PTC: Perceptual Transform Coding for Bandwidth Reduction of Speech in the Analog Domain, Part 2

Our exploration continues with an experimental codec, downloadable .WAV files and examination of the coded signals.

By Doug Smith, KF6DX

[Note: The author apologizes for the length of time between Parts 1 and 2. He will, no doubt, blame it on the editor—Ed.]

Recapitulation of Part 1

In Part 1 of this article,[1] properties of human speech and hearing were examined for quantization effects that may be exploited in bandwidth-reduction schemes. When we left off, we were ready to choose methods of efficiently representing speech signals. Here in Part 2, we'll take a look at the actual implementation of a PTC codec, beginning with principles of subband decomposition. We'll finish with tests and analysis of the resulting frequency-compandored signals.

Subband Decomposition

Choosing how to represent a signal is an important problem in DSP—just as important as how a signal is manipulated. In this section, I'll show how *subband decomposition* helps meet the requirement for frequency resolution proportional to frequency (see Part 1) while minimizing the computational burden of analysis and synthesis operations. This technique is moderately well documented in contemporary literature but is poorly understood in general. Most of the texts currently available were written by mathematicians for other mathematicians. That often results in stuff that's too insensible and that rarely comes close to the goal of explaining things clearly. I went through a lot of brain wracking doing this and I don't expect you to get it right away. Let me know if you have questions.

Mathematical language can be concise and elegant; it is also frequently ambiguous and sometimes reveals little of its underlying usefulness at first glance. I will use it where I have to, but I will also try to blow away some of the fog surrounding what should be part of Amateur Radio's repertoire.

Review of Traditional Spectral Analysis Methods for Speech

The fast Fourier transform (FFT) has traditionally played a major role in speech communications research. Portions of speech such as sustained

[1] Notes appear on page 17.

PO Box 4074
Sedona, AZ 86340
kf6dx@arrl.org

From Mar/Apr 2001 QEX © ARRL

vowel sounds or fricatives, for example, can be modeled as the output of a linear system excited by a source either periodically or randomly varying with time. The output of such a system is simply the product of the frequency response of the vocal tract and the spectrum of the excitation. Fourier analysis is useful in extracting these separate factors from speech waveforms (see Rabiner and Schafer, Reference 7 in Part 1). Over the long term, though, speech signals are considerably more complex than this simple model. Thus, standard Fourier-transform representations that are satisfactory for periodic, stationary signals are not necessarily appropriate for speech signals whose properties rapidly and distinctly change with time.

For reasons that will become apparent, it is reasonable and convenient to assume that the spectral content of speech doesn't change much over short time intervals, say 30 ms or so. This key unlocks a door to some of the redundancy we're seeking as a target for bandwidth-reduction algorithms (more on this later). First, let's consider how certain properties of spectral analysis systems pertain to an analog perceptual speech coder.

In a previous series,[2] I described the FFT and showed that it is a *block transform*; it operates on a block of input samples and produces a block of output samples that portray the frequency content of the input. In another segment,[3] I showed how the damn-fast Fourier transform (DFFT) produces a nearly identical spectral analysis on a sample-by-sample basis. Note that the FFT has fixed frequency resolution directly proportional to the sampling frequency, f_s, and inversely proportional to the length of the input block, N:

$$\Delta f = \frac{f_s}{N} \qquad \text{(Eq 1)}$$

DFFT frequency resolution, on the other hand, can be different for each bin and we don't have to calculate all the bins to get a result.

We are seeking a method of spectral analysis that falls in line with what was shown previously for human hearing: Differential frequency threshold is somehow proportional to frequency. In other words, it is more difficult to detect differences in frequency the higher the frequency of the sounds. It is reasonable is to suspect that an algorithm exploiting this fact will be more efficient than a straight FFT for

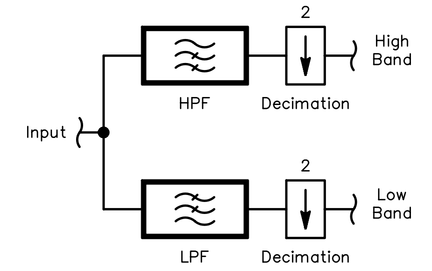


Fig 1—The first stage of subband decomposition.

