SWS: an overview and history.

Philip Rubin

Sinewave synthesis of speech is a technique for creating digital acoustic signals based on natural speech utterances. This is done by generating simulated pure tones that track the natural resonant frequencies of the vocal tract. Usually, between two to four of these time-varying tones occurring at the same time are sufficient to create a signal that can be identified as speech and transcribed fairly accurately by most individuals. But these sinewave signals are very unusual. They lack most of those short-term features that are characteristic of natural speech, such as harmonic structure, broadband formants, the transient noises seen in fricatives, aspirates, and consonant bursts, etc. Our intuition about these sinewave signals is that we should hear a set of independent whistles, changing rapidly in time and frequency, as if we were listening to a slide whistle trio. Musical, perhaps; speech, unlikely. But if you are encouraged to listen to this electronic ensemble as some strange form of speech, that is usually what you will hear. Out of a chaotic jumble of random whistles, a coherent linguistic message emerges. Why? How can separate streams of sound so quickly merge into a coherent percept? When this does happen, why is it so difficult to go back to hearing only the chaotic jumble of sound? What does this all mean and what possible scientific use can we make of this apparent acoustic oddity?

The technique now commonly used to rapidly create these unusual sounds and the first software sinewave synthesizer (*SWS*) were originally developed by Philip Rubin at Haskins Laboratories in 1977 (Rubin, 1980). SWS built on the pioneering work at Haskins Laboratories in the 1950s that used an early hardware device, the Pattern Playback (Cooper et al.., 1951), to determine the information critical for speech perception by synthesizing acoustic signals based on spectrographic information (see, e.g., Delattre, et al., 1952). This approach led to a sustained research program that uncovered the details of the speech code (Liberman, 1957; Liberman et al., 1967). Other influences were numerous, particularly the work on syllable recognition by Cutting (1974) and Bailey and colleagues (Bailey et al., 1977) and theoretical considerations of event perception (Gibson, 1950, 1979; Jenkins, 1974, 1985).

The sinewave synthesis system developed by Rubin had, perhaps, a different focus than the approaches used by earlier investigators. At the heart of this new system was the desire to create a tool that could be used to explore the global, spectral-temporal properties of speech signals by simplifying the synthesis of sentence-length utterances derived from natural speech and minimizing the attention usually given to the short-term aspects of the signal. For over twenty-five years this technique and variations of the original program have been used by Robert Remez and colleagues (Remez et al., 1981; see *References* for a complete list) and other scientists to explore a range of issues in the area of speech perception, phonetics, cognitive psychology, and related areas. This paper provides an informal history of the development of the technique, includes details on some of the technical aspects of sinewave synthesis; provides sample *MATLAB* algorithms (Ellis, 1996; Rubin and Frost, 1996) and parameter data for sinewave synthesis, summarizes some of

the experimental approaches that have used the technique, and discusses new research and technical directions.

Serendipity and intuition

On a Thursday afternoon in Sep. 1977, Alvin Liberman, at that time President of Haskins Laboratories in New Haven, Connecticut, engaged in a systematic search of the laboratories looking for me. He found me hidden away in one of the lab's sound-reducing IAC booths. Al had a simple request. Could I help to produce some "non-speech" tokens for a speech perception experiment? Al knew that I had been spending a portion of my time working on a primitive tone-based software music synthesizer. His hope was that I could create some combination of musical tones that would sound somehow like speech, but that would clearly not be speech and would not be phonetically identifiable. The use of nonspeech material in perception experiments was a commonly used technique, including in the speech perception research at Haskins, such as the early work with the Pattern Playback, and in what was at the time recent experimental work (Cutting, 1974; Bailey et al., 1977). Al, however, was always interested in experimental innovation and in this case was looking for something that was a little more "speechlike" than usual and that could be varied in some way to make it more or less speechlike.

I decided to attempt to base the non-speech tokens on real speech. At that time, a number of techniques were available for altering (degrading) natural and/or synthesized speech tokens to systematically manipulate the intelligibility of signals. One simple approach involved mixing the natural signal with varying amounts and types of noise. Many other techniques had been used, including reversing the speech signal, inverting it, filtering it in a variety of ways, etc. I chose another approach. Just as modern music synthesizers can be driven by transcriptions of scores that are translated into frequencies and durations, I wondered about the possibility of creating non-speech tone combinations by driving my music synthesizer with frequencies derived from real speech.

As a first pass, I decided to simply input the center frequency values of the first (lowest) three formants of natural utterances. Formants are the rapidly changing resonant frequencies of the vocal tract that result as a direct consequence of sound being "shaped" by this rapidly changing, deformable structure. Although formants are not simply numbers, they do correspond to the peaks of the acoustic spectrum (see Figure 1, below). At the time we were working with an early version of a new commercial software signal analysis system called *ILS* (Interactive Laboratory System) that could easily and quickly estimate the frequency and amplitude of spectral peaks (formant center frequency values) from digitized signals. *ILS* estimated formant values using linear predictive techniques and could save the estimated values in data files. I quickly modified my music synthesis program so that it could input information about frequency, amplitude, and time, from these *ILS*-generated data files. The new program synthesized up to 20 simultaneous time-varying tones (software frequency oscillators) that could then be combined into a digitized waveform for immediate listening or subsequent storage (Whalen et al., 1990). Because

the sound generation module used a constantly changing sine function, I dubbed the program SineWave Synthesis (*SWS*).

After analyzing the first speech token from a set spoken by Arthur Abramson ("Where were you a year ago?"), the formant values for the first three formants were converted to a format that could be read by *SWS*. *ILS* often made errors during analysis. These "errors" were not "corrected" for this first attempt. I expected that this token would be non-speech, but I was hoping that I would be able to hear certain events, such as "beats" corresponding to the syllable structure of the original utterance.

