

CM613—Multimedia Storage & Retrieval

Compression and streaming: Serving, shrinking, and otherwise messing about with perfectly good audio files

Men believe in the truth of all that is seen to be strongly believed in.

—Nietzsche

Loudness, power, sampling error, and signal-to-noise ratios

In the last chapter we discussed the deployment of audio assets in various formats, such as MP3, RealAudio, etc. In this chapter we will explore some of the technologies available for compressing and streaming audio and the issues that arise from that. Before we do that, we need quickly to cover a few more points related to audio signals and sampled signals.

Loudness and power

Sounds have different loudness because they press on your eardrum with different pressures. The pressure waves in a sound have different amounts of *power*. A signal with more power has more force to push against your eardrum. Sound is carried electrically (e.g. through a microphone cable) by varying a voltage; the higher the voltage the greater the amplitude of the signal. Physics tells us that the power of a sound is proportional to the square of its amplitude (or voltage).

Sampling error and noise

Remember that the audio CD format uses a sampling rate of 44.1 KHz and a resolution of 16 bits. This means that the voltage of the sampled signal can be represented by values in the range -32768 to $+32767$ (or 0 to 65535 if no sign bit is used). But the sampled signal is a continuously varying analogue wave whose instantaneous voltage can be any real value within the range. Therefore, when sampling we are rounding the amplitude of the signal to the nearest integer (called *quantisation*), and so we are introducing errors. Because we're only rounding to the nearest integer the maximum possible error is 0.5. In a 16-bit system 0.5 is 2^{-16} as loud as the loudest possible sample value. As power is related to the square of the amplitude, then the error has a power 2^{-32} as loud as the loudest possible signal (Kientzle, 1997, page 21). Thus, the ratio between the power of the loudest signal and the noise is $2^{32}:1$. In decibels this is $10 \log_{10}(2^{32}) \approx 96.3$ dB¹. This figure of 96 dB is called

¹ Actually, to reduce the distortion that is introduced by the sampling process (a sine wave could be represented by only two samples per wave length which would produce a square wave shape) a low-level noise, or *dither* is introduced into the signal before it goes to the ADC. The dither has a noise level around 96 dB below the maximum level, thus the maximum SNR of a 16 bit system is 96 dB. However, the spectrum of the dither is usually manipulated to reduce its audibility by placing most of its energy in the 15-20 KHz band

the *signal-to-noise ratio* (SNR) and it gives a measure of the relative loudness of noise in the signal. By contrast, an 8-bit sampling system (common on early computers) has an SNR of approximately 48 dB ($10 \log_{10}(2^{16})$)². From this you can see there is a rough rule of thumb that adding 1 bit to your sampling resolution gives you an extra 6 dB of SNR. Recall that the threshold of pain is 120 dB to 130 dB and you can see that, good as it is, the CD format does not have a dynamic range as wide as the human auditory system. On the other hand, a 20-bit sampler gives an SNR of approximately 120 dB which is a much better match and allows a more realistic reproduction of amplitude.

It is important to realise that SNR is dependent upon context. By definition, the SNR is calculated by comparing the size of the sampling error with the loudest possible signal. However, rarely will the loudest possible signal occur; in fact, most samples will be quite small values and the differences between them will also be small. When the input signal is relatively quiet, then the noise is much more noticeable.

In Chapter 5 Table 2 (page 84) listed some of the more common computer audio storage formats. In this chapter we will expand upon those and look at the differences between uncompressed and compressed audio. The standard for uncompressed audio is *pulse code modulation* (PCM). In a PCM stream, the amplitudes of the sampled signal are converted to binary values. These values are stored in sequence and the original sound can be reproduced by sending samples in turn to a digital-to-analogue converter. Thus, uncompressed PCM audio has a very simple coding system: each sample value represents the amplitude of the signal at a point in time. Uncompressed PCM audio is typically stored in .WAV (wave) files. However, the WAV format is a very flexible standard and supports a range of compressed and uncompressed coding systems. In fact, just under 100 compression codes have been registered with Microsoft for use in .WAV files. However, many of the codes are variations on a common theme, so the main decompression codes that can be stored in a .WAV file are ADPCM, A-Law, μ -Law, DPCM and MPEG (see below).

