Music 422 Final Project

Jatin Chowdhury, Arda Sahiner, Abhipray Sahoo

Center for Computer Research in Music and Acoustics (CCRMA)

Table of Contents

Block Switching

Block Switching: Motivation

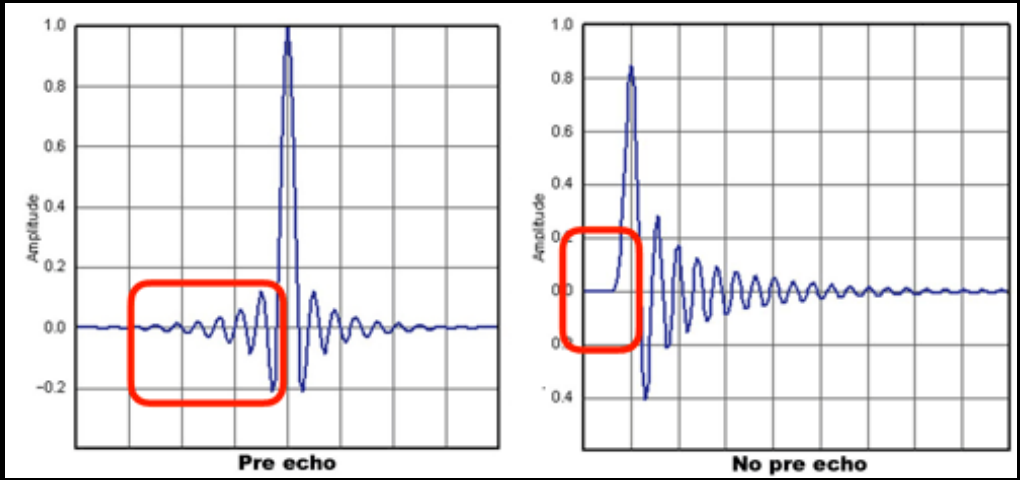
Time-frequency trade-off (Fourier uncertainty principle[1])

- Long blocks have good frequency resolution (poor time resolution)
- Short blocks have good time resolution (poor frequency resolution)

[1] Broglie, *On the Theory of Quanta*.

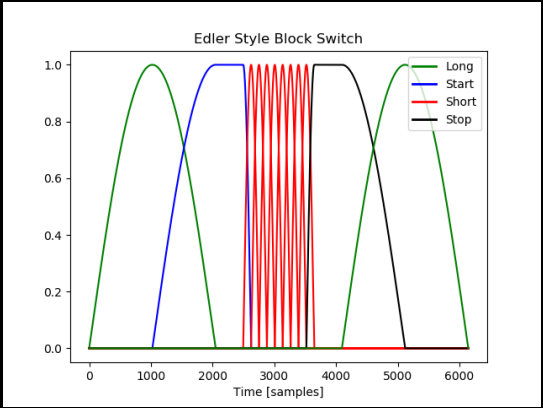
Block Switching: Motivation

How does this manifest in audio coding? “Pre-echo”



Block Switching: Motivation

Solution: Use long blocks for steady-state signal, and short blocks for transient signals.[2]



[2] Edler, “Coding of Audio Signals with Overlapping Transform and Adaptive Window Shape”.

Block Switching: Components

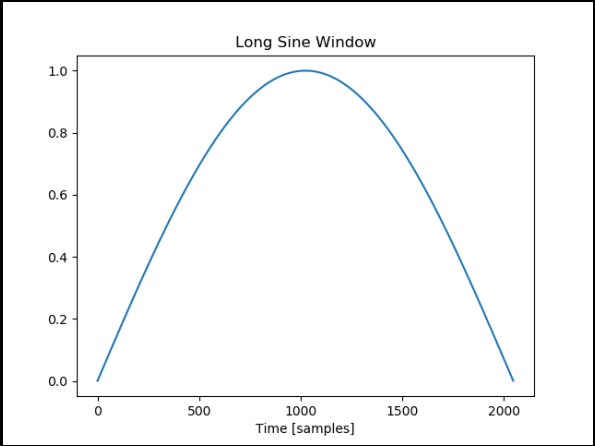
Window equations: Sine window[3]

$$w[n] = \sin\left(\left(n + \frac{1}{2}\right)\frac{\pi}{N}\right) \tag{1}$$

[3] Smith, *Spectral Audio Signal Processing*.