SWS quickly generated its first token. I was very surprised that I clearly heard "Where were were you a year ago?" over a background of whistles, pops, and other non-speech sounds. This is not what I expected to hear. I had been expecting to hear an acoustic token with prominent events that, perhaps, I would be able to count, but not anything that was intelligible. I also expected to hear separate "streams" of sound corresponding to the individual tones. Instead, I perceived a coherent signal conveying a linguistic message while simultaneously perceiving the strangeness of the signal. Although the message was clear, the "voice" quality of the synthesized signal was not natural, sounding very computer-like and musical. Of course, I already knew what the utterance was, so this was not a fair test of the intelligibility of this token. Ratcheting up my scientific methodology, I called my wife on the phone. I put the phone up to the speaker and hit the play button. She had no trouble identifying the token, but she was biased by me into considering it as speech. Unfortunately, the ease with which people can be trained to hear the linguistic content in sinewave analogues of speech, and the difficulty that people have hearing these signals as nonspeech after hearing them as speech, meant that they did not appear to be the kind of tunable nonspeech tokens that Liberman desired for use in his experiments. However, they appeared to me to have potential use for exploring certain global aspects of speech perception. I called my colleague, Robert Remez, a professor at Barnard College. Robert also had no trouble identifying this strange signal. After listening to the sound, and identifying the sentence, he added "There's lots of research to be done." He was correct and this research focusing on questions of perceptual organization has now continued for almost 25 years. The research and related uses of sinewave synthesis for perceptual experiments will be described, below. Before doing so, however, additional technical details will be provided on the approach that was used to create sinewave tokens modeled on actual utterances.

Technology

The technique of synthesizing sinewave tokens based upon natural utterances involves a fairly standard series of steps. A variety of different approaches and tools can be used to follow these steps. The usual approach involves:

1. Analysis of the natural utterance to obtain formant center frequencies and, where possible, information about formant amplitude or bandwidth, and overall amplitude for the original utterance. Originally this analysis was done using *ILS* to conduct an LPC analysis of the signal. Fourier analysis has often yielded better analysis results. Our use of ILS was replaced by the development and use of the *HADES* system (Rubin, 1995) which provided for both types of analysis and also supported both the automatic extraction of formant center frequencies and the hand tracing or correction of formant frequency values. These days, standard packages such as *MATLAB* and/or other commercial tools are usually used to analyze speech signals and produce formant center frequency estimates that are converted into input for the sinewave synthesis routines.

2. Corrections of analysis errors, if so desired. Corrections can be done automatically and/or by hand. Examples of corrections include setting onset and offset amplitudes of individual tone portions to zero to avoid sharp discontinuities at the beginning or end of segments. Other corrections involve the proper identification of the continuity of a tone, once again to avoid discontinuities. These discontinuities sometimes occur because the analysis technique misses a particular formant (that is, missing the second formant and, thus, counting the third formant as the second format). Interpolation can be used to provide for continuity where appropriate. Finally, it is always possible to introduce information into the data based upon linguistic knowledge about the structure of the speech signal.

3. Conversion of the analysis values into a file compatible with the *SWS* program. This file is called a Sine Wave Input file (.SWI). More recent versions of *SWS* can use tab-delimited text files generated by *Microsoft Excel* or other programs. See Appendix I for an example of a portion of the data in a SWI file.

4. Synthesis of digital sound data. The general approach involves the use of a computer program to input the source data, simulate coupled pure tone (sinewave) oscillators to generate output data in digital, sound data, and then save these values in sampled data form (PCM, .AIFF, .WAV, etc.). An example of early *Fortran* code for a portion of this process is provided in Appendix II. A portion of a *MATLAB* version is provided in Appendix III.

5. Using software tools, convert the sampled data files into acoustic signals for audio output for listening or use in perceptual tests.

Details of the analysis/synthesis approach used to create sinewave tokens are provided below.

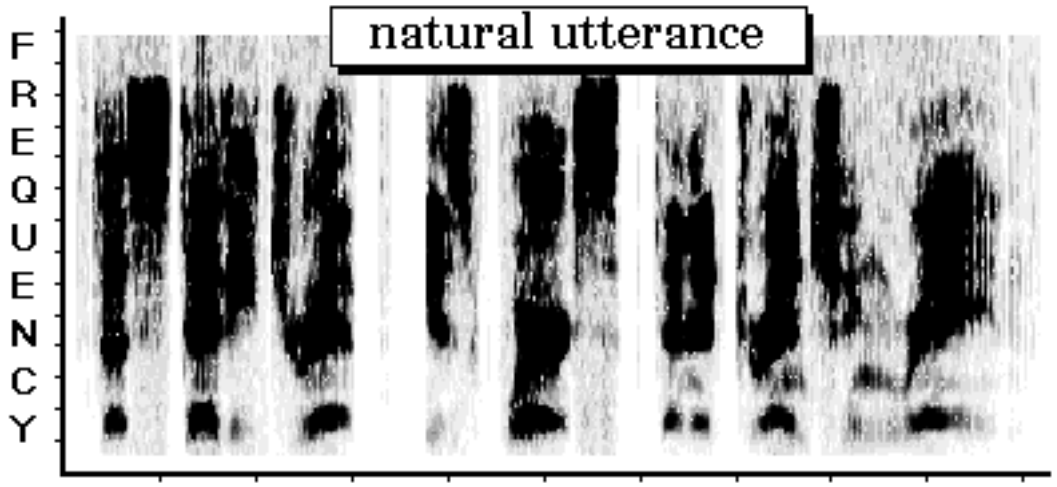

Figure 1: Spectrogram of "The steady drip is worse than a drenching rain."

