T_total = [ 1 0; 0 1 ];
129
130     T1 = duct_segment_transfer_matrix( f, rho0, c, (0.3048 + 0.0762), 0.0020268 );
131     T3 = duct_segment_transfer_matrix( f, rho0, c, (0.4572 - 2*0.0762), 0.050671 );
132     T5 = duct_segment_transfer_matrix( f, rho0, c, (1.8288 + 0.0762), 0.0020268 );
133
134     k = 2*pi*f/c;
135     Z_A = -1j*rho0*c/annulus_area_squared_meters*cot( k * ( 0.0762 + L_o ) );
136     T2 = [ 1 0; 1/Z_A 1 ];
137     T4 = T2;
138
139     T_net = T5 * T4 * T3 * T2 * T1 * T_total;
140
141
142     T11 = T_net(1, 1); T12 = T_net(1, 2); T21 = T_net(2, 1); T22 = T_net(2, 2);
143     TL( frequency_index ) = 10 * log10( abs( ( T11 + 0.0020268*T12/(rho0*c) + (rho0*c)*
144         T21/0.0020268 + T22 ) / 2 )^2 );
145     %
146     % The transmission loss calculation does not require a load impedance.
147 end
148
149 TL_partb = TL;
150 %
151 % Expected behaviour:

```

```

152 %
153 % 1.) 0 dB at 0 Hz.
154 % 2.) 0 dB at same locations as a simple expansion chamber.
155 % 3.) Peaks at 1,125 Hz and 3,376 Hz;
156
157 % Frequency at which the quarter-wavelength is 0.0762 meters.
158 %  $343 / (4 * 0.0762)$ ; % 1,125 Hz.
159
160 % Also work at three-quarter-wavelength.
161 %  $3 * 1125$ ; % 3,375 Hz
162
163
164
165 %% Part c - Cascaded, Double-tuned Expansion Chamber
166
167 annulus_area_squared_meters = pi/4 * ( 0.254^2 - 0.0508^2 );
168
169 L_o = 0; % Assume that the L_o extension is negligible.
170
171
172 nFreq = length( frequency_set );
173 TL = zeros( nFreq, 1 );
174
175 for frequency_index = 1:1:nFreq
176
177     f = frequency_set( frequency_index );
178     k = 2*pi*f/c;
179     Z_A = -lj*rho0*c/annulus_area_squared_meters*cot(k * ( 0.0762 + L_o ) );
180
181     T_total = [ 1 0; 0 1 ];
182
183     T1 = duct_segment_transfer_matrix( f, rho0, c, (0.3048 + 0.0762), 0.0020268 ); % Duct -
184     % Outlet
185     T2 = [ 1 0; 1/Z_A 1 ]; % Straight Side Branch
186     T3 = duct_segment_transfer_matrix( f, rho0, c, (0.2286 - 2*0.0762), 0.050671 ); % Duct
187     T4 = T2; % Straight Side Branch
188     T5 = duct_segment_transfer_matrix( f, rho0, c, 2*0.0762, 0.050671 ); % Duct
189     T6 = T2; % Straight Side Branch
190     T7 = T3; % Duct
191     T8 = T2; % Straight Side Branch
192     T9 = duct_segment_transfer_matrix( f, rho0, c, (1.8288 + 0.0762), 0.0020268 ); % Duct -
193     % Inlet
194
195     T_net = T9 * T8 * T7 * T6 * T5 * T4 * T3 * T2 * T1 * T_total;
196
197     T11 = T_net(1, 1); T12 = T_net(1, 2); T21 = T_net(2, 1); T22 = T_net(2, 2);
198     TL( frequency_index ) = 10 * log10( abs( ( T11 + 0.0020268*T12/(rho0*c) + (rho0*c)*
199     T21/0.0020268 + T22 ) / 2 )^2 );
200
201 end
202
203 TL_partc = TL;
204
205
206 %% Part d - Cascaded, Double-tuned Expansion Chamber
207
208 nFreq = length( frequency_set );
209 TL = zeros( nFreq, 1 );
210
211 for frequency_index = 1:1:nFreq
212
213     f = frequency_set( frequency_index );
214     k = 2*pi*f/c;
215     Z_A = -lj*rho0*c/annulus_area_squared_meters*cot(k * ( 0.0762 + L_o ) );
216
217     T_total = [ 1 0; 0 1 ];
218
219     T1 = duct_segment_transfer_matrix( f, rho0, c, (0.3048 + 0.0762), 0.0020268 ); % Duct -
220     % Outlet
221     T2 = [ 1 0; 1/Z_A 1 ]; % Straight Side Branch
222     T3 = duct_segment_transfer_matrix( f, rho0, c, 0.0076, 0.050671 ); % Duct
223     T4 = T2; % Straight Side Branch
224     T5 = duct_segment_transfer_matrix( f, rho0, c, 2*0.0762, 0.050671 ); % Duct
225     T6 = T2; % Straight Side Branch
226     T7 = duct_segment_transfer_matrix( f, rho0, c, (0.29718 - 2*0.0762), 0.050671 ); % Duct
227     T8 = T2; % Straight Side Branch
228     T9 = duct_segment_transfer_matrix( f, rho0, c, (1.8288 + 0.0762), 0.0020268 ); % Duct -

```