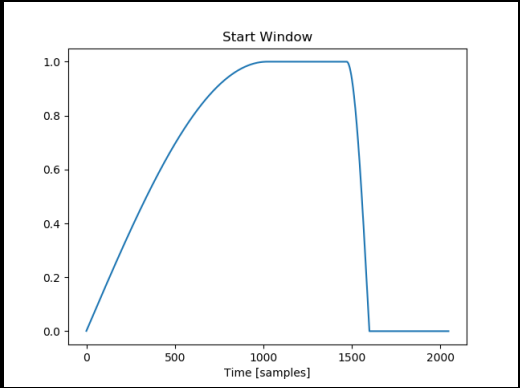
Block Switching: Components

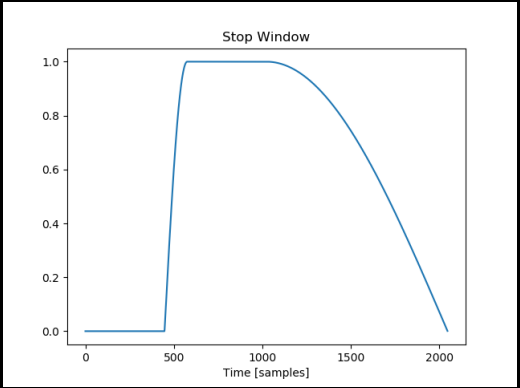
Window types: Sine window



Block Switching: Components

Window types: Transition windows





Block Switching: Components

Transient Detection: peak-to-average ratio (PAR)

$$PAR = \frac{\max(|x_1|, |x_2|, \ldots, |x_N|)}{\frac{1}{N}\sum_{i=1}^{N} |x_i|} \quad (2)$$

Block Switching: Components

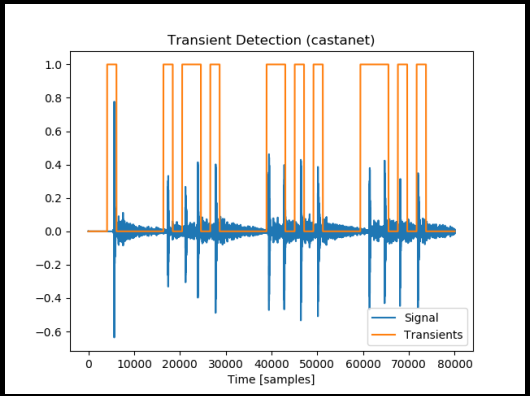
Transient Detection: modified PAR

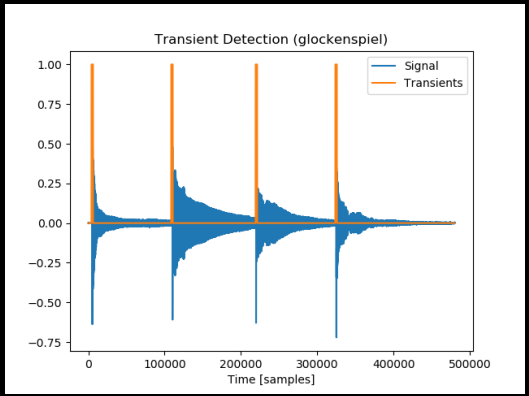
$$n_{max} = \text{argmax}(|x_1|, |x_2|, \ldots, |x_N|)$$

$$PAR = \frac{\max(|x_1|, |x_2|, \ldots, |x_N|)}{\frac{1}{n_{max}} \sum_{i=1}^{n_{max}} |x_i|} \quad (3)$$