The display above is called a spectrogram, which provides an acoustic "picture" of a speech utterance. In this type of display, time is represented on the horizontal axis, frequency on the vertical axis, while amplitude corresponds to the darkness of portions of the signal. This spectrogram illustrates some important characteristics of natural speech acoustics. The regular vertical striations are due to glottal pulsing (caused by the activity of the vocal cords); the broadband formats (the dark horizontal bands) are each a natural resonance sustained by the column of air enclosed by the vocal tract between the larynx and the lips; aperiodic sources and transients can be attributed to consonantal releases (e.g. /b/, /d/ sounds), frication (e.g. /s/, /v/ sounds), and aspiration (e.g. /h/ sound). The spectrogram above was obtained by analyzing the utterance: "*The steady drip is worse than a drenching rain*."

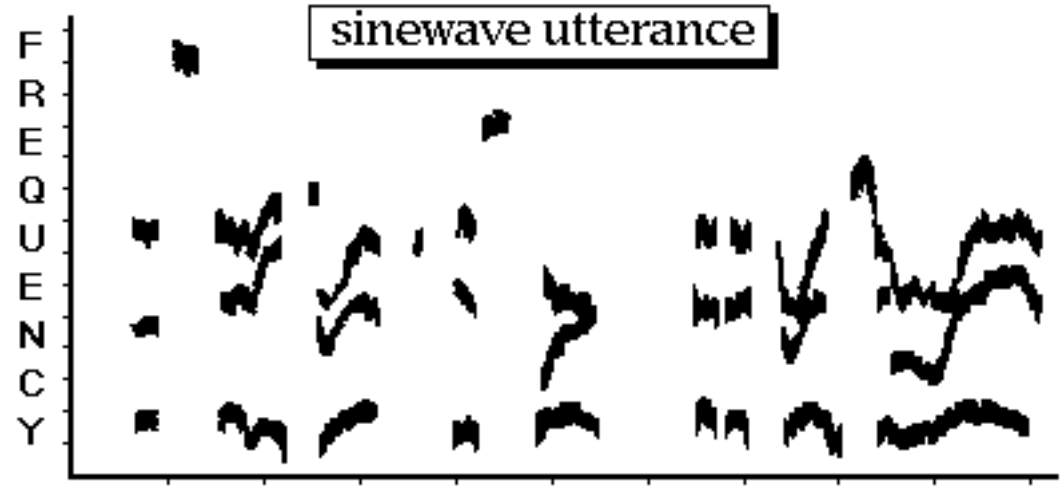

Figure 2: Spectrogram of sinewave version of "The steady drip is worse than a drenching rain."

A sinewave replica of a natural utterance (shown above) discards the fine-grain acoustic properties of speech, retaining only the coarse-grain changes in the spectra over time. This pattern of spectral changes is estimated by linear predictive analysis or other methods of spectral peak-picking. The result is a record of formant center-frequencies and amplitudes at regular intervals throughout an utterance. When this numerical description of the spectra of an utterance is used as the parameter set for the SineWave Synthesizer (*SWS*), the result is a pattern of sinusoids, each one fit to the frequency and amplitude track of a formant in the natural utterance. Without imitating the spectra of the actual signal components, a sinewave complex replicates the overall pattern of spectral changes of the utterance. Phonetic information is preserved in these changes and is evidently not the sole preserve of the traditional momentary acoustic "cues." The spectrogram above is of a sinewave replica of the utterance: "*The steady drip is worse than a drenching rain*." Note that the apparent greater than expected bandwidth of the tones shown in the figure above is a result of the analysis of the signal.

The standard approach that we used when creating sinewave tokens was based an analysis of the signal that yielded estimates of formant frequencies every 10 msec. Figure 3, below, provides a schematic overview of the input parameters used to generate a sinewave simulation of the sentence "Where were you a year ago?" Note that the vertical striations correspond to the 10 msec. analysis frames and that the height of these values loosely corresponds to input amplitude values (quickly estimated by using inverse bandwidth from the ILS analysis). Three or four tones were usually used to generate the sinewave tokens. These tones usually tracked the formant center frequencies of the voiced portions of the signal. Fricatives and aspirates could be simply simulated by using tones that tracked the approximate frequency paths of these very transient aspects of the signal. These pure tone whistles usually proved adequate to yield intelligible tokens, given training of the subjects. *SWS* could generate from 1-20 parallel tones. Playing individual tones to subjects never yields a phonetic percept, rather subjects report hearing whistles or, in the case of the tones tracking the third formant, bird calls. Subjects who receive training can often reliably identify 2-tone tokens, but not with the degree of reliability of 3-tone tokens. This is discussed in the Research section, below. Also discussed is the important observation that non-native speakers of English usually do very poorly when asked to identify sinewave speech based on English productions.