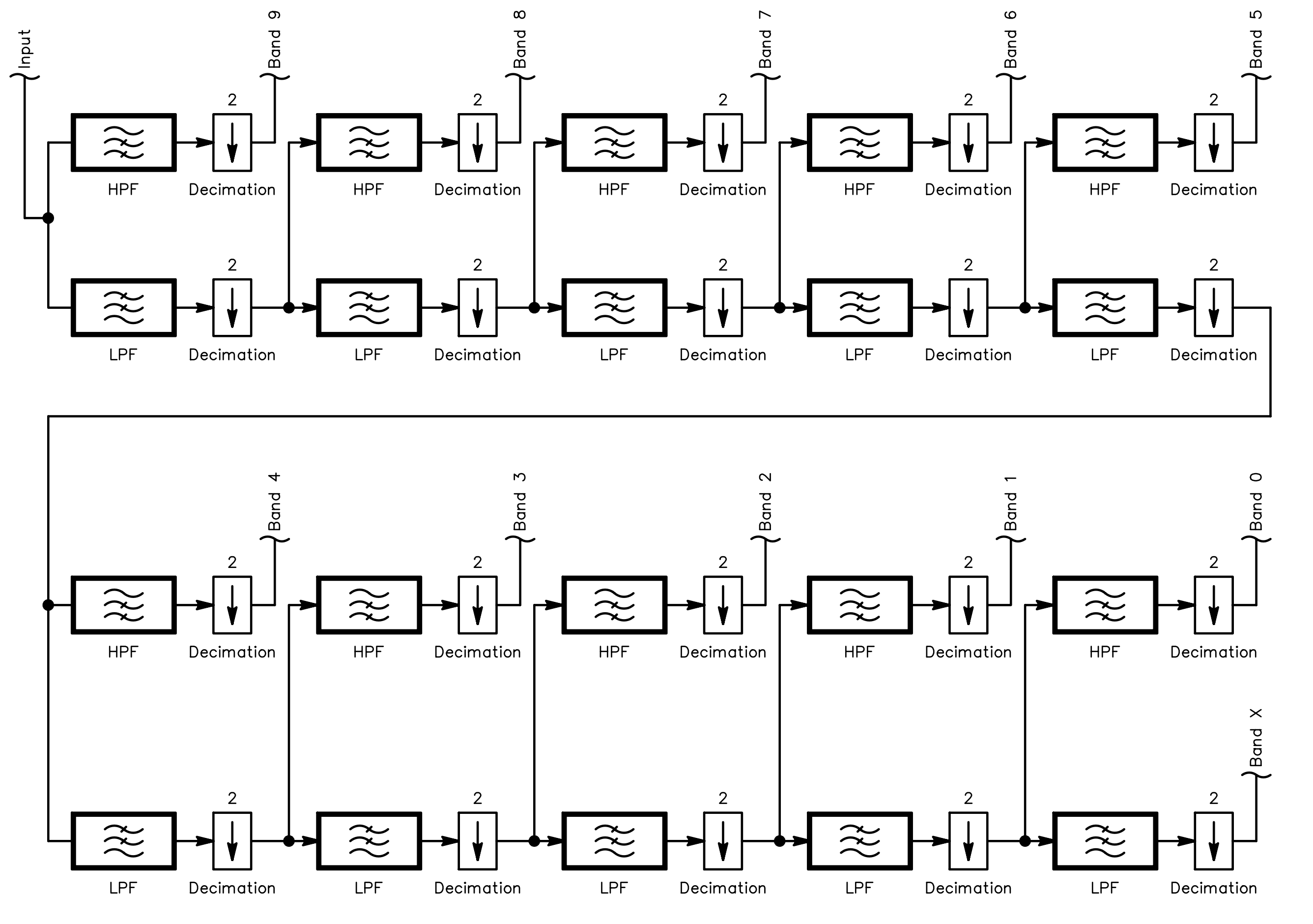


Fig 2—Complete 10-stage subband decomposition.

the analysis of speech signals, because the total number of bins calculated would be greatly reduced. The DFFT seems to meet the requirements of variable resolution and local calculation of selected bins, but the total computational burden can be reduced still further, where facilities exist for high-speed DSP filtering.

Critically Sampled Filter Banks

Digital signal processors are optimized for the computation of *convolution sums* of the form:

$$y_n = \sum_{k=0}^{L-1} h_k \;\; x_{n-k} \qquad \text{(Eq 2)}$$

Such calculations are called multiply-and-accumulate (MAC) calculations. Those are exactly what are required to implement finite-impulse-response (FIR) filters in DSP. FFT and even DFFT algorithms don't necessarily make good use of MACs, so any filtering operation that reduces the complexity of subsequent FFTs is usually beneficial.

Rabiner and Schafer at Bell Labs worked on what are now called *multirate filter banks*.[4, 5] In the first step of one such scheme, the signal under analysis first passes through two filters: a high-pass and a low-pass. See Fig 1. These filters have nearly identical cutoff frequencies and thus separate the input spectrum into high- and low-frequency bands. Since each filter's bandwidth is half the original signal's bandwidth, the sampling rate at each filter's output may be reduced by a factor of two without destroying information. This is Nyquist's criterion. The process of lowering the sampling rate is called *decimation*. In it, every other sample is simply discarded. We could calculate the filter outputs at the higher rate before decimating them, but we save time by calculating only those we intend to keep.

Decimation filters with bandwidth equal to half of the input bandwidth (one quarter of the sampling frequency) are called *half-band* filters. When correctly designed, they have certain properties that lead to further computational savings.

In the second step, the decimated high-pass output is saved for later processing. The decimated low-pass output is further split into two subbands using half-band filters as before. The high-pass output is saved and the low-pass output split again. This process continues until no further band splitting can occur. The result is shown as the block diagram of Fig 2. This is known as a *tree-structured* filter bank. The output of each filter is *maximally decimated* or *critically sampled* because its sampling rate is minimized.

Note that the sampling frequency is halved at each step; hence, the number of samples available in any particular time span is also halved. Band splitting must end when we are left with only a single sample. Except for the final division, outputs from the system all come from the high-pass filters. These are further processed by FFTs that compute frequency content. This decomposition has made it easier to achieve good frequency resolution at the lower frequencies since fewer samples represent fewer frequency bins of an FFT applied there. Because the FFT is a block transform, the size of input blocks for each FFT (that is, the time span) is directly related to the size of blocks coming through the filters. Perhaps this is easier to fathom by studying the following example.

Refer to Fig 2. Let's say the system's raw sampling rate is 31,250 Hz. The input bandwidth is, therefore, half that or 15,625 Hz. In keeping with our premise that speech doesn't change much over time spans on the order of 30 ms, we'll take that to be the length of the input block at the left-hand side of the diagram. To get the whole thing to work nicely, it would be nice if the input block contained a number of samples equal to an integral power of two. By inspection, 2^{10} input samples looks like a good number, since:

$$\begin{aligned} \textit{time span} &= \frac{\textit{number of samples}}{\textit{sampling frequency}} \\ &= \frac{N}{f_s} \\ &= \frac{2^{10}}{31{,}250} \qquad \text{(Eq 3)} \\ &= 32.768 \text{ ms} \end{aligned}$$

At the output of the first stage of filters, the sampling rate is reduced to $f_s/2$ = 15,625 Hz; the number of samples in 32.768 ms is now 2^9. The input signal is split into two bands: 0-7812.5 Hz and 7812.5-15,625 Hz. At the next stage, the number of samples in the low-pass path is reduced to 2^8 and the signal is split into bands 0 to 3906.25 Hz and 3906.25 to 7812.5 Hz. This pattern shows that we are going to perform $\log_2 N$ = 10 iterations of band splitting before we get down to a single sample.

