```
1   function data = Min_SLM_A_Fast_44p1_kHz( pZ )
2
3   % NOTE: MIN does not require an index for analysis start point -
4   % it is determined in this code itself ( see line ~ 12 )
5
6   % pZ = Unweighted sound pressure, pa
7   % fs = Sampling frequency
8   % Frequency Weighting = Z
9   % Time Weighting = fast
10
11  fs = 44100; % Function is only valid for 44.1 kHz
12  idxStart = 1 * fs; % To allow time for filters to stabilize
13  idxStop = length( pZ );
14
15  %-------------------------------------------------------------------------
16  %  A-Weighting - Overall
17  %-------------------------------------------------------------------------
18
19  load AFilter_fs_44p1_kHz.mat B A % Load filters for A-weighting
20  pA = Compute_A_Weighting_44p1_kHz( pZ, B, A );
21
22  %-------------------------------------------------------------------------
23  %  Exponential Time Window Averaging - Overall
24  %-------------------------------------------------------------------------
25  % Time weighting must be done last since it operates on the pressure
26  % squared, not the pressure, hince a non-linear operation
27
28  spl = Compute_Fast_Time_Weighting_44p1_kHz( pA );
29
30  %-------------------------------------------------------------------------
31  % Statistical Metrics - Overall Levels
32  %-------------------------------------------------------------------------
33
34  [ minSPL, idxMinSPL ] = min( spl( idxStart : idxStop ) );  % Only search for max between "triggers"
35  idxMinSPL = idxMinSPL + idxStart; % makes idxMinSPL correct for full signal
36
37  %-------------------------------------------------------------------------
38  %  Generate One-Third Octave Band Sub-Channels
39  %-------------------------------------------------------------------------
40
41  % Load one-third octave band filters from 315 Hz to 5 kHz
42  load FOBFiltersNHTSA Hd F0 % Note we are using this version to get 315 Hz to 5 kHz
43  % Each row contains the time signal associated with a specific 1/3 OB
44
45  % Note that the value associated with the minimum is AT idxMinSPL for all
46  % 1/3 OBs, but we need earlier part of signal for filter stabilization
47  oneThirdOB_pA = Compute_One_Third_Octave_Bands_44p1_kHz_Hd( Hd, pA( 1 : idxMinSPL ) );
48
49  %-------------------------------------------------------------------------
50  %  Exponential Time Window Averaging - One-Third Octave Bands
```

```
%-------------------------------------------------------------------------
% Time weighting must be done last since it operates on the pressure
% squared, not the pressure

% Initialize OBL
oneThirdOBSPLs = zeros( size( oneThirdOB_pA ) );
% Each row is a 1/3 OB, each column is the Level at the associated time

for ink = 1 : length( Hd )
    oneThirdOBSPLs( ink, : ) = ...
        Compute_Fast_Time_Weighting_44p1_kHz( oneThirdOB_pA( ink, : ) );
end

%-------------------------------------------------------------------------
% Statistical Metrics - One-Third Octave Bands
%-------------------------------------------------------------------------

% Statistics correlating to Overall Level Statistics, e.g. max of LAfMax
OTOBsAtMin = oneThirdOBSPLs( :, end ); % Computation stopped at idxMinSPL

%-------------------------------------------------------------------------
% Format Results
%-------------------------------------------------------------------------

data.timeWeighting = 'fast';
data.frequencyWeighting = 'A';
data.fs = fs;
data.fc = F0;
data.idxSPL = idxMinSPL;

% All Data Below are A-weighted
data.maxSPL = [];
data.minSPL = round( 10 * minSPL ) / 10 ;
data.oneThirdOctaveBandLevelsAtMax = [];
data.oneThirdOctaveBandLevelsAtMin = round( 10 * OTOBsAtMin ) / 10 ;
```