```

226
227 T_net = T9 * T8 * T7 * T6 * T5 * T4 * T3 * T2 * T1 * T_total;
228
229 T11 = T_net(1, 1); T12 = T_net(1, 2); T21 = T_net(2, 1); T22 = T_net(2, 2);
230 TL( frequency_index ) = 10 * log10( abs( ( T11 + 0.0020268*T12/(rho0*c) + (rho0*c)*
T21/0.0020268 + T22 ) / 2 )^2 );
231
232 end
233
234 TL_partd = TL;
235
236
237
238 %% Plot Transmission Loss Profiles
239
240 Y_LIMITS = [ -5 320 ];
241
242 h_figure_1 = figure( ); ...
243 plot( frequency_set, TL_parta, 'LineWidth', 1.0, 'LineStyle', ':', 'Color', 'r' ); hold on;
244 plot( frequency_set, TL_partb, 'LineWidth', 0.9, 'LineStyle', '—', 'Color', 'b' );
245 plot( frequency_set, TL_partc, 'LineWidth', 0.9, 'LineStyle', '—', 'Color', 'k' ); grid on;
246 grid on;
247 legend( ...
248     'Simple Expansion Chamber', ...
249     'Double-tuned Expansion Chamber', ...
250     'Cascaded Double-tuned Expansion Chamber', ...
251     'Location', 'SouthOutside' );
252 xlabel( 'Frequency [Hz]' ); ylabel( 'Transmission Loss [dB]' );
253 title( 'Transmission Loss Profiles' );
254 %
255 Ax = gca;
256 Ax.XAxis.TickLabelInterpreter = 'latex';
257 Ax.YAxis.TickLabelInterpreter = 'latex';
258 %
259 axis( [ -50 5e3+50 Y_LIMITS ] );
260
261
262 Y_LIMITS = [ -5 315 ];
263
264 h_figure_2 = figure( ); ...
265 plot( frequency_set, TL_partc, 'LineWidth', 1.0, 'LineStyle', '—', 'Color', 'k' ); hold on;
266 plot( frequency_set, TL_partd, 'LineWidth', 0.6, 'LineStyle', '—', 'Color', 'b' ); grid on;
267 legend( ...
268     'Cascaded Double-tuned Expansion Chamber', ...
269     'Modified Cascaded Double-tuned Expansion Chamber', ...
270     'Location', 'SouthOutside' );
271 xlabel( 'Frequency [Hz]' ); ylabel( 'Transmission Loss [dB]' );
272 title( 'Transmission Loss Profiles – Cascaded and Modified Double-tuned Cascaded Systems' );
273 %
274 Ax = gca;
275 Ax.XAxis.TickLabelInterpreter = 'latex';
276 Ax.YAxis.TickLabelInterpreter = 'latex';
277 %
278 axis( [ -50 5e3+50 Y_LIMITS ] );
279
280
281
282 %% Clean-up
283
284 if ( ~isempty( findobj( 'Type', 'figure' ) ) )
285     monitors = get( 0, 'MonitorPositions' );
286     if ( size( monitors, 1 ) == 1 )
287         autoArrangeFigures( 2, 2, 1 );
288     elseif ( 1 < size( monitors, 1 ) )
289         autoArrangeFigures( 2, 2, 1 );
290     end
291 end
292
293 if ( PRINT_FIGURES == 1 )
294     exportgraphics( h_figure_1, 'Assignment 1 – Question 2 Figure All TL Profiles.pdf', '
Append', true );
295     exportgraphics( h_figure_2, 'Assignment 1 – Question 2 Figure Comparison TL Plot For
Cascaded Systems.pdf', 'Append', true );
296 end
297
298 fprintf( 1, '\n\n*** Processing Complete ***\n\n\n' );

```


3 Appendix - Matlab Code for Problem 3