Now we have ten 32.768-ms blocks of samples to analyze, each with a different number of samples. Let's start with the highest-frequency block, which I'll call Band 9. Its bandwidth is 7812.5 Hz and its sampling frequency is twice that. Were we to apply an M-point FFT to these data, we'd have a frequency resolution of $f_s/2M$. Since the block length is $N/2$, we may perform $N/(2M)$ FFTs on adjacent sub-blocks. In other words, we're confronted with a trade-off between good *temporal resolution* and frequency resolution. We're only interested, though, in the content of the entire 32.768-ms block: Its content doesn't change significantly during this period. From what we know about differential frequency threshold, we decide a frequency resolution of about 500 Hz is adequate for this subband. FFT size M therefore need only be:

$$\begin{aligned} M &= \frac{f_s}{\Delta f} \\ &= \frac{15{,}625}{500} \qquad \text{(Eq 4)} \\ &\approx 32 \end{aligned}$$

For a real input signal, this produces 16 analysis frequencies or bins. Actually, 32 bins are produced, but the bins in the top and bottom groups of 16 are just the complex conjugates of one another, and so are redundant. Here we are with a block of 2^9 = 512 samples and needing only 32 for our frequency analysis. Simple and direct would be to compute the 512/32 = 16 FFT blocks and average them. Nevertheless, as it turns out, we may select virtually any contiguous 32-sample block from within the input block, since frequency content doesn't change much over the input block.

So, for band 9, a 32-point FFT taken on the 32-sample block that was harvested. See Fig 3. We now know the frequency content of this band over a 32.768-ms period to a resolution of:

$$\begin{aligned} \Delta f &= \frac{f_s}{M} \\ &= \frac{15{,}625}{32} \qquad \text{(Eq 5)} \\ &\approx 488 \text{ Hz} \end{aligned}$$

The same process is performed on bands 6-8. At band 5, no block harvesting is necessary since the decimated block is already 32 samples in length. When we get to band 4, we run into a little snag: The block is only 16 samples long. It is tempting to just perform a 16-point FFT on this block, but then the frequency resolution would be fs/16 = 488/16 ≈ 30.5 Hz, or the same as for Band 5. This is a getting a bit higher than the Weber fraction (see Part 1), so we decide to analyze this subband over a time period twice that of band 5, or 65.536 ms. We then have our 32 samples and twice the frequency resolution.

Likewise, with band 3, doubling again requires a block-length increase to 131.072 ms to get the 32 samples and a frequency resolution of about 7.6 Hz. This band represents a frequency range of roughly 122 to 244 Hz—getting pretty low. For frequencies below 122 Hz (Bands 0-2), 7.6 Hz is deemed to be sufficient resolution and smaller FFTs are performed on 131.072-ms blocks. Band X, an 11th band, is just the left-over LPF output from the split that produces band 0.

This alteration of subband block lengths reflects the main axiom under which the system operates: Good temporal resolution is more valuable than good frequency resolution at higher frequencies; at low frequencies, good frequency resolution is more important. This is supported by many of the studies cited previously and by common sense.

A conclusion is that above a certain level, improvement in temporal resolution is useless because speech doesn't contain information changing so rapidly; further, the human hearing system cannot distinguish the rapid changes in spectral content that would be produced. Below a certain frequency threshold, improvement in frequency resolution is useless because the information contained in low frequencies is limited. The theory of natural selection[6] seems to indicate that animals do not develop their senses beyond what is necessary. It is therefore no surprise that our hearing matches our ability to communicate verbally. Animals in the wild present a somewhat different story, since they must be able to detect the presence of their enemies through subtle sounds, smells and visual attributes. Still, it is found that surviving species acquired the necessary tools and many of those that are extinct did not.

Perceptual Transforms

Perhaps some readers have experienced Internet audio systems, many of which use perceptual audio coders in one form or another. A data stream at 33.6 kbps occupies a bandwidth of 33.6/2 = 16.8 kHz (when reconstructed) and we know this can be coded in an analog format to fit through a 3-kHz-bandwidth telephone line. This approximately 5.5:1 compression ratio shows that there is hope!

As early as 35 years ago, attempts were made to reduce speech bandwidth by brute-force methods that squeezed all spectral components closer together in frequency.[7] At that time, the fast Fourier transform (FFT) was undergoing a rebirth.[8] I guess it should have been evident from the nature of the beast that such frequency compres-