Compression

You should already be familiar with the basic ideas involved in data compression. Lossless algorithms allow the compression of a data stream that, when uncompressed again results in a signal that is bit for bit identical to the original. Programs like Winzip use lossless compression very effectively on word processor documents and other files. Lossless algorithms work by examining the data for repeating patterns. The pattern is then stored once and every subsequent occurrence is replaced by a code to identify the pattern. Lempel-Ziv-Welch (LZW) is a popular lossless compression technique that looks for long sequences of bytes that appear in several places. LZW works well on text documents owing to the repetition of words and phrases.

However, audio files do not have many repeated sequences because of the random background noise in the signal:

Any analog process contains a small amount of random error. When you record an analog sound, this random error shows up in the least significant bits of your sound files. Because it's random, it tends to prevent exact patterns from being identified (Kientzle, 1997).

where the ear is not very sensitive. This raises the effective level of the SNR and the dynamic range to above 96 dB (Moore, 1997).

² This is worse than a standard audio tape deck that has an inherent SNR of 55 dB which itself is considered too low for hi-fi reproduction. Noise reduction systems such as Dolby B and Dolby C boost the SNR of a compact cassette to about 65 dB and 73 dB respectively.

Table I shows the results of applying lossless compression to two audio files (using WinZip). The first file, *girl2.wav* is a piece of sampled guitar music. Notice that the compressed file (*girl2.zip*) is hardly smaller than the original. Now look at *528 Hz.wav*. This file contains a sine wave generated digitally by Cool Edit Pro. Because it's a single repeating sine wave there is much more repetition in the signal, hence the much smaller zip file (369 Kb, a compression ratio of 2.33:1³). Of course, apart from libraries of test tones very few audio files will consist of unchanging signals so you will not meet with any great success.

Table I Lossless compression on audio files

| Uncompressed | | Compressed | |
|-------------------|--------|-------------------|--------|
| <i>Girl2.wav</i> | 594 Kb | <i>girl2.zip</i> | 566 Kb |
| <i>528 Hz.wav</i> | 862 Kb | <i>528 Hz.zip</i> | 369 Kb |

Other lossless techniques (e.g. Huffman) look for byte values, or pairs of bytes, that occur more often than others and again replace these values by shorter codes. These techniques work to a limited extent on audio because sound files have an uneven distribution of values. But these techniques only give modest compression of sound. In addition, the compression is not uniform. Many applications (e.g. internet streaming) require the signal to be transferred at a constant rate with uniform packet size. For instance, a modem can transfer 4,000-5,000 bytes per second at typical connection speeds. To stream audio over such a connection each second of audio data must fit into 4,000 bytes, but this cannot be guaranteed by lossless compression techniques as some parts of the signal will be more compressed than others. Below are short descriptions of some common audio compression techniques.

Differential PCM

With a reasonably large sampling rate the differences between individual samples are likely to be quite small. If the differences are small then they need fewer bits to store them. So the sound could be stored using fewer bits per sample by storing the differences between successive samples rather than the samples themselves (Kientzle, 1997, page 95). This coding system is called *differential PCM* (DPCM). Combining a non-linear scheme with DPCM gives reasonable compression.

For example, consider reducing 8-bit PCM data to a series of 4-bit differences. Assume that three of the samples in the signal have the values 17, 28, and 30. The differences are 11 and 2. But a 4-bit system only allows values between -8 and +7 (1000...0111 in two's complement). Thus 11 is an overflow value. If we simply truncate, or clip, the value down to 7 and store the differences as 7 and 2, then the decompressor would produce the sample values 17, 24, and 26. However, if the compressor uses the difference between the next actual sample and the previous decoded sample it would code the differences as 7 and 6, and the decompressor would produce 17, 24, 30 which has only sample in error⁴.

³ Entirely by coincidence, this is nearly the same as a popular wide-screen movie aspect ratio (2.35:1).

⁴ First sample is 17. Difference between it and next sample, 28, is 11, which is clipped to give 7. If we now decode this sample, we get 24 (11 + 7) and the difference between 24 and the next actual sample (30) is 6. So, the DPCM compressor decodes its own output as it goes along in order to balance out the errors.