```
1
2
3
4 %% Synopsis
5
6 % Question 3 – Bugle Recorder
7
8
9
10 %% Note(s)
11
12 % For the lowest frequency, use 1 duct segment with an open-ended
13 % impedance (see the example of the horn in class).
14
15
16 % See "2025-01-31 13_07_33-Zoom Meeting.png".
17
18
19
20 %% Environment
21
22 close all; clear; clc;
23 % restoredefaultpath;
24
25 % addpath( genpath( '' ), '-begin' );
26 addpath( genpath( '../00 Support' ), '-begin' );
27
28 % set( 0, 'DefaultFigurePosition', [ 400 400 900 400 ] ); % [ left bottom width height ]
29 set( 0, 'DefaultFigurePaperPositionMode', 'manual' );
30 set( 0, 'DefaultFigureWindowStyle', 'normal' );
31 set( 0, 'DefaultLineLineWidth', 1.5 );
32 set( 0, 'DefaultTextInterpreter', 'Latex' );
33
34 format ShortG;
35
36 pause( 1 );
37
38 PRINT_FIGURES = 0;
39
40
41
42 %% Constants and Anonymous Functions
43
44 rho0 = 1.21; % Density of air (kg per cubic-meter).
45 c = 343; % Speed of sound in air (meters per second).
46
47
48 h_R_A = @( rho0, c, S, k, delta_mu, D, w, gamma, h, epsilon, M ) ...
49 ( rho0*c/S ) * ( ( k*delta_mu*D*w ) / ( 2*S ) ) * ( 1 + ( gamma - 1 ) * sqrt( 5 / ( 3*gamma ) ) ) +
50 0.288*k*delta_mu*log10( ( 4*S ) / ( pi*h^2 ) ) + ( epsilon*S*k^2 / ( 2*pi ) + 0.7*M );
51 % See Equation 8.34 on page 479 of Bies et al (2024).
52
53
54
55 %% Define Shape
56
57 L_mouth_piece = 0.09; % Meters
58
59 pipe.inner_diameter = 0.009; % Meters
60 pipe.thickness = 0.004; % Meters
61
62 % The recorder is unflanged.
63
64 hole_diameter = 0.006; % Meters
65
66
67
68 %% Part a
69
70 % Determine the length of the recorder to produce 523 Hz.
71
72 % The total length of the recorder, including the 0.09 meter long mouthpiece, is L.
73
74 a = 0.009 / 2; % Meters
```

```

75     L_o = 0.61*a; % Slide 18 of Lecture 2 slide set.
76
77     f = 523; % Hz
78     k = 2*pi*f/c; % The wave number for the respective frequency.
79
80     S = pi/4*(0.009)^2; % squared-meters
81
82
83     test_lengths = 0:1e-3:1;
84     test_lengths = test_lengths + 0.09;
85
86     nLengths = length( test_lengths );
87     A = zeros( nLengths, 1 );
88
89     for iLength = 1:1:nLengths
90
91         L = test_lengths(iLength);
92
93         T_total = [ 1 0; 0 1 ];
94
95         L_e = L + L_o;
96         % Z = 1j * rho0 * c / S * tan( k * L_e );
97         Z = open_end_impedance( f, rho0, c, 0, S(1), 0 );
98
99         T = [ ...
100             cos(k*L), 1j*rho0*c/S*sin(k*L); ...
101             1j*S/(rho0*c)*sin(k*L), cos(k*L) ...
102             ];
103
104
105         T_total = T * T_total;
106         T11 = T_total(1, 1); T12 = T_total(1, 2);
107
108         A( iLength ) = -10*log10( abs( T11 + T12 / Z )^2 );
109
110     end
111
112
113     figure( ); ...
114     plot( test_lengths * 1e3, A ); grid on;
115     xlabel( 'Total Recorder Length [mm]' ); ylabel( 'Amplitude [dB]' );
116     title( 'Amplification Versus Recorder Length' );
117
118     return
119
120 %% Part b
121
122 % L_net = 0.325; % Meters - First Peak
123 L_net = 0.653; % Meters - Second Peak
124
125 a = 0.009 / 2; % Meters
126 L_o = 0.61*a; % Slide 18 of Lecture 2 slide set.
127
128 f = 698; % Hz
129 k = 2*pi*f/c; % The wave number for the respective frequency.
130
131 S = pi/4*(0.009)^2; % squared-meters
132
133
134 test_lengths = 0:0.001:0.5;
135 test_lengths = L_net - test_lengths;
136
137 nLengths = length( test_lengths );
138 A = zeros( nLengths, 1 );
139
140 for iLength = 1:1:nLengths
141
142     L = test_lengths(iLength);
143     L_duct_2 = L;
144     L_duct_1 = L_net - L_duct_2;
145
146     T_total = [ 1 0; 0 1 ];
147
148
149     % End section.
150     T_1 = [ ...
151         cos(k*L_duct_1), 1j*rho0*c/S*sin(k*L_duct_1); ...
152         1j*S/(rho0*c)*sin(k*L_duct_1), cos(k*L_duct_1) ...

```