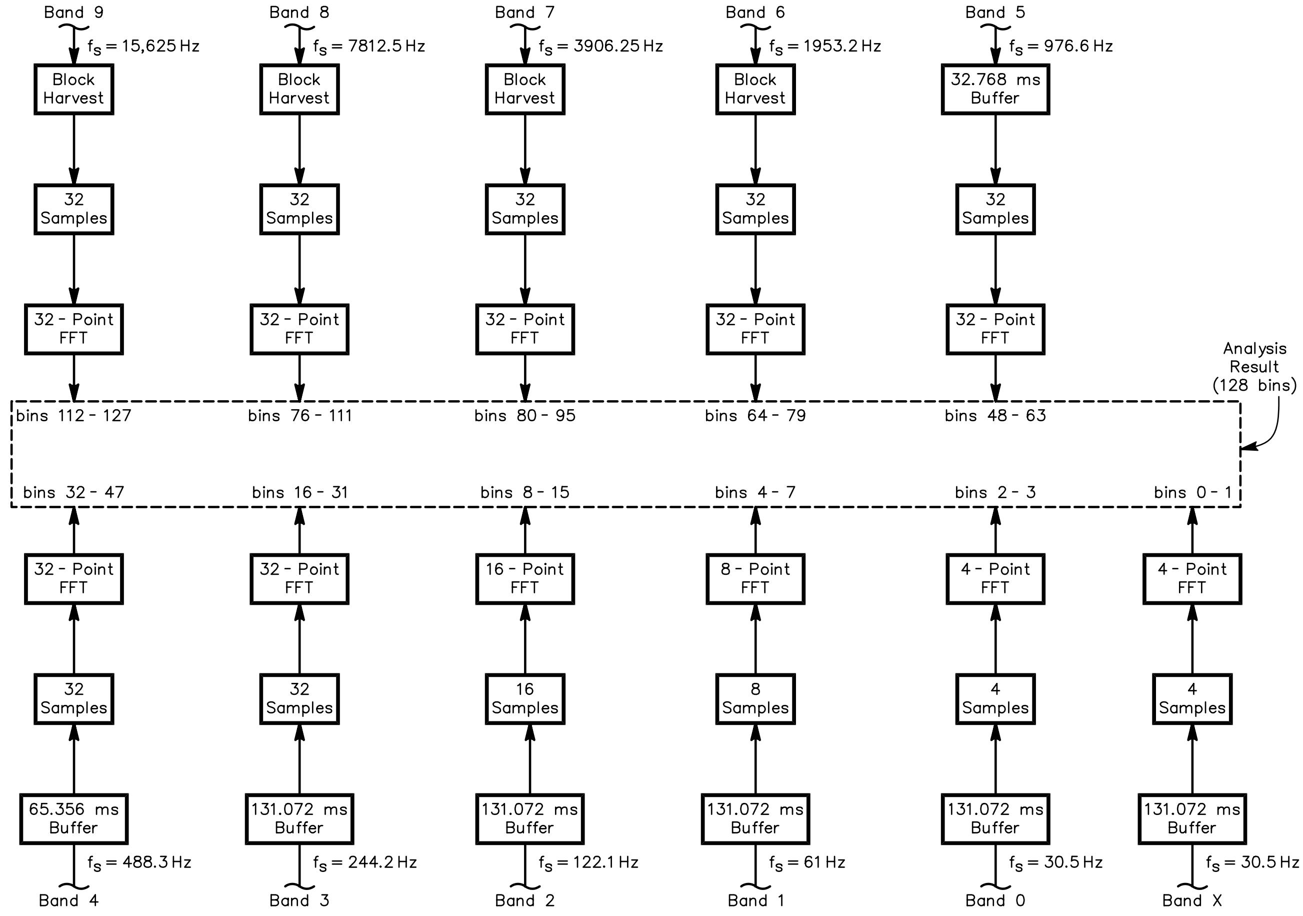


Fig 3—The processing details of subband analysis. Note: Although FFT results are identified only by bin numbers 0-127, each of those bins has a complex-conjugate mate in the range 128-255; mating bins are implied where they are needed to compute inverse transforms. Bin numbers 0-127 correspond to the 128 analysis frequencies.

sion forces discrete transform elements to overlap, resulting in rather serious distortion. Perceptual coding cannot be obtained quite this simply. More-recent efforts utilizing subband coding and other methods[9] must have achieved at least some success, but we still don't see such schemes being employed generally.

Amateurs have also undertaken the quest for analog bandwidth compression. John Ash, KB7ONG, Fred Christiansen, KA6PNW, and Rob Frohne, KL7NA, wrote about a system somewhat similar to mine in *QEX* a few years ago.[10] Their premise was along the same lines that I've explained above: Certain parts of speech are redundant or irrelevant and so may be discarded. I've not heard what their results sound like and I cannot comment on the viability of their approach. I can only write that my tactics are a bit different from theirs in that spectral information is generally preserved across the frequency band of interest.

Anything reducing speech bandwidth by at least a factor of two ought to find immediate application in many services worldwide. It would reduce congestion on our crowded amateur bands as well as on commercial and military channels. It might increase telephone-circuit traffic manifold. It would not play in Peoria, though, unless it met the quality goals set in Part 1. I reluctantly infer, therefore, that previous bids have fallen short.

We have produced samples in the frequency domain of a signal sampled in blocks 32.768 ms long. Now I propose to construct an analog signal from those frequency-domain samples that is also 32.768 ms long but has a greatly reduced bandwidth. I will use the bins obtained from the subband decomposition above as the inputs to a 256-point inverse FFT (FFT^{-1}). The sampling frequency of the output will therefore be:

$$
\begin{aligned}
f_s &= \frac{\textit{number of samples}}{\textit{time span}} \\
&= \frac{256}{0.032768} \qquad \text{(Eq 6)} \\
&= 7812.5 \text{ Hz}
\end{aligned}
$$

In so doing, the bins will represent frequencies spaced 1/0.032768 s ≈ 30.5 Hz apart. The highest-frequency bin will correspond to the highest-frequency bin of the FFT done on band 9. The next-highest-frequency bin will represent the second-highest-frequency bin of the FFT done on band 9, and so on until all analysis bins have been *down-shifted* to their respective places in the coder's *synthesis FFT*$^{-1}$. Note that no temporal-resolution rules have been violated since each bin represents a 32.768-ms block in both FFTs. See Fig 4.

Frequencies of analysis bands are listed in Table 1; synthesis frequencies are listed in Table 2. Frequency resolution in synthesis is proportional to frequency. A speech signal of BW = 15.625 kHz has been coded into BW ≈ (30.5)(128) = 3.90625 kHz! The frequency compression ratio is four. Note that this system, when restricted to half the input bandwidth, produces approximately the same compression ratio. An input bandwidth of 3.90625 kHz, for example, produces output bandwidth of about 977 Hz.

An additional, significant benefit of the system is that it may remove the restrictions placed on high- and low-frequency response by the characteristics of IF and AF filters in transceivers. Table 2's data reveal that very low frequencies are shifted upward by several hundred Hz. That means the low-frequency response of the system is preserved even when the coded signal passes through two bandwidth-limiting filters: one in the transmitter and one in the receiver.