Block Switching: Results

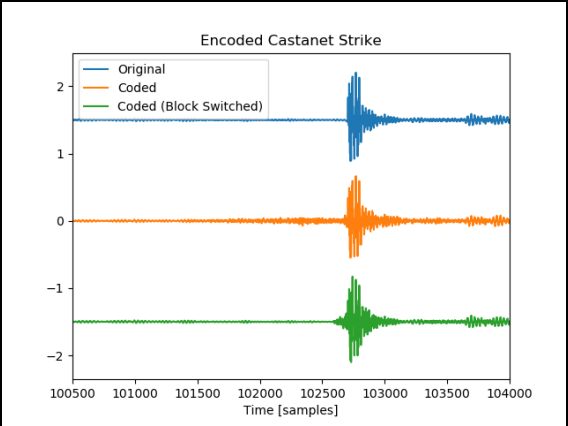
Transient Detection





Block Switching: Results

Pre-echo suppression



Spectral Band Replication

Spectral Band Replication: Motivation

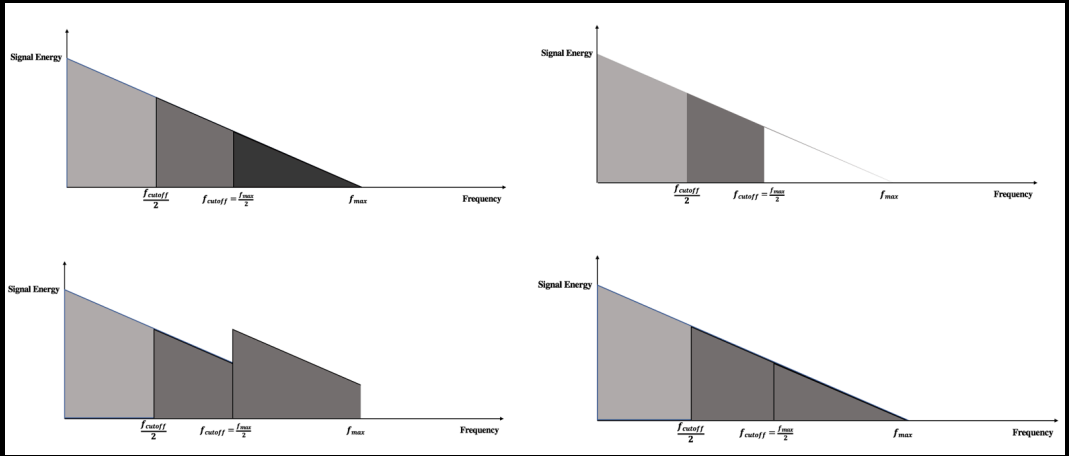
- For low bit-rates, baseline will not allocate any bits to high-frequency
  - Higher signal-to-mask ratios (SMR) and fewer lines per band at lower frequency bands
  - Output sounds low-pass filtered

- Need to represent high-frequency components better–use redundancy!
  - We know higher frequencies are correlated with lower frequencies–exploit this for better results

Spectral Band Replication: Approach

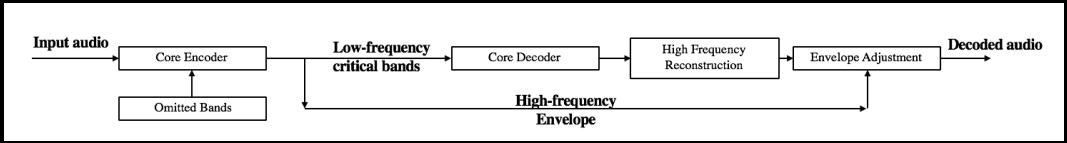
- **Encoder:** Transmit estimate of envelope of the signal at higher frequencies
  - Use DFT to represent envelope
  - For our coder, use one value per high frequency band for this envelope
- **Decoder:** Transpose lower frequency MDCT lines to higher frequency portion
  - Transpose low frequencies onto their integer multiple (e.g. 11kHz transposed to 22kHz)
  - Adjust energy by received envelope
- Only activate at low bit rates (96 kbps)

Spectral Band Replication: Illustration



Ideal model of Spectral Band Replication.