Predictor-based compression

Many techniques attempt to predict the next sample value by examining data that has gone before. The reason for using prediction is that if you can predict what comes next, then you don't need to store the value because the decompressor uses the same scheme to generate the predicted value. A simple scheme is to compare the predicted value with the actual value and if they match output a 1 otherwise output a 0 followed by the actual value. If the prediction algorithm is good then the output will be highly compressed (1 bit for every 16). Of course, it's hard to maintain predictive accuracy. Most prediction-based schemes output the difference between the prediction and the actual value. If the predictions are reasonably good, then the differences will be small and hence so is the output. You can see, therefore, that DPCM is a predictive technique where the prediction is that a sample is identical to its predecessor: remember, DPCM simply outputs differences.

Adaptive DPCM

Adaptive differential PCM (ADPCM) is a predictor-based compression technique. The algorithm outputs predicted differences based on the set of previous samples. If the prediction is accurate then the difference between the actual and predicted samples will have a lower variance than the actual samples, and will be accurately quantised with fewer bits.

The Interactive Multimedia Association (IMA) has defined its own ADPCM standard. IMA ADPCM encodes an audio stream into packets. Each packet contains a header (2 to 8 bytes) with state information and a set of 4-bit compressed samples. Each 4-bit code represents the difference between two successive 16-bit uncompressed PCM samples. The packet arrangement means that the stream can be decoded in the middle because each packet contains the state information necessary to seed the decompression algorithm. The standard does not specify how to store stereo data and so the Apple and Microsoft implementations store different information. The Microsoft ADPCM doesn't specify packet size or format. Instead it has one packet format for mono files and another for stereo files. Two fields in the .WAV file header specify the size of each packet. The Apple ADPCM system uses the same packet size for mono and stereo.

Sub-band coding

Because they have fewer cycles per second, low-frequency sounds produce lots of small differences whilst higher-frequency sounds produce larger differences between samples. Differential encoding systems can be improved by dividing the sound into a series of frequency ranges (*sub-bands*) and compress them individually. This way, high-frequency bands can be compressed in an optimal way, and lower frequency bands can be compressed in ways more suitable to their larger differences.

With a knowledge of psycho acoustics, even more gains can be made. The human auditory system is more sensitive to some frequency ranges than others. Therefore, those sensitive sub-bands can be copied accurately, whilst those to which we are less sensitive can be more highly compressed. Because very low frequencies are non-directional (it's hard to hear where they're coming from) they can be converted from stereo to mono. In fact, some frequencies can be dropped altogether without their loss being noticed. These techniques can offer compression ratios of 10:1 or even 20:1. For audio, this is considered to be a very high compression rate. Compare this to video which can be satisfactorily compressed at ratios of up to 70:1. This suggests that the information content of motion video is much less than that of audio.

MPEG audio, Dolby AC-2 and AC-3, Mini Discs, and RealAudio all use sub-band compression techniques in their algorithms.

Compressing speech

Musical sound has little silence in it. Whilst the various instruments in a piece of music will have many silences in their parts, the orchestration of the music means that the overall auditory stream is, more or less, a continuous one. However, a single spoken voice contains many silences and pauses. A speech stream can be compressed by up to 50% simply by identifying the silences and replacing them with much smaller duration codes.

Checkpointing

Prediction-based compressors make their predictions on a knowledge of prior data. Likewise, the decompression system must have complete knowledge of the previous decisions in order to make a correct decompression judgement. This requires that decompression always begins at the start of a signal. However, it is often necessary to start decompression part way through a signal (perhaps you want to play a sound file from the middle), and if the signal is being sent over a streamed network connection, then breaks in transmission may occur requiring back tracking, or even lost packets. In these circumstances it would be impossible to correctly decompress a signal as the algorithm state information will have been lost. A solution is for the compressor periodically to insert *checkpoints* into the signal. A checkpoint might be a block of uncompressed data that enables the decompression algorithm to restart from scratch (this technique is used by Apple's Sound Manager program). The IMA ADPCM system dumps the compressor's internal state data into checkpoints so that the decompressor can reset itself if necessary.

Non-linear coding

A high sampling resolution gives a larger signal-to-noise ration and a wider dynamic range. Reducing the sampling rate from, say 16 bits to 8 bits, halves the storage requirements of the sound, but also reduces the dynamic range from 96 dB to 48 dB. As the dynamic range governs the difference between the quietest and the loudest sounds, reducing it lowers the accuracy of the sample. Standard PCM coding is *linear* which means that a value of 50 represents twice the amplitude of a value of 25. In an 8-bit system there are only 256 possible sample values which means that any sound less than $1/256^{\text{th}}$ of the loudest possible sound will disappear.