```

153 ];
154
155
156 % Orifice side branch.
157 epsilon = 0.006 / 0.009; % 0.67
158 a = 0.006 / 2;
159 %
160 L_o = a * ( 0.9326 - 0.6196*epsilon ); % Lecture 3, Slide 11
161 L_e = 0.004 + 2*L_o;
162 %
163 Z_A = 1j * rho0 * ( 2 * pi * f ) * L_e / ( pi*0.006^2/4 );
164 %
165 k = 2*pi*f/c; % The wave number for the respective frequency.
166 S_hole = pi/4*(0.006)^2; % squared-meters
167 mu = 1.83e-5; % kg per meter-second; online reference.
168 delta_mu = sqrt( ( 2 * mu ) / ( 2*pi*f * rho0 ) );
169 D = pi * 0.006;
170 w = 2*pi*f;
171 gamma = 1.4;
172 h = 0.003; % Larger of the edge radius or delta_mu.
173 Mach_number = 0;
174 R_A = h_R_A( rho0, c, S_hole, k, delta_mu, D, w, gamma, h, epsilon, Mach_number );
175 %
176 % Z_A = Z_A + R_A
177 T_Branch = [ 1 0; 1/Z_A 1 ];
178
179
180 % Section next to mouthpiece.
181 T_2 = [ ...
182 cos(k*L_duct_2), 1j*rho0*c/S*sin(k*L_duct_2); ...
183 1j*S/(rho0*c)*sin(k*L_duct_2), cos(k*L_duct_2) ...
184 ];
185
186
187 T_total = T_2 * T_Branch * T_1 * T_total;
188 T11 = T_total(1, 1); T12 = T_total(1, 2);
189
190 A( iLength ) = -10*log10( abs( T11 + T12 / Z )^2 );
191
192 end
193
194
195 figure( ); ...
196 plot( test_lengths * 1e3, A ); grid on;
197 set( gca, 'XDir', 'reverse' );
198 xlabel( 'Offset from End of 162.6 mm Length Recorder [mm]' ); ylabel( 'Amplitude [dB]' );
199 title( 'Amplification Versus Offset from End of Recorder' );
200
201 return
202
203 %% Part b Verification
204
205 a = 0.009 / 2; % Meters
206 L_o = 0.61*a; % Slide 18 of Lecture 2 slide set.
207
208 f = 698; % Hz
209 k = 2*pi*f/c; % The wave number for the respective frequency.
210
211 S = pi/4*(0.009)^2; % squared-meters
212
213
214 f = 0:1:5e3;
215
216 nFreq = length( f );
217 A = zeros( nFreq, 1 );
218
219
220 L1 = 0.2
221 L2 = 0.325 - L1;
222
223
224 for iFreq = 1:1:nFreq
225
226 k = 2*pi*f(iFreq)/c;
227
228 T_total = [ 1 0; 0 1 ];
229
230 % End duct.

```

```

231 T_1 = [ ...
232 cos(k*L1) , 1j*rho0*c/S*sin(k*L1); ...
233 1j*S/(rho0*c)*sin(k*L1) , cos(k*L1) ...
234 ];
235
236
237 % Orifice side branch.
238 epsilon = 0.006 / 0.009; % 0.67
239 a = 0.006 / 2;
240
241 L_o = a * ( 0.9326 - 0.6196*epsilon ); % Lecture 3, Slide 11
242 L_e = 0.004 + 2*L_o;
243 %
244 Z_A = 1j * rho0 * 2 * pi * f(iFreq) * L_e / ( pi*0.006^2/4 );
245 T_Branch = [ 1 0; 1/Z_A 1 ];
246 %
247 % R_A is neglected (energy loss).
248
249
250 % Front duct.
251 T_2 = [ ...
252 cos(k*L2) , 1j*rho0*c/S*sin(k*L2); ...
253 1j*S/(rho0*c)*sin(k*L2) , cos(k*L2) ...
254 ];
255
256
257 T_total = T_2 * T_Branch * T_1 * T_total;
258
259 T11 = T_total(1, 1); T12 = T_total(1, 2);
260 A( iFreq ) = -10*log10( abs( T11 + T12 / Z )^2 );
261
262 end
263
264
265 figure( ); ...
266 plot( f, A ); grid on;
267 xlabel( 'Frequency [Hz]' ); ylabel( 'Amplitude [dB]' );
268 title( 'Amplification Versus Frequency' );
269
270
271
272 %% Clean-up
273
274 if ( ~isempty( findobj( 'Type', 'figure' ) ) )
275     monitors = get( 0, 'MonitorPositions' );
276     if ( size( monitors, 1 ) == 1 )
277         autoArrangeFigures( 2, 2, 1 );
278     elseif ( 1 < size( monitors, 1 ) )
279         autoArrangeFigures( 2, 2, 1 );
280     end
281 end
282
283
284 fprintf( 1, '\n\n*** Processing Complete ***\n\n' );
285
286
287
288 %% Reference(s)

```

4 Appendix - Matlab Code for Problem 4