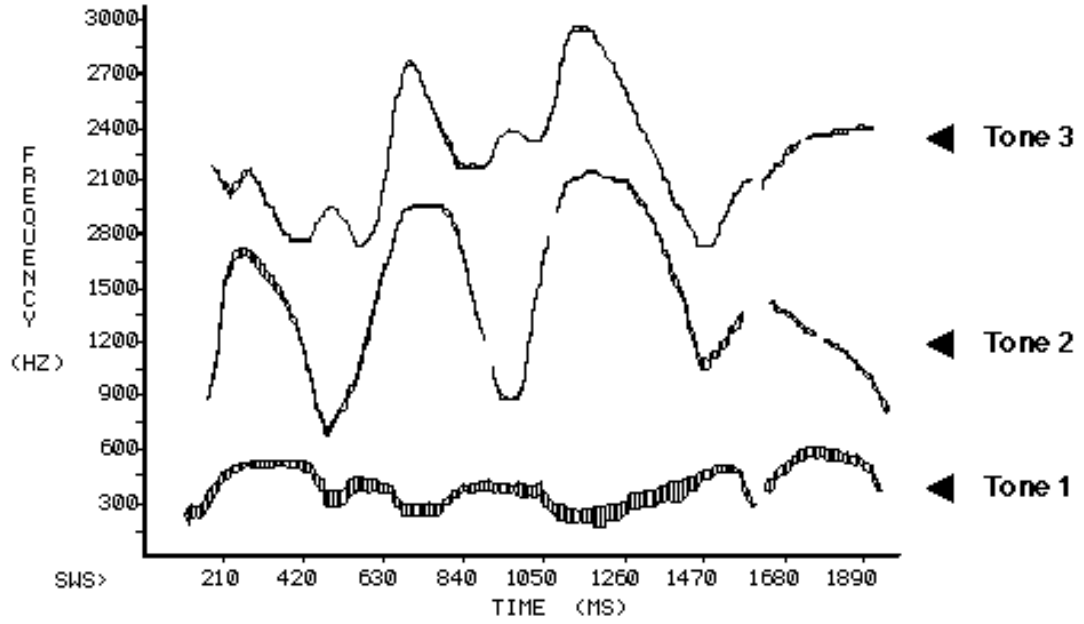

Figure 3: Input parameters for the Haskins Sinewave Synthesis program (*SWS*) for the token "Where were you are year ago?"

As noted above, the figure above is a display of the parameters used by *SWS* to synthesize tokens created for studying the temporal aspects of speech. The horizontal axis shows time in milliseconds; the vertical axis shows frequency in hertz (Hz; i.e. cycles per second)). The pattern is a graph of frequency and amplitude variations of three sinusoids. Height in the plane indicates frequency; the thickness of each tracing indicates amplitude. The properties of tonal analogs of speech vary over time. Accordingly, the tones rise and fall in frequency and amplitude in imitation of the frequency and amplitude variations of vocal resonances over the course of an utterance. Note, however, that unlike the natural speech signal, sinewave speech does not have the normal structure – there are no broadband formants; there is no regularly pulsed source; the normal short-time "cues" found in speech signals are apparently missing; etc. What remains are just 3 (or sometimes 4) rapidly changing pure tones. For most listeners, these signals are sufficient to convey a phonetic message (that is, listeners hear them as speech and can identify the individual speech sounds). Why? The pattern of variation imposed on the sinusoidal carriers is sufficient information for the perception of phonetic attributes despite the elimination of natural acoustic elements. This reveals that perception is sensitive to information carried by patterns of stimulation independent of the elements composing the pattern.

It is difficult to understand the character of sinewave speech without actually hearing it. In order to make this easier and to provide additional background experience, an interactive website (Haskins Laboratories SineWave Synthesis web demo) was created (Remez, Rubin and Pardo, 1996). This web demonstration includes sample parameter files and an on-line sound files that lets you listen to sinewave examples, compare them to their natural speech models, and explore other experimental conditions. The SWS demo was based on a *HyperCard* stack (*HyperSWS;* Rubin, 199*2*) that was developed by Philip Rubin

at the suggestion of Carol Fowler to demonstrate the phenomenon of sinewave replication at the Haskins Laboratories Board of Trustees Meeting, on Nov. 4, 1992.

Martin Cooke and colleagues have created *MATLAB* demonstrations (Cooke, 1998; Cooke & Browne, 1999) that were motivated primarily by studies into the perception of simultaneous sine-wave speech utterances (Barker & Cooke, 1997). In these experiments, listeners were asked to transcribe pairs of sine-wave sentences presented simultaneously. Results were compared against (phoneme-level) transcription scores for pairs of natural utterances. As noted, above, other experiments have examined the effect of dichotic presentation (Remez et al, 1994), reduced numbers of sine-wave 'formants', further reduction of the synthesized tones to constant amplitudes or frequencies (Remez & Rubin, 1990) and the role of amplitude modulation (Carrell & Opie, 1992; Barker, 1998). Cooke's demonstrations allow all of these manipulations to be explored.

.