Linear coding systems do not take account of how the human auditory system works. Our hearing is relatively insensitive to small errors and changes in loud sounds but is very sensitive to small changes in quiet sounds. Linear PCM, by definition, gives the same weight to differences regardless of their amplitude. Because of its simplicity linear PCM is the easiest format to use when editing and manipulating sound. But as a storage format it is wasteful. The dynamic range can be increased, and the loss of quieter sounds reduced by switching to a *non-linear* PCM system. In a non-linear system, a value of 1 might represent much less than $1/50$ of the intensity represented by a value of 50. Thus, non-linear systems use more bits for quiet sounds and fewer bits for the very loud sounds.

Logarithmic compression

The two most common non-linear audio formats, μ -Law and A-Law, use logarithmic compression techniques to convert linear-coded PCM samples into 8-bit codes. The systems work well because they provide greater accuracy for small (quiet) samples which form the bulk of any audio signal. In addition, the human auditory system has an approximately logarithmic response which means that a change in a quiet sound is more noticeable than the same size change in a louder sound. So,

logarithmic encoding provides the highest accuracy where it is most audible. The dynamic ranges of μ -Law and A-Law are 14 bits and 13 bits respectively (or 84 dB and 78dB). Though this is a loss compared to the 96 dB attainable with linear PCM, it is a large increase on the 48 dB available with linear 8-bit PCM data.

Logarithmic encoding systems have interesting side effects on the SNR. In a linear system the sample error is always less than 0.5, regardless of the signal's amplitude, meaning that the SNR decreases with amplitude. In a logarithmic system the errors are larger for large samples/amplitudes but small for quieter sounds. As the signal power increases, so does the error size, and so in logarithmic encoding the SNR is reasonably constant!

Perceptual coding

Despite their good results, DPCM, ADPCM, and the logarithmic techniques do not yield high-enough compression ratios to allow high-quality audio to be delivered in the limits and file size limitations demanded by multimedia and internet applications. *Perceptual coding* techniques use knowledge about the masking properties of the human auditory system to give convincing compression at greatly reduced bit rates. Linear PCM sampling tries to capture audio as it is, whilst perceptual coding attempts to capture audio as it sounds (Sellars, 2000). In fact, perceptual coders introduce significant amounts of noise and distortion into the bitstream, but these artefacts are inaudible because they are *masked* by the intended, or desired, signal. The MiniDisc ATRAC system, digital compact cassette (DCC), and MPEG audio all use perceptual coding in their compression algorithms. Essentially, the perceptual coders discard all the information from an audio signal that is not detectable by our ears.

Masking

How can part of an audio signal be inaudible? A process known as *masking* explains how this happens. Because the human auditory system is adaptive it responds to and process sounds according to the context in which they occur. For example, if you have your car radio set to a comfortable volume when driving along at 30 mph, you find that you need to turn it up when you approach motorways speeds. This is because the engine and road noise become loud enough that they mask the signal from the radio rendering it (at least partially) inaudible. A sudden hand clap in a quiet room sounds very loud. But if the clap came immediately after an even louder sound (such as a gun shot) it would seem much quieter. So, when two sounds occur simultaneously (or near simultaneously) one may be masked by the other. Masking has a standard definition given by the American Standards Association (1960):

1. The process by which the threshold of audibility for one sound is raised by the presence of another (masking) sound
2. The amount by which the threshold of audibility of a sound is raised by the presences of another (masking) sound. The unit customarily used is the decibel (quoted from Moore, 1997 page 89)

Perceptual coding takes advantage of the masking effects of our auditory systems to assign fewer bits to masked frequencies (removing them altogether would actually be noticeable). Using fewer bits for the masked elements reduces the storage requirements.

Perceptual coders filter the input into a number of frequency bands (usually 32). For a typical 44.1 KHz PCM stream this would create frequency bands approximately 625 Hz wide. The sample

rate for each band is just over twice the width of the frequency band (remember Nyquist?). Only five or six bits are needed to code the samples of each band because the noise/distortion introduced by the low resolution is not audible provided its level is more than 30 dB below the level of the original signal and its spectrum overlaps that of the signal (Moore, 1997, pp. 323-324). The noise and distortion are filtered out at the stage the compressed signal is converted back to analogue. The sound is divided into blocks of 1.5-12 ms duration. For each band and each block a scaling factor is computed that codes the overall level in that band and block.