Spectral Band Replication: Block Diagram



Block Diagram for Spectral Band Replication

Spectral Band Replication: Encoder

- Set and store a certain number of critical bands as “omitted bands” to get the desired frequency cutoff (top 2 bands)
- Modify “nLines” parameter for these bands to equal 1 (we only transmit one value)
- Mantissa bits for this one value determined by optimal bit allocation

Spectral Band Replication: Decoder

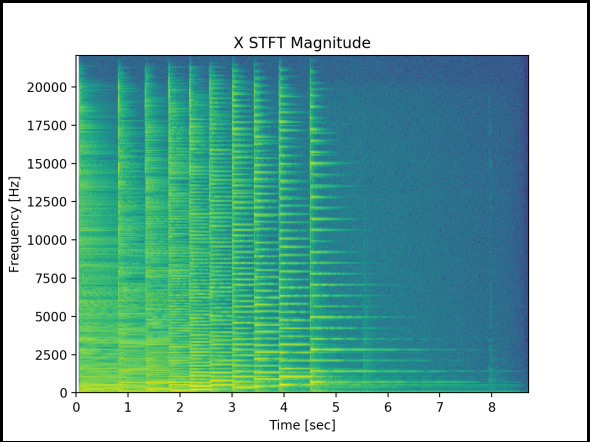
- Transpose relevant lower frequency MDCT values to integer multiple higher frequencies
  - MDCT frequencies are *affine* not linear, so frequency MDCT line k is not exactly half of frequency at $2k$
  - Use linear spline interpolation to adjust for this and provide more accurate frequency values

- Apply Gaussian filter to received envelope to smooth it out, then adjust energy of transposed frequency lines by this filtered envelope

Spectral Band Replication: Results

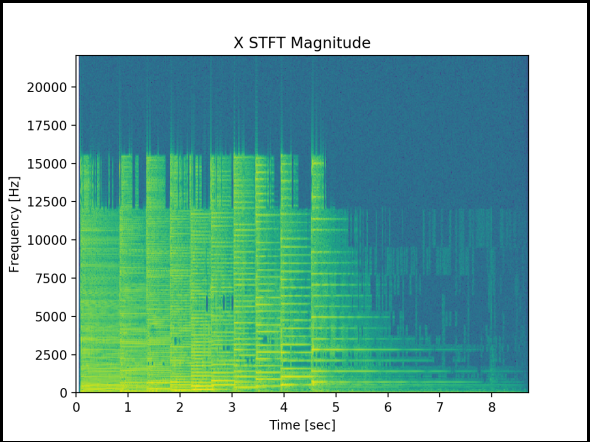
- Empirically, there is a clear improvement to our coder with the inclusion of SBR (see listening test results)
- Can visually verify these improvements in spectrum of encoded signals
- Still room for improvement: envelope is likely too coarse, could transmit multiple values per critical band instead of just one. This value could even be chosen adaptively!

Spectral Band Replication: Example



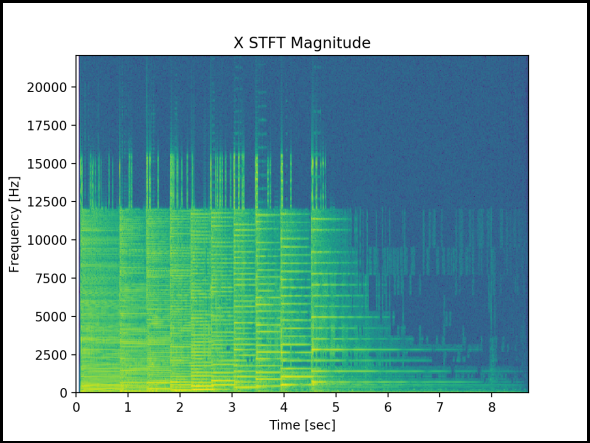
Harpsichord 16-bit PCM STFT

Spectral Band Replication: Example



Harpsichord, output of our coder at 96 kbps, no SBR

Spectral Band Replication: Example



Harpsichord, output of our coder at 96 kbps, with SBR

Gain-Shape Quantization

Gain-Shape Quantization: Motivation

The most important advice I got was “always make sure the shape of the energy spectrum is preserved”
-Jean-Marc Valin, creator of CELT. Advice from the designer of Ogg Vorbis[4]

- Explicitly encode energy or gain for each band with scalar quantization
- Encode the unit-norm band using pyramid vector quantization (PVQ)

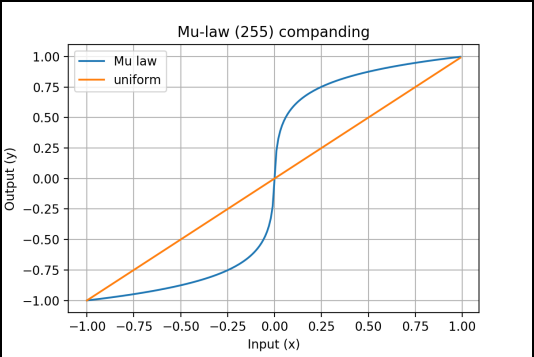
[4] *jmvalin | Recent Entries.*

Gain-Shape Quantization: Energy

For a band of MDCT Coefficients x, the energy or gain is the L2 norm:

$$G = ||x||_2$$

Uniform quantize after mu-law function transformation.