```
1
2
3 %% Synopsis
4
5 % Question 4 – Intake Duct
6
7
8
9 %% To Do
10
11 % Check loss profile.
12
13 % Implement the loss Helmholtz resonator.
14
15
16
17 %% Note(s)
18
19 % Search for FIXMEs.
20
21
22 % In class note, the areas for the impedance might have bee wrong; switch them?
23
24
25 % Use negative Mach numbers in the equations. The analysis for this case
26 % is the same as for the horn example. Inlet on the left, outlet on the
27 % right.
28
29
30
31 %% Environment
32
33 close all; clear; clc;
34 % restoredefaultpath;
35
36 % addpath( genpath( '' ), '-begin' );
37 addpath( genpath( '../40 Assignments/00 Support' ), '-begin' );
38
39 % set( 0, 'DefaultFigurePosition', [ 400 400 900 400 ] ); % [ left bottom width height ]
40 set( 0, 'DefaultFigurePaperPositionMode', 'manual' );
41 set( 0, 'DefaultFigureWindowStyle', 'normal' );
42 set( 0, 'DefaultLineLineWidth', 1.5 );
43 set( 0, 'DefaultTextInterpreter', 'Latex' );
44
45 format ShortG;
46
47 pause( 1 );
48
49 PRINT_FIGURES = 0;
50
51
52
53 %% Constants
54
55 rho0 = 1.21; % Density of air (kg per cubic-meter).
56 c = 343; % Speed of sound in air (meters per second).
57
58 h_area = @( diameter ) pi * diameter^2 / 4;
59
60
61
62 %% Define Shape
63
64 % Source
65 duct_1.diameter_meters = 0.0254; % 1 inch
66 duct_1.length_meters = 0.1524; % 6 inches
67 duct_1.area = h_area( duct_1.diameter_meters );
68
69 % Outlet
70 duct_2.diameter_meters = 0.1016; % 4 inches
71 duct_2.length_meters = 0.127; % 5 inches
72 duct_2.area = h_area( duct_2.diameter_meters );
73 %
74 % Flanged.
75
```

```

76 %
77
78 flow_rate_cubic_meters_per_second = 1.04772 / 60; % or 37 cubic-feet per minute
79
80 % return
81
82 %% Part a
83
84 % Calculate the Mach number of the flow in both pip sections.
85
86 duct_1.Mach = -1.0 * flow_rate_cubic_meters_per_second / (pi * duct_1.diameter_meters^2 / 4) / c
87 ; % 0.100 unitless
88 duct_2.Mach = -1.0 * flow_rate_cubic_meters_per_second / (pi * duct_2.diameter_meters^2 / 4) / c
89 ; % 0.00628 unitless
90
91 % return
92
93 %% Part b
94
95 % No flow.
96
97 outlet_flanged = true; % Flanged end.
98
99 TEST_FLAG = 1; % 1: right-to-left.
100
101 frequency_set = 0:0.1:2.5e3;
102 nFreq = length( frequency_set );
103 TL = zeros( nFreq, 1 );
104
105 for frequency_index = 1:1:nFreq
106
107     f = frequency_set( frequency_index );
108
109
110     T_total = [ 1 0; 0 1 ];
111
112
113     if ( TEST_FLAG == 1 )
114
115         % Right-to-left.
116         %
117         T_outlet = duct_segment_transfer_matrix( f, rho0, c, duct_2.length_meters, duct_2.area );
118         T_contraction = [ 1 0; 0 duct_2.area/duct_1.area ];
119         T_inlet = duct_segment_transfer_matrix( f, rho0, c, duct_1.length_meters, duct_1.area );
120
121         Z = open_end_impedance( f, rho0, c, duct_2.length_meters, duct_2.area, outlet_flanged );
122
123     else
124
125         % Left-to-right.
126         %
127         T_outlet = duct_segment_transfer_matrix( f, rho0, c, duct_1.length_meters, duct_1.area );
128         T_contraction = [ 1 0; 0 duct_1.area/duct_2.area ];
129         T_inlet = duct_segment_transfer_matrix( f, rho0, c, duct_2.length_meters, duct_2.area );
130
131         Z = open_end_impedance( f, rho0, c, duct_1.length_meters, duct_1.area, outlet_flanged );
132
133     end
134
135
136     T_net = T_inlet * T_contraction * T_outlet * T_total;
137
138
139     T11 = T_net(1, 1); T12 = T_net(1, 2); T21 = T_net(2, 1); T22 = T_net(2, 2);
140
141     if ( TEST_FLAG == 1 )
142         TL( frequency_index ) = 10 * log10( abs( ( T11 + duct_1.area*T12/(rho0*c) + (rho0*c)*
143         T21/duct_2.area + T22 ) / 2 )^2 );
144     else
145         TL( frequency_index ) = 10 * log10( abs( ( T11 + duct_2.area*T12/(rho0*c) + (rho0*c)*
146         T21/duct_1.area + T22 ) / 2 )^2 );
147     end
148
149 end % End: for f = frequency_set

```