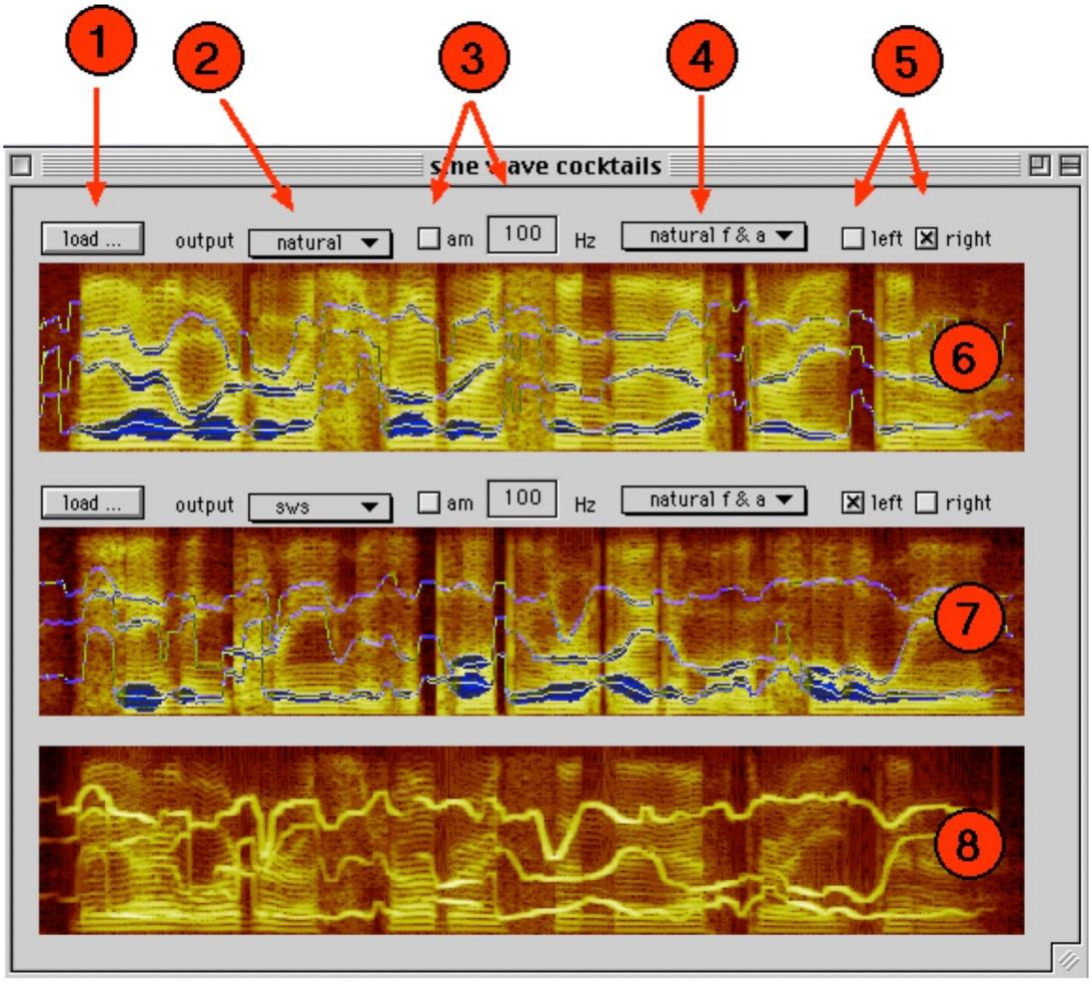

Figure 4: Sinewave demonstration module from Cooke's MAD system

Figure 4, above, shows a portion of the Cooke sinewave demonstration that is part of his MAD system. Launching the demo brings up a window similar to the one above (initially without the spectrograms,). "The window contains three display panels (6,7,8). The top two (6,7) are used to display spectrogams and SWS tracks for a pair of utterances, which are loaded via the buttons (1). The lower panel (8) displays a spectrogram of the mixture. Once spectrograms and SWS tracks are loaded, clicking on the spectographic image results in the associated signal being played. SWS formants can be selected and unselected by clicking on the tracks. Unselected formants do not contribute to the sound output, and their absence can be noted in the mixture spectrogram. A popup menu (2) selects which signal is used for playback. Options are 'natural', 'SWS' and 'silent'. The latter prevents the signal from contributing to the mixture. Amplitude modulation can be added to the SWS waveform. If checkbox (3) is checked, AM at the specified rate is applied to the SWS signal. Sidebands will be visible in the mixture for all but the lowest rates of AM. By default, SWS tracks use frequency and amplitude values extracted from the natural utterance (for details on the mainly-automatic procedure used, see Barker, 1998). Optionally, via popup menu (4), the listener can select constant amplitude or constant frequency SWS tracks. Finally, the two signals can be presented diotically or dichotically via checkboxes (5)." (Cooke, 1998). The natural utterances used to generate the sinewave tokens come from the TIMIT database (Garofolo et al., 1993).

Research

(***Note***: This section was originally intended to summarize the Remez, et al. research program that has used sinewave speech and was unfortunately never completed for this draft. Portions of the abstracts have been extracted for use in this draft. This was followed by examples of the use of sinewave speech by other researchers provided by Rubin and Remez.)

Remez and colleagues

Remez, R. E., Rubin, P. E., Pisoni, D. B., & Carrell, T. D. (1981). Speech perception without traditional speech cues. *Science*, *212*, 947-950.

Remez, R. E., & Rubin, P. E. (1984). Perception of intonation in sinusoidal sentences. *Perception & Psychophysics*, *35*, 429-440.

Remez, R. E., Rubin, P. E., Nygaard, L. C., & Howell, W. A. (1987). Perceptual normalization of vowels produced by sinusoidal voices. *Journal of Experimental Psychology: Human Perception and Performance*, *13*, 40-61.

Remez, R. E., & Rubin, P. E. (1990). On the perception of speech from time-varying attributes: Contributions of amplitude variation. *Perception & Psychophysics*, *48*, 313-325.

Remez, R. E., Rubin, P. E., Berns, S. M., Pardo, J. S., & Lang, J. M. (1994). On the perceptual organization of speech. *Psychological Review*, *101*, 129-156.

In the first controlled experiment by Remez and colleagues that used sinewave speech (Remez et al., 1981), a three-tone sinusoidal replica of a naturally produced utterance was identified by listeners, despite the readily apparent unnatural speech quality of the signal. The time-varying properties of these highly artificial acoustic signals are apparently sufficient to support perception of the linguistic message in the absence of traditional acoustic cues for phonetic segments. Sinewave replicas of natural utterances discard the fine-grain acoustic properties of speech, retaining only the coarse-grain changes in the spectra over time. Although sinewave replicas do not, on first impression, sound a lot like speech, listeners can do a good job identifying them depending upon experimental conditions (Remez et al, 1981; Barker & Cooke, in revision).

