Problem Set 8:
Stereo Audio Coder

Please send back to me via NYU Classes

- A zip archive named as
  `PS08_<your name as FirstLast>.zip`
  containing
    - The Matlab scripts that implements all aspects of this problem.

Total points: 75
The assignment of points is indicated in the problem statement.

Introduction

In the previous problem set you created the Matlab functions to implement a basic audio coder, in which each channel is coded separately. In this problem set you will create two new scripts to implement Mid/Side stereo joint channel coding. Use all of your scripts from the previous problem plus two (revised) instructor-supplied scripts to create a stereo perceptual audio coder.

The instructor has supplied to you a Matlab top-level script:

```
mdct_coder.m
```

and

```
common.m
```

These are different from the scripts of the same name from the previous problem set, in that

```
mdct_coder.m
```

now has calls to two new scripts:

```
ms_encode()
ms_decode()
```

and

```
common.m
```

defines some new parameters.

It is your task in this problem set to write the new scripts:

```
ms_encode.m
ms_decode.m
```

Stereo Audio Coder (75 points)

Consult the lecture slides

L09_AuditoryPerceptionStereo
L10_BasicPerceptualCoder-2

In order to understand how to construct the scripts for M/S encode and decode.

The top level script shows the position of this processing in the entire encoder/decoder signal flow. The M/S encoding operates on the unquantized MDCT coefficients, `X`. These are optionally quantized, according to the `quant_mode` argument. If `quant_mode == 0`, then `X` is not quantized, but instead copied to `Y`. The M/S decoding operates on `Y`. Since M/S coding is perfectly invertible (in the absence of quantization), the output of the top-level script is a copy of the input signal, subject only to the computational imprecision ("noise") of the MDCT transform.

(**50 points**) The M/S encoding script has arguments as shown here:

```
      [ms_flags, X(k+1:k+Nc,:), mdct_qs_bands_hat] = ...
            ms_encode(X(k+1:k+Nc,:), mdct_qs_bands_hat,
            Xs_pow_bands, freq_band_top);
```

The only new array is

ms_flags

This has an entry for each frequency band. As noted in the lecture slides, if the spread power in the band is such that

Sum_pow/Diff_pow > 3dB

Then code as MS. You can compute this as

```
      L_pow = Xs_pow_bands(i, LEFT);
      R_pow = Xs_pow_bands(i, RIGHT);
      sum_dB = 10*log10(L_pow + R_pow);
      diff_dB = 10*log10(L_pow - R_pow)
```

and then test

```
      if ( (sum_db – diff_db) > 3 )
            %setup for Mid coding
            ms_flag(i) = MS_Coding;
            %more operations to implement
            % M = L+R   (sum)
            % S = L-R   (difference)
      else
            %setup for Side coding
            ms_flag(i) = LR_Coding;
      end
```

For MS coding, you have to copy the sum (L+R) signal for the bins in that band into the MID MDCT channel and the difference (L-R) signal for the bins in that band into the SIDE MDCT channel. For LS coding, you don't have to do anything except correctly set the ms_flag(i) entry.

The parameter values `MS_Coding, LR_Coding, LEFT, RIGHT, MID, SIDE` are defined in the new version of `common.m.`

(**25 points**) The inverse M/S matrixing is done by a call to

```
      Y(k+1:k+Nc,:) = ms_decode(Y(k+1:k+Nc,:), ms_flags,…
             freq_band_top);
```

This only has to look at the `ms_flags` array and reconstruct the output MDCT coefficient array as:

L = (M+S)/2
R = (M-S)/2

When you get the two scripts running correctly, execute the script as

```
mdct_coder('trilogy.wav', 0)
```

Upon completion, it will print

```
SNR is 258.703993 dB
```

(and some other printout which can be ignored at this point).

The fact that the SNR is equivalent to the MDCT/IMDCT analysis/synthesis indicates that the M/S encode and decode (which is invertible) is working correctly.

Also, you can un-comment this line at line 193 of `mdct_coder()`

```
%                  fprintf('%d', ms_flags); fprintf('\n');
```

To see a printout of the ms_flags array.

Finally, execute the script as

```
mdct_coder('trilogy.wav', 1)
```

Upon completion, it will print (or something close):

```
SNR is 19.578514 dB
Average bits per block for:
  QSS    588.18
  Coef  7166.25
Bits per sample:   3.79, Bit rate: 364.167 kb/s
Compression ratio:   4.22
```