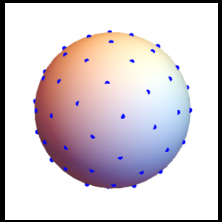


Gain-Shape Quantization: Shape

For a band of MDCT Coefficients x, shape is $S = \frac{x}{G} = \frac{x}{||x||_2}$

- Vector lies on N-dimensional hyper-sphere

- Easy codebook is to uniformly distribute points on the hypersphere. This is mathematically impossible for N > 3. An approximation is to use the Pyramid Vector Quantization.

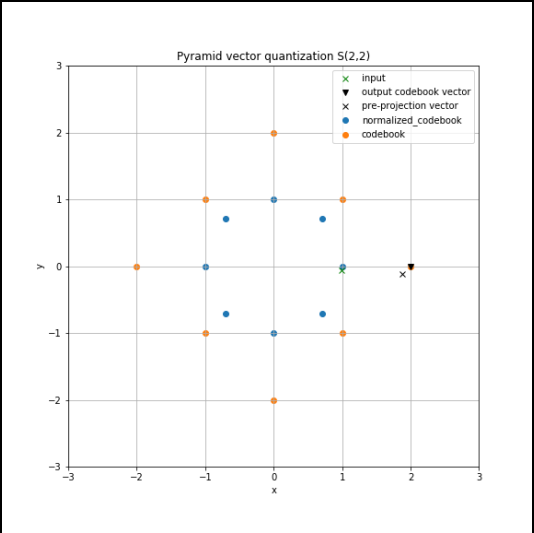


Gain-Shape Quantization: Pyramid VQ

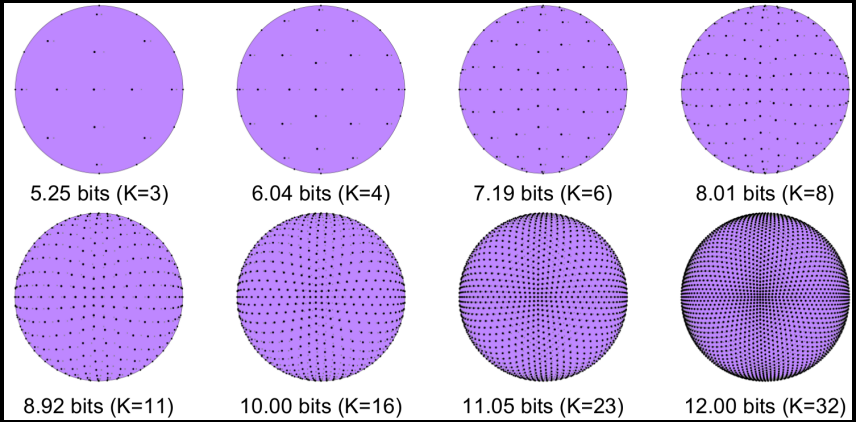
$$S(L,K) = \left\{ \frac{x}{||x||_2} : x \in \{\mathbb{Z}^n : \sum |x_i| = K\} \right\}$$

- Algebraic codebook with fast lookup for finding closest codebook vector
- Enumeration algorithm for mapping codebook vector to codebook index

Gain-Shape Quantization: Pyramid VQ



Gain-Shape Quantization: Bit allocation



Codebooks for different values of K and same dimension L

Gain-Shape Quantization: Band Splitting

If bits allocated to a vector is greater than 32 bits, split into two halves and joint-encode with MS scheme.

$$M = \frac{(x_l + x_r)}{2} \quad S = \frac{(x_l - x_r)}{2} \quad \theta = \arctan \frac{||S||}{||M||}$$