Most familiar synthetic speech aims to copy natural acoustic elements meticulously. That is why synthetic speech sounds voice-like, despite the mechanical quality of its articulation. In contrast, sinewave replication discards all of the acoustic attributes of natural speech, except one: the changing pattern of vocal resonances. By fitting 3 or 4 sinusoids to the pattern of resonance changes, sinusoidal signals preserve the dynamic properties of utterances without replicating the short-term acoustic products of vocalization.

If speech perception depended upon the particular sounds produced by talkers (the pop of the "p", the hiss of the "s", the hum of the "m", the click of the "k", or the buzz of the "z"), then sinusoidal signals lacking these attributes should not evoke impressions of consonants, vowels, words, etc. In fact, listeners who were asked to identify sinewave signals, reported "bad electronic music," "radio interference," etc., and no speechlike qualities. However, when asked to transcribe a "strangely-synthesized sentence," listeners readily reported the words of the natural utterances on which the sinewave signals were modeled.

The use of sinusoidal replicas of speech signals reveals that listeners can perceive speech solely from temporally coherent spectral variation of nonspeech acoustic elements (Remez and Rubin, 1983).

When listeners hear a sinusoidal replica of a sentence, they perceive linguistic properties despite the absence of short-time acoustic components typical of vocal signals. Is this accomplished by a post-perceptual strategy that accommodates the anomalous acoustic patterns ad hoc, or is a sinusoidal sentence understood by the ordinary means of speech perception? If listeners treat sinusoidal signals as speech signals however unlike speech they may be, then perception should exhibit the commonplace sensitivity to the dimensions of the originating vocal tract. A study by Remez and colleagues (Remez et al., 1987) employing sinusoidal signals raised this issue by testing the identification of target /bVt/, or b-vowel-t, syllables occurring in sentences that differed in the range of frequency variation of their component tones. Vowel quality of target syllables was influenced by this acoustic correlate of vocal-tract scale, implying that the perception of these non-vocal signals includes a process of vocal-tract normalization. Converging evidence suggests that the perception of sinusoidal vowels depends on the relations among component tones and

not on the phonetic likeness of each tone in isolation. The findings support the general claim that sinusoidal replicas of natural speech signals are perceptible phonetically because they preserve time-varying information present in natural signals.

A general account of auditory perceptual organization has developed in the past several decades. It relies on primitive devices akin to the Gestalt principles of organization to assign sensory elements to probable groupings and invokes secondary schematic processes to confirm or to repair the possible organization. Although this conceptualization is intended to apply universally, the variety and arrangement of acoustic constituents of speech violate Gestalt principles at numerous junctures, cohering perceptually, nonetheless. Experiments by Remez and colleagues (Remez et al., 1994) have examined organization in phonetic perception, using sinewave synthesis to evade the Gestalt rules and the schematic processes alike. These findings falsify a general auditory account, showing that phonetic perceptual organization is achieved by specific sensitivity to the acoustic modulations characteristic of speech signals.

Other research

Bailey, P., Summerfield, Q., & Dorman, M. (1977). On the identification of sine-wave analogues of certain speech sounds. *Haskins Laboratories Status Report on Speech Perception, SR-51/52*, 1-26. Haskins Laboratories, New Haven, CT.

Best, C.T., Morrongiello, B. & Robson, R. (1981). Perceptual equivalence of acoustic cues in speech and nonspeech perception. *Perception & Psychophysics 29*, 191-211.

Best, C.T., Studdert-Kennedy, M., Manuel, S. & Rubin-Spitz, J. (1989). Discovering phonetic coherence in acoustic patterns. *Perception & Psychophysics 45*, 237-250.

Carrell, T. & Opie (1992). *Perception & Psychophysics 52*, 437-445.

Johnson, K. & Ralston, J. V. (1994). Automaticity in speech perception: Some speech/nonspeech comparisons. *Phonetica 51*, 195-209.

Barker , J. (1998). *PhD Thesis*, University of Sheffield.

Barker, J. & Cooke, M. (1999). *Speech Communication*.

New Directions

(This section provides examples of new technical and research directions.)

Audio-visual perception

Saldaña, H. M., Pisoni, D. B., Fellowes, J. M., & Remez, R. E. (1996). Audio-visual speech perception without speech cues. In *Proceedings of the International Conference on Spoken Language Processing—96*. (pp. 2187-2190). Philadelphia: ICSLP.

Saldaña, H. M., Fellowes J. M., Remez, R. E., & Pisoni, D. B. (1996) Audio-visual speech perception without speech cues: A first report. In D. G. Stork and M. E. Hennecke (Eds.), *Speechreading by Man and Machines: Models, Systems and Applications* (pp. 145-151). Berlin: Springer-Verlag.

Goh, W.D., Pisoni, D.B., Kirk, K.I., & Remez, R.E. (2001). Audio-visual perception of sinewave speech in an adult cochlear implant user: A case study. *Ear and Hearing*, *22*, 412-419.

Lachs, L. & Pisoni, D. B. (2004). Specification of cross-modal source information in isolated kinematic displays of speech. *Journal of the Acoustical Society of America 116*, 507-518.