```

150 TL_part_b = TL;
151
152 % return
153
154 %% Part c
155
156 % Flow present (use Mach numbers).
157
158 outlet_flanged = true; % Flanged end.
159
160
161 frequency_set = 0:0.1:2.5 e3;
162 nFreq = length( frequency_set );
163 TL = zeros( nFreq, 1 );
164
165
166 for frequency_index = 1:1:nFreq
167
168     f = frequency_set( frequency_index );
169
170
171     T_total = [ 1 0; 0 1 ];
172
173     T_outlet = duct_segment_transfer_matrix_flow( f, rho0, c, duct_2.length_meters, duct_2.area,
174     duct_2.Mach );
175
176     T_expansion = duct_expansion_connection_transfer_matrix( rho0, c, duct_2.area, duct_1.area,
177     duct_1.Mach );
178
179     T_inlet = duct_segment_transfer_matrix_flow( f, rho0, c, duct_1.length_meters, duct_1.area,
180     duct_1.Mach );
181
182     T_net = T_inlet * T_expansion * T_outlet * T_total;
183
184
185     Z = open_end_impedance( f, rho0, c, duct_2.length_meters, duct_2.area, outlet_flanged );
186
187     T11 = T_net(1, 1); T12 = T_net(1, 2); T21 = T_net(2, 1); T22 = T_net(2, 2);
188     TL( frequency_index ) = 10 * log10( abs( ( T11 + duct_1.area*T12/(rho0*c) + (rho0*c)*
189     T21/duct_2.area + T22 ) / 2 )^2 );
190
191
192 end % End: for f = frequency_set
193
194 TL_part_c = TL;
195
196 % return
197
198 %% Part d
199
200 % Flow present (use Mach numbers).
201 % Helmholtz resonator in place (between lefthand duct and expansion).
202
203 outlet_flanged = true; % Flanged end.
204
205
206 % Resonance
207 w_o = 2*pi*136.6; % Estimated from plot.
208
209
210 % helmholtz_diameter_cavity =
211 % helmholtz_diameter_neck = 1e-3;
212 % helmholtz_L01 = 0.82 * ( 1 - 1.33*(helmholtz_diameter_neck/helmholtz_diameter_cavity ) );
213 %
214 % epsilon = helmholtz_diameter_cavity / duct_1.length_meters;
215 % helmholtz_L02 =
216 %
217 % helmholtz_volume = 1e-3;
218 %
219 % % keyboard
220 %
221 % helmholtz_L_neck = 1e-3;
222 % Q = 2;
223 %
224 % R_A = rho0*c / Q * sqrt( L_e / ( pi*helmholtz_diameter_neck^2/4 * helmholtz_volume ) );

```

```

224 frequency_set = 0:0.1:2.5 e3;
225 nFreq = length( frequency_set );
226 TL = zeros( nFreq, 1 );
227
228
229 for frequency_index = 1:1:nFreq
230
231     f = frequency_set( frequency_index );
232
233
234     T_total = [ 1 0; 0 1 ];
235
236     T_outlet = duct_segment_transfer_matrix_flow( f, rho0, c, duct_2.length_meters, duct_2.area,
237 duct_2.Mach );
238
239     T_expansion = duct_expansion_connection_transfer_matrix( rho0, c, duct_2.area, duct_1.area,
240 duct_1.Mach );
241
242     %  $Z_A = 1j * \rho_0 * 2 * \pi * f * L_e / ( \pi * \text{helmholtz\_neck\_diameter}^2 / 4 ) - 1j * \rho_0 * c^2 / ( \text{helmholtz\_volume} * 2 * \pi * f ) + R_A$ ;
243     %  $T_{\text{Helmholtz}} = [ 1 \quad 0; \quad 1/Z_A \quad 1 ]$ ;
244
245     T_inlet = duct_segment_transfer_matrix_flow( f, rho0, c, duct_1.length_meters, duct_1.area,
246 duct_1.Mach );
247
248     %  $T_{\text{net}} = T_{\text{inlet}} * T_{\text{Helmholtz}} * T_{\text{expansion}} * T_{\text{outlet}} * T_{\text{total}}$ ;
249     T_net = T_inlet * T_expansion * T_outlet * T_total;
250
251
252     Z = open_end_impedance( f, rho0, c, duct_2.length_meters, duct_2.area, outlet_flanged );
253
254     T11 = T_net(1, 1); T12 = T_net(1, 2); T21 = T_net(2, 1); T22 = T_net(2, 2);
255     TL( frequency_index ) = 10 * log10( abs( ( T11 + duct_1.area * T12 / (rho0 * c) + (rho0 * c) *
256 T21 / duct_2.area + T22 ) / 2 )^2 );
257
258 end % End: for f = frequency_set
259
260 TL_part_d = TL;
261
262 % return
263


---


264 %% Plot
265
266 Y_LIMITS = [ 0 50 ];
267
268 figure( ); ...
269 plot( frequency_set, TL_part_b ); hold on;
270 plot( frequency_set, TL_part_c, '—' );
271 plot( frequency_set, TL_part_d, '—.' ); grid on;
272 legend( ...
273     'No Flow — Part b', ...
274     'Flow — Part c', ...
275     'Flow and Resonator — Part d', ...
276     'Location', 'SouthOutside' );
277 xlabel( 'Frequency [Hz]' ); ylabel( 'Amplitude [dB]' );
278 title( 'Transmission Loss Profiles' );
279 %
280 Ax = gca;
281 Ax.XAxis.TickLabelInterpreter = 'latex';
282 Ax.YAxis.TickLabelInterpreter = 'latex';
283 %
284 % axis( [ -50 5e3+50 Y_LIMITS ] );
285 %
286 if ( PRINT_FIGURES == 1 )
287     exportgraphics((gcf, 'Figure TL All Profiles.pdf', 'Append', true );
288 end
289
290 return
291