PVQ normalized $m = M/||M||$ and $s = S/||S||$. θ captures energy distribution between M and S in a variable. Uniformly quantized.

$$x_l = \frac{\hat{m}\cos(\hat{\theta}) + \hat{s}\sin\hat{\theta}}{\sqrt{2}} \quad x_r = \frac{\hat{m}\cos(\hat{\theta}) - \hat{s}\sin\hat{\theta}}{\sqrt{2}} \tag{4}$$

Results

Results: Audio

Glockenspiel (128 kbps)

Reference: Play
Baseline: Play
Coder: Play

Glockenspiel (96 kbps)

Reference: Play
Baseline: Play
Coder: Play

Results: Audio

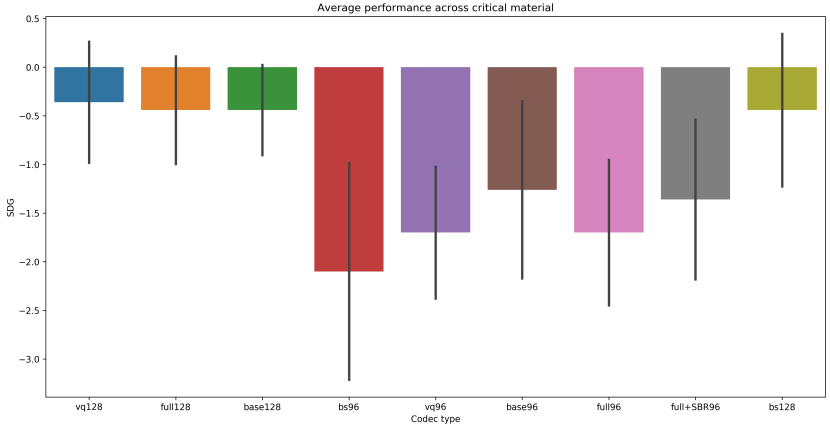
Speech (128 kbps)

Reference: Play
Baseline: Play
Coder: Play

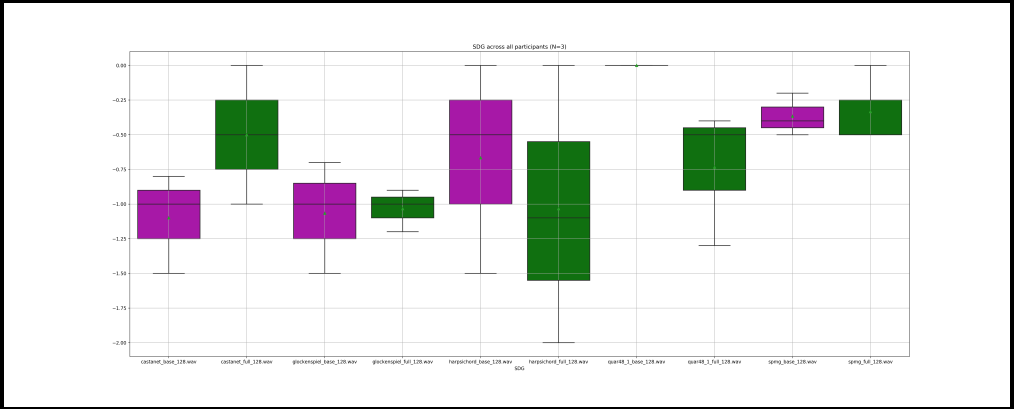
Speech (96 kbps)

Reference: Play
Baseline: Play
Coder: Play

Results: Listening Tests



Results: Listening Tests



Results

Analysis:

- At 128 kbps all versions of our coder performs comparable to the baseline.
- At 96 kbps only our full coder with SBR switched on performs as well as the baseline.
- For some material our coder performs better, for other material the baseline performs better.

Results

Future improvements:

- Block switching: Better transient detection, compare with AC-2 style block switching
- SBR: Use finer spectral resolution
- Gain-shape quantization: Better handling of edge cases.

Business Plan

- We would need to supply a new decoder.

- In order to convince people to use our coder, we could to point to clear improvements in audio quality, including results from the listening tests comparing our coder to the baseline, as well as better objective test results such as PEAQ scores.

- Target market: cloud storage services. Demonstrable effect of our advanced coding methods for increased compression rates for the same quality.

Thanks!