PTC Decoder

The decoder reconstructs the signal using exactly the reverse of the process used in the coder. See Fig 5. It first translates the signal to the frequency domain using a standard, 256-point FFT at the sampling rate of 7812.5 Hz. Input block length for each FFT is 256 samples or 32.768 ms. This produces analysis bins corresponding to 128 discrete frequencies. These samples are then inverse-Fourier transformed by band, with an additional provision for generating time-domain sequences longer than 32 samples for bands 6-9 in that synthesis operation. The

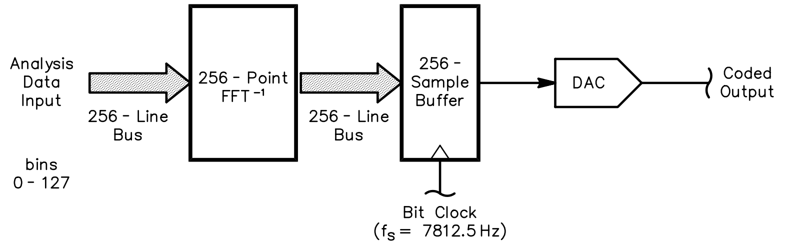


Fig 4—Final processing of the coded analog signal.

Table 1—PTC Codec Example with BW$_{IN}$ = 15,625 Hz, BW$_{OUT}$ = 3906 Hz

Band	*Low-Pass Range (Hz)*	*High-Pass Range (Hz)*	*Sampling Rate (Hz)*	*Samples per 32.768 ms*	*Frequency Resolution (Hz)*
9	0-7812.5	7812.5-15,625	15,625	512	488.3
8	0-3906.3	3906.3-7812.5	7812.5	256	244.2
7	0-1953.2	1953.2-3906.3	3906.3	128	122.1
6	0-976.6	976.6-1953.2	1953.2	64	61
5	0-488.3	488.3-976.6	976.6	32	30.5
4	0-244.2	244.2-488.3	488.3	16 (32 in 65 ms)	15.3
3	0-122.1	122.1-244.2	244.2	8 (32 in 131 ms)	7.6
2	0-61	61-122.1	122.1	4 (16 in 131 ms)	7.6
1	0-30.5	30.5-61	61	2 (8 in 131 ms)	7.6
0	0-15.3	15.3-30.5	30.5	1 (4 in 131 ms)	7.6
X	0-15.3	NA	30.5	1 (4 in 131 ms)	7.6

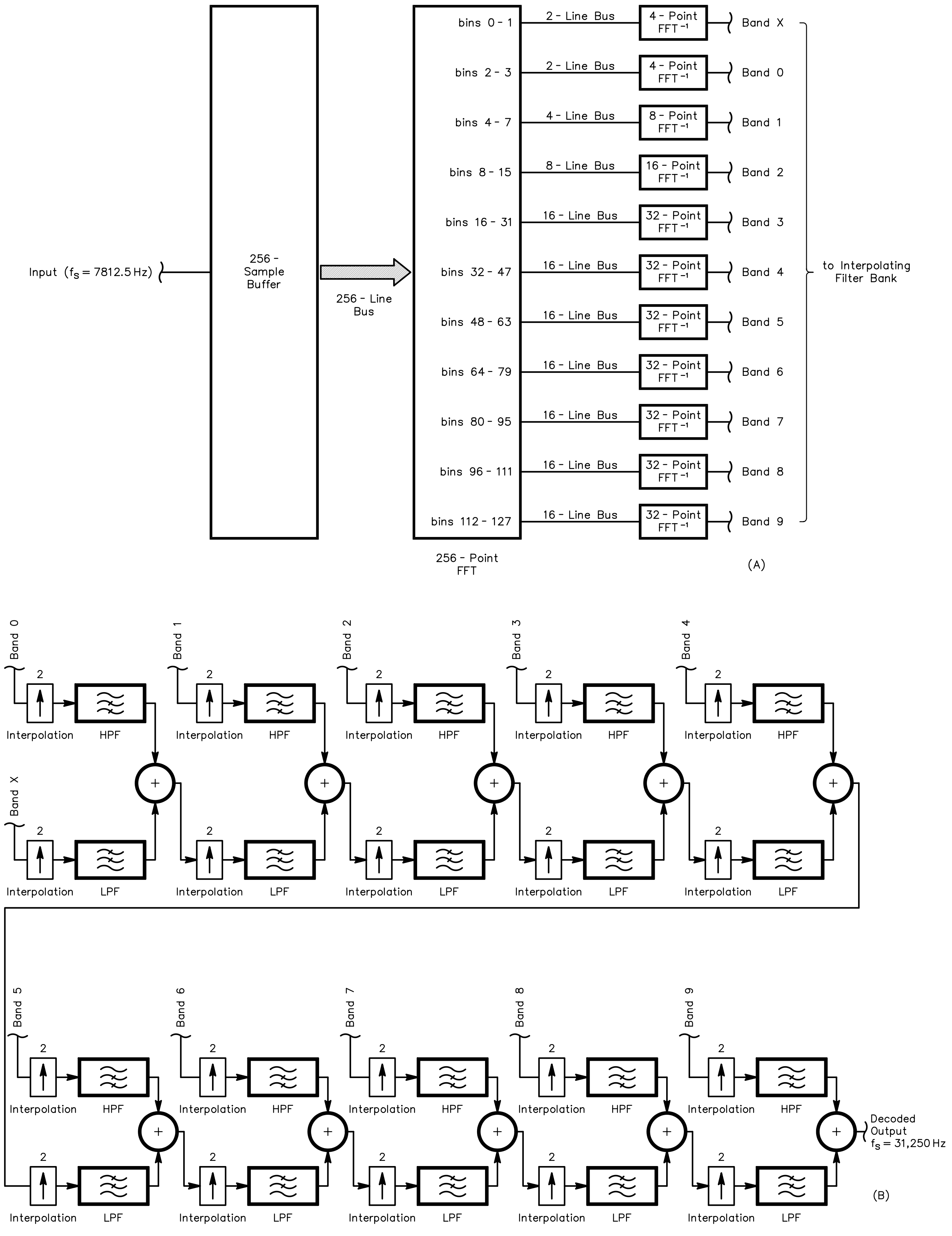


Fig 5—A complete 10-band subband decoder.