---


292 %% Part c
293
294 return
295
296

```

```

297 %% Part d
298
299 return
300
301 %% Clean-up
302
303 if ( ~isempty( findobj( 'Type', 'figure' ) ) )
304     monitors = get( 0, 'MonitorPositions' );
305     if ( size( monitors, 1 ) == 1 )
306         autoArrangeFigures( 2, 2, 1 );
307     elseif ( 1 < size( monitors, 1 ) )
308         autoArrangeFigures( 2, 2, 1 );
309     end
310 end
311
312
313 fprintf( 1, '\n\n*** Processing Complete ***\n\n' );
314
315
316
317 %% Reference(s)

```

5 Appendix - Matlab Code for Problem 5

```
1
2
3 %% Synopsis
4
5 % Question 4 – Intake Duct
6
7
8
9 %% To Do
10
11 % Check loss profile.
12
13
14
15 %% Note(s)
16
17 % Search for FIXMEs.
18
19
20 % In class note, the areas for the impedance might have been wrong; switch them?
21
22
23 % Use negative Mach numbers in the equations. The analysis for this case
24 % is the same as for the horn example. Inlet on the left, outlet on the
25 % right.
26
27
28 % The S in the diagram (pink area) for the Helmholtz resonator is the
29 % cross-sectional area of the resonator neck connecting it to the tube.
30
31
32 % For Lo2, use the value for a quarter-wavelength side tube.
33
34
35 % The area expansion ratio is determined using the original duct diameters.
36
37
38
39 %% Environment
40
41 close all; clear; clc;
42 % restoredefaultpath;
43
44 % addpath( genpath( '' ), '-begin' );
45 addpath( genpath( '../40 Assignments/00 Support' ), '-begin' );
46
47 % set( 0, 'DefaultFigurePosition', [ 400 400 900 400 ] ); % [ left bottom width height ]
48 set( 0, 'DefaultFigurePaperPositionMode', 'manual' );
49 set( 0, 'DefaultFigureWindowStyle', 'normal' );
50 set( 0, 'DefaultLineLineWidth', 1.5 );
51 set( 0, 'DefaultTextInterpreter', 'Latex' );
52
53 format ShortG;
54
55 pause( 1 );
56
57 PRINT_FIGURES = 0;
58
59
60
61 %% Define Constants and Anonymous Functions
62
63 rho0 = 1.21; % Density of air (kg per cubic-meter).
64 c = 343; % Speed of sound in air (meters per second).
65
66
67 h_area = @( diameter ) pi * diameter^2 / 4;
68
69
70
71 %% Define Shape
72
73 % Source
74 duct_1.diameter_meters = 0.0254; % 1 inch
75 duct_1.length_meters = 0.1524; % 6 inches
```

```

76 duct_1.area = h_area( duct_1.diameter_meters );
77
78 % Outlet
79 duct_2.diameter_meters = 0.1016; % 4 inches
80 duct_2.length_meters = 0.127; % 5 inches
81 duct_2.area = h_area( duct_2.diameter_meters );
82 %
83 % Flanged.
84
85 %
86
87 flow_rate_cubic_meters_per_second = 1.04772 / 60; % or 37 cubic-feet per minute
88
89 % return
90
91 %% Part a
92
93 % The -35 dB transmission loss dip|notch is at about 1,150 Hz.
94 f = 1150;
95
96 % Total attenuation required is +35 dB to make it 0 dB.
97
98 l = 0.0381; % meters (1.5 inches)
99 h = 0.0127; % meters (0.5 inches)
100
101 % The total attenuation of the lining from Figure 8.37 (Bies et. al., Fifth Edition),
102 %
103 %  $m = (h\_area(0.1016) - h\_area(0.0254)) / h\_area(0.0254)$ 
104 %
105 % ASSUMPTION: Cross-section area of lined duct is the same as the inlet.
106 m = 1;
107
108 k = 2*pi*1.15e3;
109
110 length_of_expansion_chamber = 0.127; % meters
111
112 %  $kL = k*length\_of\_expansion\_chamber$ 
113
114 % ASSUMPTION:
115 %
116 % Expansion ratio is  $m = 1$ ;
117 % Assume peak difference is 10 dB.
118 % Total attenuation of lining is 10 dB.
119 %
120 % The attenuation rate is about 10 dB / 0.127 meters or 78.7 dB per meter.
121
122 % return
123
124 %% Part b
125
126 l = 0.0381; % meters (1.5 inches)
127
128 h = 0.011255; % meters
129 h_validate = 0.5*sqrt( ( pi*( duct_2.diameter_meters - 2*l )^2 ) / 4 ); % Same value.
130
131
132
133 %% Part c
134
135 % The liner thickness ratio is ,
136 l / h; % 3.3852 unitless
137
138 % The normalized frequency is ,
139 ( 2 * h ) / ( 343 / f ); % 0.075471 unitless
140
141
142
143 %% Part d
144
145 % Assume the attenuation rate from Part a is 18 dB per meter.
146
147 % Use bottom, right subplot (16).
148
149 % The approximate resistivity parameter is 16.
150
151
152
153 %% Part e

```