Cochlear Implants

Audio-visual perception of sinewave speech in an adult cochlear implant user: a case study.
Goh WD, Pisoni DB, Kirk KI, Remez RE.
Ear Hear. 2001 Oct;22(5):412-9.
Indiana University, Bloomington, USA.
OBJECTIVE: The purpose of this case study was to investigate multimodal perceptual coherence in speech perception in an exceptionally good postlingually deafened cochlear implant user. His ability to perceive sinewave replicas of spoken sentences, and the extent to which he integrated sensory information from multimodal sources was compared with a group of adult normal-hearing listeners to determine the contribution of natural auditory quality in the use of electrocochlear stimulation. DESIGN: The patient, "Mr. S," transcribed sinewave sentences of natural speech under audio-only (AO), visual-only (VO), and audio-visual (A+V) conditions. His performance was compared with the data collected from 25 normal-hearing adults. RESULTS: Although normal-hearing participants performed better than Mr. S for AO sentences (65% versus 53% syllables correct), Mr. S was superior for VO sentences (43% versus 18%). For A+V sentences, Mr. S's performance was comparable with the normal-hearing group (90% versus 86%). An estimate of the amount of visual enhancement, R, obtained from seeing the talker's face

showed that Mr. S derived a larger gain from the additional visual information than the normal-hearing controls (78% versus 59%). CONCLUSIONS: The findings from this case study of an exceptionally good cochlear implant user suggest that he is perceiving the sinewave sentences on the basis of coherent variation from multimodal sensory inputs, and not on the basis of lipreading ability alone. Electrocochlear stimulation is evidently useful in multimodal contexts because it preserves dynamic speech-like variation, despite the absence of speech-like auditory qualities.

Loizou, P. C., Dorman, M. & Tu, Z. (2004). On the number of channels needed to understand speech. *Journal of the Acoustical Society of America 106 (4)*, 2097-2103.

Laflen, J. B. & Talavage, T. M. (2003). Locating implant stimulation sites at any location along the cochlea.

Cognitive neuro-imaging and sinewave speech

Wong, D., Miyamoto, R.T., Pisoni, D.B., Sehgal, M., & Hutchins, G. (1999). PET imaging of cochlear-implant and normal-hearing subjects listening to speech and nonspeech stimuli. *Hearing Research 132*, 34-42.

Wong, D., Pisoni, D.B., Learn, J., Gandour, J., Miyamoto, R.T., and Hutchins, G.D. (2002). PET imaging of differential cortical activation to monaural speech and nonspeech stimuli. *Hearing Research 166/1-2*, 9-23 (April).

Binder and Liebenthal:
"Welcome to the Language Imaging Laboratory at the Medical College of Wisconsin. We are part of the Department of Neurology at MCW, and an affiliated lab of the Functional Imaging Research Center at MCW.
(If you are looking for the main Neurology website, click here.)

Our main focus is on using functional MRI to study the neurophysiological correlates of language processes. Though our interests range widely, the chief focus is on left temporal lobe systems associated with perceptual processes and memory stores underlying language behavior, particularly single word recognition.

A second major focus of our laboratory is on development and testing of methods for presurgical functional localization of language and episodic memory systems. Our aim is to use the basic knowledge gained from fMRI studies of normal language processing to predict and prevent neuropsychological defcits in patients who must undergo surgery in sensitive brain areas.

Some examples of projects currently ongoing include:
Studies of speech perception: We manipulate spectral content (bandwidth, spectral resolution, harmonic content, formant transition parameters), signal-to-noise ratio, lexical

status, and phonological neighborhood characteristics of natural and synthetic speech signals during passive listening, discrimination, lexical decision, and categorization tasks. The goal is to tease apart levels of processing involved in auditory word recognition. Many of these experiments focus on the speech/nonspeech distinction using signals (sinewave speech, multichannel signal-correlated noise) that can be perceived as either speech or nonspeech depending on context."

Liebenthal E., Binder J.R., Piorkowski R.L., Remez R.E. (2003). Short-term reorganization of auditory cortex induced by phonetic experience. *Journal of Cognitive Neuroscience*. *15*, 549-558

What does it all mean?

Sinewave speech has some interesting properties that have made it a useful technique for a exploring a number of questions, including perceptual organization, the importance of global aspects of the speech signal, the boundaries between speech and non-speech, the relationship between perception and production, and a host of other issues. Leading scientists in related areas of research vary in their opinions regarding the importance of the technique and what it has to tell us. Two of these opinions are provided, below.

Robert E. Remez: "How does a listener know what a talker just said? A fundamental perceptual component in acts of spoken communication is the analysis of sensory samples of speech. However, perception of the phonetic properties in stimulation cannot proceed as if sensory activity stems from speech sources alone. We speak and listen to each other amid multiple sources of sound. Indeed, the vocal apparatus itself is a source of respiratory and ingestive sound as well as speech. In this respect, the perception of speech naturally entails two functions: 1) an organizational function that identifies a sensory pattern attributable to a spoken source; and 2) an analytical function that identifies the phonetic attributes conveyed in a sensory pattern. Traditional accounts of each function rely on the similarity of sensory samples to perceptual standards designated as the most likely sensory effects of consonants and vowels. Studies of sinewave replicas of speech undermine this conceptualization, because intelligible sinewave signals are not similar to vocally produced sound; likewise, sinewave signals are not familiar to listeners, neither as auditory forms nor as phonetic sequences. This evidence supports a conclusion about the boundary conditions on a perceptual explanation of speech: Early sensory coding is exquisitely sensitive to coarse-grain spectro-temporal properties of the signal independent of momentary or likely sensory effects."

Al Bregman (Bregman, 1992): "Sine-wave-analog speech bears the same relation to a recording of real speech as a cartoon does to a photograph of a real face. The interesting thing, from a psychologist's point of view is that the recognition can be accomplished in either case. Something important must have been retained in the cartoon. The recognition also points to the flexibility of the recognition system. Sine-wave analog speech is a useful experimental tool because it allows some aspects of speech to be retained while others are discarded. I believe that the sudden "snap" from hearing it as noises to hearing it as speech represents the switching in of speech schemas, either due to their elicitation by properties of the signal or to suggestion by the experimenter. The heavy contribution of top-down processes without a lot of bottom-up support makes this an interesting stimulus."