From Mar/Apr 2001 QEX © ARRL

sequences are interpolated (interpolation is the inverse of decimation), filtered and combined in a manner opposite to that of the coder. The net result is a 32.768-ms block of output samples at the original sampling frequency of 31,250 Hz. Notice that the bin order of each FFT^{-1} must be reversed; the subsequent interpolation and HPF operations (in Fig 5B) invert the spectrum of the band being processed. The final output is obviously not a perfect reconstruction of the original input, since a compromise has been made between temporal and frequency resolution. In fact, it is quite remarkable how different it looks on a 'scope compared with the original and yet sounds so remarkably the same!

Computational Details

Now let's look at some other details of the processing algorithms. Emphasis will be placed on computational efficiency. My PTC system is currently implemented on a fast PC and does not come close to operating in real time. Chunks of speech may be coded and decoded only after initial recording. Obviously, the next step is to build a codec that processes speech on the fly. One heck of a lot of computation goes on in these algorithms. I calculate that a PTC codec may be implemented on a dedicated DSP platform that has only modest processing power by today's standards. Without the shortcuts outlined below, much more horsepower would be required. Alternatively, increased processing capability would allow greater frequency resolution and therefore improved quality.

In the coder, the output of one filtering stage forms the input to the next. Notice that enough output samples from one stage must be accumulated before the next stage's output can be computed. Further, the input buffer for a particular filter stage must grow beyond 32.768 ms by the length of the filter's impulse response. Finally, the filter's impulse response must be long enough to achieve *orthogonality* between subbands. This term means that no frequency component appearing in either the high- or low-pass subbands appears at significant amplitude in the other filter's output. That is, the filters must be sharp enough not to let frequency components appear simultaneously in both the high-pass and low-pass outputs. This requirement obviously presents itself most critically in and near the transition regions of the filters' frequency responses. Either some overlap or some exclusion of analysis frequencies must be tolerated, since short filters are not very sharp-skirted.

FIR half-band filters can be designed in DSP with impulse responses having odd-numbered coefficients equal to zero. See Fig 6. This is achieved using Fourier design methods when the total number of taps is odd.[11] The significance of this is that the total computational burden is reduced by a factor of two, since those taps with coefficients equal to zero don't need to be computed or added to the convolution sum. In addition, it turns out that half-band, high-pass and low-pass filters may be designed so that their impulse responses are nearly the opposites of one another. For a filter of length L, the coefficients of a half-band high-pass filter, h_k, are simply the negative of the coefficients of a half-band low-pass filter, except for the coefficient at the center of the filter, $h_{(L-1)/2}$. See Fig 7. This further reduces computational complexity by a factor of two, since the output of either filter is just the convolution sum using coefficients $\pm h_k$ plus the term produced with the center coefficient.

Filters of length L = 33 designed using a rectangular window barely meet the requirements above. To avoid having to insert delays in the FFT paths, it is well to store all the input samples for both coder and decoder before calculating all the filter

Table 2—Frequency Mapping in Synthesis of a 4:1 PTC Coder

Band	*Input Frequency Range (Hz)*	*Output Frequency Range (Hz)*	*Number of Frequencies*
9	7812.5-15625	3418.0-3906.25	16
8	3906.3-7812.5	2929.7-3418.0	16
7	1953.1-3906.3	2441.4-2929.7	16
6	976.6-1953.1	1953.1-2441.4	16
5	488.3-976.6	1464.8-1953.1	16
4	244.2-488.3	976.6-1464.8	16
3	122.1-244.2	488.3-976.6	16
2	61-122.1	244.2-488.3	8
1	30.5-61	122.1-244.2	4
0	15.3-30.5	61-122.1	2
X	0-15.3	0-61	2
TOTALS	0-15625 Hz	0-3906.25 Hz	128 frequencies

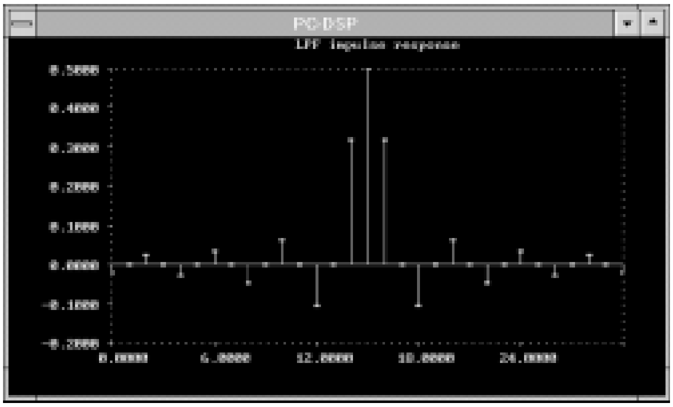


Fig 6—The impulse response of a 33-tap, half-band low-pass filter. Notice that odd-numbered coefficients have a value of zero, save the center coefficient.

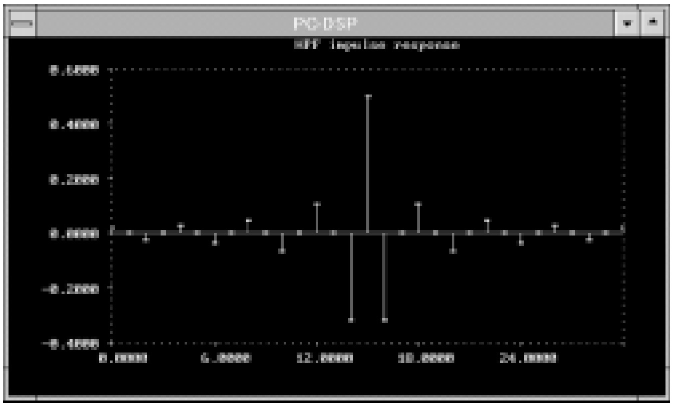


Fig 7—The impulse response of a 33-tap, half-band high-pass filter. Notice that, except for the center coefficient, all the coefficients are simply the negative of the filter depicted in Fig 6.

outputs. This isn't always possible, though, since it results in significant throughput delay. Note that a small delay is always precipitated by the wait for buffers to fill. A complete filter stage using this *polyphase* approach is shown in Fig 8A. The frequency responses of the FIR filters I actually use are shown in Fig 8B.