```

154
155 % Calculate the flow resistivity.
156
157 R1 = 16 * rho0*c / l; % 1.74e5 kg / m^3*s
158
159
160
161 %% Placeholder
162
163 % % Flow present (use Mach numbers).
164 % % Helmholtz resonator in place (between lefthand duct and expansion).
165 %
166 % outlet_flanged = true; % Flanged end.
167 %
168 %
169 % % Resonance
170 % w_o = 2*pi*136.6; % Estimated from plot.
171 %
172 %
173 % % helmholtz_diameter_cavity =
174 % % helmholtz_diameter_neck = 1e-3;
175 % % helmholtz_L01 = 0.82 * ( 1 - 1.33*(helmholtz_diameter_neck/helmholtz_diameter_cavity ) );
176 % %
177 % % epsilon = helmholtz_diameter_cavity / duct_1.length_meters;
178 % % helmholtz_L02 =
179 % %
180 % % helmholtz_volume = 1e-3;
181 % %
182 % % % keyboard
183 % %
184 % % helmholtz_L_neck = 1e-3;
185 % % Q = 2;
186 % %
187 % % R_A = rho0*c / Q * sqrt( L_e / ( pi*helmholtz_diameter_neck^2/4 * helmholtz_volume ) );
188 %
189 %
190 %
191 % frequency_set = 0:0.1:2.5e3;
192 % nFreq = length( frequency_set );
193 % TL = zeros( nFreq, 1 );
194 %
195 %
196 % for frequency_index = 1:1:nFreq
197 %
198 % f = frequency_set( frequency_index );
199 %
200 %
201 % T_total = [ 1 0; 0 1 ];
202 %
203 % T_outlet = duct_segment_transfer_matrix_flow( f, rho0, c, duct_2.length_meters, duct_2.area
, duct_2.Mach );
204 %
205 % T_expansion = duct_expansion_connection_transfer_matrix( rho0, c, duct_2.area, duct_1.area,
duct_1.Mach );
206 %
207 %
208 % Z_A = 1j*rho0*2*pi*f(frequency_index) * L_e / ( pi*helmholtz_neck_diameter^2/4 ) - 1j*
rho0*c^2/(helmholtz_volume*2*pi*f(frequency_index)) + R_A;
209 % % T_Helmholtz = [ 1 0; 1/Z_A 1 ];
210 %
211 %
212 % T_inlet = duct_segment_transfer_matrix_flow( f, rho0, c, duct_1.length_meters, duct_1.area,
duct_1.Mach );
213 %
214 %
215 % T_net = T_inlet * T_Helmholtz * T_expansion * T_outlet * T_total;
216 % T_net = T_inlet * T_expansion * T_outlet * T_total;
217 %
218 %
219 % Z = open_end_impedance( f, rho0, c, duct_2.length_meters, duct_2.area, outlet_flanged );
220 %
221 % T11 = T_net(1, 1); T12 = T_net(1, 2); T21 = T_net(2, 1); T22 = T_net(2, 2);
222 % TL( frequency_index ) = 10 * log10( abs( ( T11 + duct_1.area*T12/(rho0*c) + (rho0*c
)*T21/duct_2.area + T22 ) / 2 )^2 );
223 %
224 %
225 % end % End: for f = frequency_set
226 %

```

```

227 % TL_part_d = TL;
228
229 % return
230
231 %% Plot
232
233 % Y_LIMITS = [ 0 50 ];
234 %
235 % figure( ); ...
236 %     plot( frequency_set, TL_part_d ); hold on;
237 %     % plot( frequency_set, TL_part_c, '-' );
238 %     % plot( frequency_set, TL_part_d, '-.' ); grid on;
239 %     legend( ...
240 %         'No Flow - Part b', ...
241 %         'Flow - Part c', ...
242 %         'Flow and Resonator - Part d', ...
243 %         'Location', 'SouthOutside' );
244 %     xlabel( 'Frequency [Hz]' ); ylabel( 'Transmission Loss [dB]' );
245 %     title( 'Transmission Loss Profiles' );
246 %
247 %     Ax = gca;
248 %     Ax.XAxis.TickLabelInterpreter = 'latex';
249 %     Ax.YAxis.TickLabelInterpreter = 'latex';
250 %
251 %     axis( [ -50 5e3+50 Y_LIMITS ] );
252 %
253 %     if ( PRINT_FIGURES == 1 )
254 %         exportgraphics((gcf, 'Figure TL All Profiles.pdf', 'Append', true);
255 %     end
256
257 % return
258
259 %% Clean-up
260
261 if ( ~isempty( findobj( 'Type', 'figure' ) ) )
262     monitors = get( 0, 'MonitorPositions' );
263     if ( size( monitors, 1 ) == 1 )
264         autoArrangeFigures( 2, 2, 1 );
265     elseif ( 1 < size( monitors, 1 ) )
266         autoArrangeFigures( 2, 2, 1 );
267     end
268 end
269
270
271 fprintf( 1, '\n\n*** Processing Complete ***\n\n' );
272
273
274
275 %% Reference(s)

```