ENGR 451 - Lab 3

Convolution, Part II

```
test_lab3; % initialize test_lab3 function

% Problems #1-4
x = ones(1, 15);
h = ones(1, 3);
for lc = 5:5:15
 test_lab3(x, h, lc);
end
test_lab3(x, h, 50);

% Problems #5-7
for lx = 14:16
 x = ones(1, lx);
 test_lab3(x, h, 15);
end

% Problem #8-9
test_lab3(1, 1, 1);
test_lab3(1, 1, 10);

% Problem #10-12
% load lab2 % assumes you have 'seashell.wav'in your directory
x = seashell(:)';
test_lab3(x, fir_lp, 100);
test_lab3(x, fir_lp, 200);
test_lab3(x, fir_hp, 100);
```

```
Problem #1
   Your overlap-add function is correct
   Your overlap-save function is correct
Problem #2
   Your overlap-add function is correct
   Your overlap-save function is correct
Problem #3
   Your overlap-add function is correct
   Your overlap-save function is correct
Problem #4
   Your overlap-add function is correct
   Your overlap-save function is correct
Problem #5
   Your overlap-add function is correct
   Your overlap-save function is correct
Problem #6
   Your overlap-add function is correct
   Your overlap-save function is correct
Problem #7
   Your overlap-add function is correct
   Your overlap-save function is correct
Problem #8
```

```
   Your overlap-add function is correct
   Your overlap-save function is correct
Problem #9
   Your overlap-add function is correct
   Your overlap-save function is correct
Problem #10
   Your overlap-add function is correct
     Your elapsed time is 3930.64 usecs
     which is 46.8 times Matlab's elapsed time (84.04 usecs)
   Your overlap-save function is correct
     Your elapsed time is 1847.28 usecs
     which is 13.7 times Matlab's elapsed time (135.32 usecs)
Problem #11
   Your overlap-add function is correct
     Your elapsed time is 3884.1 usecs
     which is 27.6 times Matlab's elapsed time (140.68 usecs)
   Your overlap-save function is correct
     Your elapsed time is 825.04 usecs
     which is 5.73 times Matlab's elapsed time (144 usecs)
Problem #12
   Your overlap-add function is correct
     Your elapsed time is 5998.04 usecs
     which is 34.1 times Matlab's elapsed time (176 usecs)
   Your overlap-save function is correct
     Your elapsed time is 1965.8 usecs
     which is 19.8 times Matlab's elapsed time (99.2 usecs)
```

Program Listings

```
disp(' ')
disp('--- overlap_add.m --------------------------------')
type('overlap_add')
disp('--- overlap_save.m --------------------------------')
type('overlap_save')
```

```
--- overlap_add.m --------------------------------

function y = overlap_add(x, h, lc)
% OVERLAP_ADD  Convolve x and h using overlap-add method
%              y = overlap_add(x, h, lc)
%              x and h are arrays,
%              lc is the chunk size (default 50)

% Set default chunk size if not specified
if nargin < 3
    lc = 50;
end

% Get length of x and h
lx = length(x);
lh = length(h);
```

```
% Calculate the length of the FFT required for convolution
L = lc + lh - 1;
N = 2^nextpow2(L);

% Initialize the output array with zeros
y = zeros(1, lx+lh-1);

% Start processing each chunk of the input sequence
for i = 1:ceil(lx/lc)
% Get current chunk of x
    n1 = (i-1)*lc + 1;
    n2 = min(i*lc, lx);
    chunk_x = x(n1:n2);

% Pad chunk_x with zeros if necessary
    pad_length = N - length(chunk_x);
    chunk_x = [chunk_x, zeros(1, pad_length)];

% Compute the convolution of the chunk and the filter in the frequency domain
    Y = ifft(fft(chunk_x) .* fft(h, N));

% Add the output of this chunk to the final output using overlap-add method
    n3 = (i-1)*lc + 1;
    n4 = min(i*lc+lh-1, lx+lh-1);
    y(n3:n4) = y(n3:n4) + Y(1:n4-n3+1);
end

end
--- overlap_save.m --------------------------------

function y = overlap_save(x, h, lc)
% OVERLAP_SAVE Convolve x and h using overlap-save method
% y = overlap_save(x, h, lc)
% x and h are arrays,
% lc is the chunk size (default 50)

lh = length(h);
lx = length(x);

% create array to hold final convolved squence
y = zeros(1, lx+lh-1);

% if the chunk length passed to the function is greater than the length
% of x, then reset the chunk length to the length of x. If chunk length
% is greater than 500, reset to 500 (max chunk size)
if lc > lx
    lc = lx;
elseif lc > 500
 lc = 500;
end

% take the first chunk from 1 to lc
x_chunk = x(1:lc);
y_chunk = conv(x_chunk, h);
```

```
y(1:lc+lh-1) = y_chunk;

% set the y_offset to lc
y_offset = lc;
% set the x offset to lc - length of h + 1
x_offset = lc-lh+1;

% while there are still full length chunks remaining
while x_offset + lc <= lx
    % take a chunk of x from x_offset + 1 to offset + lc
    x_chunk = x(x_offset+1:x_offset + lc);
 % convolve chunk with h
    y_chunk = conv(x_chunk, h);
 % place y_chunk in the final sequence at the location starting at
    % y_offset+1. This will overwrite the unwanted values from the
    % previous chunk's convolution (y_offset is lc-1 greater than
    % x_offset to account for this)
    y(y_offset+1:y_offset + lc) = y_chunk(lh:end);
 % increment y_offset and x_offset by the chunk length minus the
    % length of h + 1
    y_offset = y_offset + lc - lh + 1;
    x_offset = x_offset + lc - lh + 1;
end

% if there are still remaining values in x, a special chunk the length
% of the remaining elements is created, convolved with h, and placed
% into y
if lx - (x_offset + lc) < 0
    x_chunk = x(x_offset + 1: end);
    y_chunk = conv(x_chunk, h);
    y(y_offset + 1:end) = y_chunk(lh:end);
end

end
```

Published with MATLAB® R2022b