Either the FFT or DFFT may be used in spectral analysis, depending on the processor. I emphasize again that the DFFT allows independent spectral-leakage control for each bin, although it may incur greater computational burden under certain circumstances. The usual rules regarding scaling of input and output data apply.

Results

The coder and decoder are not synchronized. Because the 32.768-ms frames in the coder are not likely to be aligned in time with those in the decoder, a spectral-smearing effect always occurs. The magnitude of the effect depends heavily on how much the high-frequency content of adjacent frames changes. As stated above, the low-frequency content is not liable to change very much from frame to frame. In the worst imaginable case, high-frequency content changes markedly between frames and half the energy appears in one frame, the other half in the next. Total energy content is preserved, but the temporal resolution is compromised to the tune of half the analysis-block length. This effect has not presented itself as a perceptual problem during testing.

When I started this project, I believed that PTC-coded speech compressed to one fourth of its original bandwidth would still be intelligible, but it is not. The main reason for that seems to be that frequencies corresponding to the pitch of a person's voice are shifted upward in frequency too much to allow the ear to discern them. Formant energy resides much closer to the pitch energy, rendering them indistinguishable from one another. That is not to say you can't still tell that it's speech; it just sounds—well, different.

You may download an example of PTC codec performance from the *QEX* Web site.[12] The package includes some .WAV files: an original, digital recording of my not-so-melodious voice, a PTC-coded version of same with a compression ratio of four and the decoded result. I cannot guarantee they will play exactly right on all systems because of the non-standard sampling rates, but you will get the idea. Also, notice that some work still needs to be done to restore all the naturalness of the original recording after decoding. Application of windowing to time-domain data in analysis and synthesis is the subject of ongoing experimentation. I find it is difficult to tell the difference, though, between coded/decoded speech and the original, at least over HF SSB. After years of listening to 2.4-kHz audio, it astonishes me how much the addition of some sibilance and presence improves perceived speech quality.

Many acquaintances of mine enjoy listening to SSB signals by using a much greater receiver bandwidth than that used in the transmitter. I attribute this to the IMD products appearing beyond the transmitter's bandwidth that pass for sibilance at the receiver. Good thing they can't listen to the IMD products on the other side because I don't think the results would be quite so pleasing.

You may say someone sounds like FM, but the trouble has been that the high-pass filters necessary to eliminate CTCSS tones have had a very deleterious effect on voice signals. More often, I think we're referring to the degree of quieting that is apparent. In the finish, my scheme has some effect on signal-to-noise ratio as well.

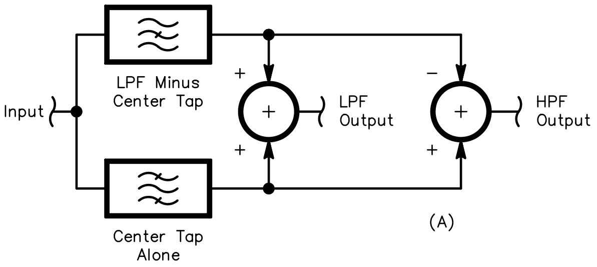
Not only have we reduced bandwidth by a factor of four, but we've also gained a signal-to-noise ratio (SNR) advantage of:

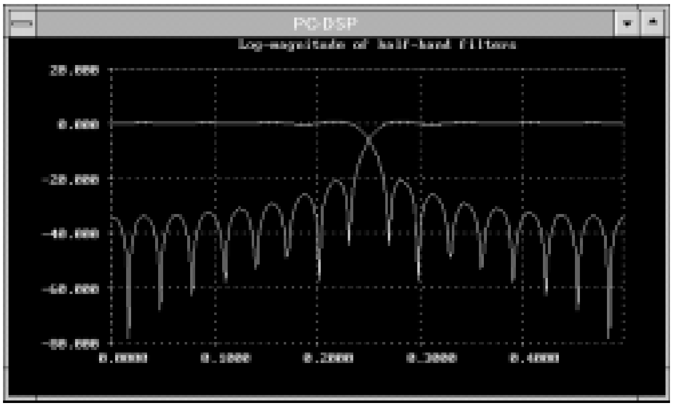
$$\Delta SNR = 10\log 4 \approx 6 \text{ dB} \qquad \text{(Eq 7)}$$

Note that we've also avoided approximately 6 dB of QRM in the process (using appropriate IF filters) and that we've relieved our neighbors in frequency by the same margin. These factors apply to the on-the-air signal, not to the result. Statistical noise from signal processing algorithms usually offsets the reduction in atmospheric noise. The system is subject to a magnified effect from any on-channel interference, if it is polyphonic. That is to say: If polyphonic, on-channel interference occupies bandwidth *m*, I will demodulate it with BW=4*m*. Selective-fading effects are also amplified by the same amount. PTC-coded speech is also a bit more susceptible to frequency errors.

As stated in Part 1, the ear seems to be sensitive to the relative phase of components lying within the same critical band. I postulate this is because such components may produce a beat frequency of greater than the critical bandwidth, resulting in an audible effect. It is interesting to hear how audio waveforms having different phase relationships between their spectral components—and that look quite different on the 'scope—sound remarkably the same.

PTC doesn't process singing, music,





(B)

Fig 8—(A) shows a complete polyphase filter stage. The term polyphase refers to the process of computing partial convolution sums for the filter in question. (B) shows frequency responses of half-band filters.

slow-scan signals or audio containing strong, discrete-frequency components very well, although the concept could certainly be optimized for that purpose. In the form of an external audio processor, PTC would be compatible with virtually any transceiver. The prospective uses of the bandwidth savings are alluring, to say the least. Transmitting unintelligible signals, though, raises a few red flags. Read on.

Is PTC Legal on Amateur Bands? In Commercial Services?