MPEG audio

The popular MPEG, or MP3 file format is an implementation of the MPEG-I Layer 3 and MPEG-2 Layer 3 standards. Do not confuse this with the MPEG-3 standard which was designed to support high definition television. The origins of MPEG audio go back to the EUREKA project of 1987 which began on perceptual audio coding for digital audio broadcasting (Fraunhofer, 1998a). An encoding algorithm was standardized as ISO-MPEG Audio Layer-3 (IS 11172-3 and IS 13818-3). There are five parts to the MPEG standard (see Table 2).

Table 2 MPEG standard (from Kientzle, 1997)

| Layers of the MPEG-I Standard | | |
|-------------------------------|-------------|--|
| Part | Name | Description |
| 1 | Systems | deals with combining multiple video and audio streams |
| 2 | Video | Deals with image compression |
| 3 | Audio | Deals with compression of 1- and 2-channel audio (mono and stereo) |
| 4 | Conformance | Tests for implementation conformance |
| 5 | Software | A reference implementation of the standard. |

MPEG Audio–Layer 3 (from now on, we’ll call it MP3) specifies the format of a legal bitstream and how to decode it, but does not specify a compression technique. In fact, the audio part of the MPEG standard is divided into three *layers*. Each layer provides greater quality than the one before. The simplest layer (layer 1) is suited to data that can be transferred quickly but computation speed is limited. Layer 3 provides the highest quality but is computationally more expensive. MP3 achieves high quality with low bit rates by making use of stereo effects and by limiting the bandwidth of the signal. MPEG Layer-3 is the most powerful layer and for a given sound quality gives the lowest bitrate, or for a given bitrate delivers the highest sound quality.

Table 3 MPEG audio layers and data reduction (from Fraunhofer, 1998a)

| Compression ratio | Method |
|-------------------|---|
| 1:4 | Layer 1 (corresponds to 384 kbps for a stereo signal) |
| 1:6...1:8 | Layer 2 (corresponds to 256...192 kbps for a stereo signal) |
| 1:10...1:12 | Layer 3 (corresponds to 128...112 kbps for a stereo signal) |

An MP3 bitstream specifies the frequency content of an audio signal and how that content varies over time. The standard sets out how to encode the information and how a decoder can construct linear PCM samples from the bitstream. Because the compression algorithm is not specified, a range of techniques can be used to ensure optimal compression/audio quality for different types of audio signals. Orchestral music with its wide dynamic range would benefit from a different compression algorithm from speech, or even pop music with its smaller dynamic range.

Recall that CD-quality PCM audio requires approximately 10 Mb/min of storage (1,411,200 bits per second) which, on a 56 kbs modem, would take around 20 minutes to download. The MP3 format uses much lower bit rates (typically 64,000 to 128,000 bits per second) to achieve high compression factors. The bit rate is the main parameter affecting the quality of an MP3 file. Generally speaking, the higher the bit rate, the higher the audio quality. Compression ratio is a fairly vague measure of quality and so *bit rate* is normally used as a measure of compression strength. Bit rate is simply the number of bits taken up by one second of audio in the compressed signal (or bitstream). Thus, uncompressed 16-bit 44.1 KHz PCM data has a bit rate of 1,411,200 bps (172 Kbytes), whilst a 128 bps MP3 file has a bit rate of 128 kbs (15.6 Kbytes)⁵. MP3 bit rates of 8000 to 18,000 only support monaural sound. Bit rates of 18,000 to 96,000 can support MS & IS stereo modes. In IS (*intensity stereo*) mode high-frequency parts of the signal are downmixed to mono and transmitted with direction information. This mode loses phase information and is not suitable for high-quality encoding. In MS (*mid/side*) stereo mode, the encoder uses correlations between both channels. The signal is matrixed into a sum (mid) and difference (side) signal. This mode does not destroy phase information. Bit rates of 96,000 to 192,000 support MS mode, whilst signals with bit rates between 192,000 and 256,000 support stereo mode. In stereo mode the encoder makes no use of any correlations between the two channels (Fraunhofer, 1998b). The MP3 bitstream comprises frames of compressed data. Each frame has a header that defines the data format. This header information is needed by the decoder in order to know how to decompress the signal.

Table 4 MP3 sound quality (from Fraunhofer, 1998a)

| Quality | Bandwidth | Mode | Bit rate | Compression ratio |
|------------------------------|-----------|--------|---------------|-------------------|
| Telephone | 2.5 kHz | mono | 8 kbps | 96:1 |
| Better than short wave radio | 4.5 kHz | mono | 16 kbps | 48:1 |
| Better than AM radio | 7.5 kHz | mono | 32 kbps | 24:1 |
| Comparable to FM radio | 11 kHz | stereo | 56...64 kbps | 26...24:1 |
| Near-CD | 15 kHz | stereo | 96 kbps | 16:1 |
| (Almost) CD | >15 kHz | stereo | 112...128kbps | 14...12:1 |

⁵ $1,411,200 \div 8 = 176,400$ bytes. Dividing by 1,024 gives 172.26 Kb. 128000 bits = 16000 bytes = 15.6 Kb. Actually, an audio CD uses even more than 176 Kb/s as extra bits are used for keeping track of time and for error correction to compensate for missing bits caused by faults on the compact disc itself (see Moore, 1997).

ATRAC

ATRAC (adaptive transform acoustic coding)⁶ is the compression system used for MiniDiscs. It achieves compression of 5:1 and, like MP3, uses perceptual coding and sub-band coding techniques. Only three sub-bands are used in ATRAC and the block lengths are variable.

Streaming audio

So far we have dealt exclusively with audio files. However, multimedia applications often require audio and video content to be served in real time across the network. For example, an internet radio broadcast must be transmitted 'live' as it does not make sense to wait until the whole programme has been broadcast before downloading it as a single file (though this is useful for archival purposes). *Streaming* is the process of serving audio (and video) signals in small slices at a continuous rate. The audio stream is sent by the transmitter and is then received by the client and decoded on the fly. Streaming thus allows a sound file to be listened to within a few seconds of starting to play it. As long as the data can be downloaded as fast as the player is using it up, then a smooth signal is heard. When using a low-bandwidth connection (such as a modem) or during times of network congestion, the stream rate can become too slow or interrupted leading to noticeable gaps and glitches in the playback. The quality of streamed playback is dependent upon the bandwidth of the stream.

There are various streaming formats available to use including RealAudio (used by many internet radio stations), Windows Media Audio/Active Streaming Format, and Apple's Quicktime format. Producing an audio stream is relatively straightforward and typically requires the creation of a streamed version of the file, and sometimes an extra header, or descriptor, file. Free programs are available for the transformation of audio files into particular streaming formats. Real Networks offer a free version of their Helix Producer⁷ while Microsoft provide a comprehensive set of tools to support their media format, including Windows Media Encoder⁸.

When encoding a stream you must choose what bandwidth/bit rates to support. The lower the bandwidth the more accessible the stream will be but the lower the quality. To avoid compromise, many web sites provide multiple versions of the stream encoded at different quality levels and allow the user to specify the bandwidth of their internet connection in order to get the best possible stream quality. To stream live broadcasts you simply need a media producer that supports live feeds.

For some example streams using the Microsoft Active Streaming Format, visit <http://computing.unn.ac.uk/staff/cgpyl/music!.htm>.

⁶ See http://www.minidisc.org/aes_atrac.html for a good summary description of the technique.

⁷ <http://www.realnworks.com/products/producer/basic.html>

⁸ <http://www.microsoft.com/windows/windowsmedia/technologies.aspx>

References

- American Standards Association (1960). *Acoustical Terminology SI, I-1960*. New York: American Standards Association
- Fraunhofer. (1998a). Retrieved 28/02/2003, from the World Wide Web:
<http://www.iis.fraunhofer.de/amm/techinf/layer3/index.html>
- Fraunhofer (1998b), "MP3EnvV3.1: Next Generation High-End MPEG Layer-3 Encoding," Fraunhofer Institute for Integrated Circuits
- Kientzle, T. (1997). *A Programmer's Guide to Sound*. Reading, Massachusetts: Addison-Wesley. ISBN 0-201-41972-6.
- Moore, B. C. J. (1997). *An Introduction to the Psychology of Hearing*. London: Academic Press. ISBN 0-12-505627-3.
- Sellars, P. (2000). "Behind the Mas: Perceptual Coding-How MP3 Compression Works." *Sound on Sound* **2000**(May).