To a non-PTC-equipped receiver, a coded USB signal appears to be several hundred hertz higher in frequency (lower for LSB) than it really is. At low compression ratios (≤2), the coded speech is still understandable without being decoded. An uncoded signal applied to the decoder can still be copied. This is a desirable situation in view of the FCC rules, which are clear when it comes to encryption of signals.[13]

As mentioned, compression ratios of greater than three make coded speech unintelligible. Now I must ask whether those signals may be transmitted legally on the ham bands. If the answer is “No, you can’t,” then another question presents itself: “How much intelligibility do I can lose before I cross the legal line?” If you hope the answer is “Yes, you can,” then certain arguments may come into play.

First, there is an analogy to the legality of unspecified digital modes that have been publicly documented, as outlined in the FCC rules.[14] Digital voice modes have been legal on the phone bands for a long time: They are just as unintelligible as high-compression, PTC signals to the unequipped—maybe more so. PTC coding carries a bandwidth-reduction feather in its cap, which many current digital-voice modes do not. However, the SNR advantage of PTC is minimal compared to that of digital modes.

That brings me to an admonishment about this stuff: *Don’t play the coded .WAV file from the Web site on the air just yet!* If I hear them there or have good reason to believe they were there, I’ll yank the whole thing and the game will be over; however, play the before and after .WAV files as much as you want.

PTC codecs allow four or more times as many voice signals to occupy a given band as compared with uncoded signals. While this may not destroy all QRM, it sure seems to offer a better chance for radio operators to happily coexist. Application of PTC to other services, such as FM land mobile, is not quite so simple. Transceivers usually have synthesizer tuning steps of 12.5 kHz or 25 kHz to match the channel spacing and IF filters. New or heavily modified designs would have to be fielded to take advantage of greater spectral occupancy. Other uses may be made of the saved spectrum without changing channel spacing. Full-frequency-range stereo or four-channel speech, for example, is possible in typical voice bandwidths. With two or more independent channels, more information can be communicated. I can’t see very much reason PTC cannot legally be employed in commercial services.

Since the vast public telephone network has already gone digital, I have to wonder whether it is useful there to increase traffic-handling capacity. Certainly, PTC coding could be applied prior to digitization to achieve a boost; however, large-scale rearrangement of multiplexing equipment would be necessary and lots of new gear would have to be purchased. In addition, it may be that speech-compression coding in digital form (after digitization) would be more cost-effective.

Summary

This work was motivated by the principle that no signal should occupy more bandwidth than necessary to convey the information it contains. Peter Martinez, G3PLX, and the immediate popularity of PSK31 drove that point home.

From the foregoing data, it’s evident that not all components of human speech are necessary to achieve high perceptual quality. Irrelevant components are sometimes made inaudible by masking or critical-band effects and therefore can be eliminated. Many modern digital coding methods make extensive use of these factors to achieve their efficiencies. We also find that speech doesn’t change much from one short time frame to the next, and so contains redundancies. This is also used to reduce bandwidth.

It was shown that if the ear is less sensitive to differences in frequency as frequency increases, then the high-frequency territory is prime ground for bandwidth compression. The underlying principle of PTC is to create an analog signal of lesser bandwidth and frequency resolution in three steps:

1. Analyze the frequency conten of the input signal with non-uniform frequency resolution
2. Combine adjacent frequency bins that are closer together than the differential frequency threshold
3. Down-shift some of the bins in frequency

The processing power required to implement PTC is moderate by today’s standards. I see no reason why affordable codecs cannot be built and put to use reducing QRM. Outboard DSP units are already common equipment at many Amateur Radio stations; many have sufficient number-crunching power for this application. While thinking about digital audio modes, I ask you to also consider this bandwidth-efficient scheme.

Thanks to Bob Heil, K9EID, and Warren Bruene, W5OLY, for their valuable input and assistance. See you in the soup, guys!

Notes

[1] D. Smith, KF6DX, “PTC: Perceptual Transform Coding for Bandwidth Reduction of Speech in the Analog Domain, Part 1,” *QEX*, May/June 2000.

[2] D. Smith, “Signals, Samples, and Stuff: Part 3,” *QEX*, Jul/Aug, 1998.

[3] D. Smith, “Signals, Samples, and Stuff: Part 4,” *QEX*, Sep/Oct, 1998.

[4] R. W. Schafer and L. R. Rabiner, “Design of Digital Filter Banks for Speech Analysis,” Bell System Technical Journal, Vol 50, No. 10, December 1971.

[5] R. W. Schafer, L. R. Rabiner and O. Herrmann, “FIR Digital Filter Banks for Speech Analysis,” Bell System Technical Journal, Vol 54, No. 3, March 1975.

[6] R. C. Stauffer, Ed., *Charles Darwin’s Natural Selection*, 1975.

[7] US Patent No. 3,349,184, Morgan, 1967; also see J. L. Flanagan and R. M. Golden, “Phase Vocoder,” Bell System Technical Journal, Vol 45, No. 9, November 1966.

[9] J. W. Cooley and J. W. Tukey, “An Algorithm for the Machine Calculation of Complex Fourier Series,” *Mathematics of Computation*, American Mathematical Society, Vol 19, April, 1965; **www.ams.org/mcom**.

[10] US Patent No. 4,374,304, Flanagan, 1983; also see US Patent No. 3,510,597, Williamson, 1970.

[11] J. Ash, KB7ONG, F. Christiansen, KA6PNW, and R. Frohne, KL7NA, “DSP Voice Frequency Compandor for use in RF Communications,” *QEX*, July, 1994.

[12] W. E. Sabin and E. O. Schoenike, Eds., *Single Sideband Systems and Circuits*, Second Edition, (New York: McGraw-Hill, 1995).

[13] You can download the .WAV files from **www.arrl.org/qexfiles**. Look for PTCWAVE.ZIP.

[14] 47 CFR 97.113 (a): “No amateur station shall transmit: . . . (4) . . . messages in codes or ciphers intended to obscure the meaning thereof . . . .”

[15] 47 CFR 97.309 (b).