Clearly, over the past 25 years the use of sinewave speech has proven itself to be useful as a research tool. It has had both practical and theoretical implications and has sparked energetic and ongoing debates in the areas of language, speech and psychology. As Robert Remez indicated in 1977, there remains much work to do and as Al Liberman would always say, many discoveries to be made.

References

Bailey, P., Summerfield, Q., & Dorman, M. (1977). On the identification of sine-wave analogues of certain speech sounds. *Haskins Laboratories Status Report on Speech Perception, SR-51/52*, 1-26. Haskins Laboratories, New Haven, CT.

Barker , J. (1998). *PhD Thesis*, University of Sheffield.

Barker, J. & Cooke, M. (1999). *Speech Communication*.

Best, C.T., Morrongiello, B. & Robson, R. (1981). Perceptual equivalence of acoustic cues in speech and nonspeech perception. *Perception & Psychophysics 29*, 191-211.

Best, C.T., Studdert-Kennedy, M., Manuel, S. & Rubin-Spitz, J. (1989). Discovering phonetic coherence in acoustic patterns. *Perception & Psychophysics 45*, 237-250.

Bregman, A. (1990). *Auditory Scene Analysis*. Cambridge, MIT Press.

Bregman, A. (1992). Sine-wave-analog speech. AUDITORY list, Sep. 3, 1992.

Carrell, T. & Opie (1992). *Perception & Psychophysics 52*, 437-445.

Cooke, M. (1998). Sine-wave speech cocktails. http://www.dcs.shef.ac.uk/~martin/MAD/sws/sws.htm

Cooke, M.P. and Brown, G. J. (1999) Interactive explorations in speech and hearing. *Journal of the Acoustical Society of Japan (E),* 20, 2, 89-97.

Cooper, F.S., Liberman, A. M., & Borst, J. M., The interconversion of audible and visible patterns as a basis for research in the perception of speech. *Proceedings of the National Academy of Science*, 1951, *37*, 318-325.

Delattre, P., Liberman, A., Cooper, F. & Gerstman, L. (1952). An experimental study of the acoustic determinants of vowel color: Observations on one- and two-formant vowels synthesized from spectrographic displays. *Word 8*, 195-210.

Ellis, D. (1996). Sinewave Speech Analysis/Synthesis in Matlab. http://www.ee.columbia.edu/~dpwe/resources/matlab/sws/

Fellowes, J. M., Remez, R. E., & Rubin, P. E. (1997). Perceiving the sex and identity of a talker without natural vocal timbre. *Perception & Psychophysics*, *59*, 839-849.

Fowler, C. A., Rubin, P. E., Remez, R. E., & Turvey, M. T. (1980). Implications for speech production of a general theory of action. In B. Butterworth (Ed.), *Language Production, Vol. I: Speech and Talk* (pp. 373-420). New York: Academic Press.

Gibson, James J. (1950). *The Perception of the Visual World*.

Garofolo, J., Lamel, L., Fisjer, W., Fiscus, J., Pallet, D. & Dahlgren, N. (1993). DARPA TIMIT: Acoustic phonetic continuous speech corpus. *NIST Technical Report*.

Goh, W.D., Pisoni, D.B., Kirk, K.I., & Remez, R.E. (2001). Audio-visual perception of sinewave speech in an adult cochlear implant user: A case study. *Ear and Hearing*, *22*, 412-419.

Jenkins, J. J. (1974). Remember that Old Theory of Memory? Well, Forget it! *American Psychologist 29(11).*

Jenkins, J. J. (1985). Acoustic information for objects, places, and events. In Warren, W. H., & Shaw, R. E., (eds.), *Persistence and change Proceedings of the first international conference on event perception*. Erlbaum, Hillsdale, NJ.

Lachs, L. & Pisoni, D. B. (2004). Specification of cross-modal source information in isolated kinematic displays of speech. *Journal of the Acoustical Society of America 116*, 507-518.

Liberman, A. M. (1957). Some results of research on speech perception. *The Journal of the Acoustical Society of America 29*, 117-123.

Liebenthal E., Binder J.R., Piorkowski R.L., & Remez R.E. (2003). Short-term reorganization of auditory cortex induced by phonetic experience. *Journal of Cognitive Neuroscience*. *15*, 549-558

Loizou, P. C., Dorman, M. & Tu, Z. (2004). On the number of channels needed to understand speech. *Journal of the Acoustical Society of America 106 (4)*, 2097-2103.

Remez, R. E. (1996). Perceptual organization of speech in one and several modalities: Common functions, common resources. In *Proceedings of the International Conference on Spoken Language Processing—96* (pp. 1660-1663). Philadelphia: ICSLP.

Remez. R. E. (2000). Perceptual requirements for organizing and analyzing speech or perceiving a message (and a messenger) in the time-varying tones. The Center for Language and Speech Processing Seminar Series, The Johns Hopkins University, Oct. 3, 2000.

Remez, R. E., Fellowes, J. M., & Rubin, P. E. (1997). Talker identification based on phonetic information. *Journal of Experimental Psychology: Human Perception and Performance*, *23*, 651-656.

Remez, R.E., Fellowes, J.M., Pisoni, D.B., Goh, W.D., & Rubin, P.E. (1997). Audio-visual speech perception without traditional speech cues: A second report. In C. Benoit & R. Campbell (Eds.), *Proceedings of the ESCA workshop on audio-visual speech processing: Cognitive and computational approaches* (pp. 73-76). Grenoble: European Speech Communication Association.

Remez, R., Fellowes, J.M., Pisoni, D.B., Goh, W.D. & Rubin, P.E. (1998). Multimodal perceptual organization of speech: Evidence from tone analogs of spoken utterances. *Speech Communication, 26*, 65-73.

Remez, R. E., Pardo, J.S., Piorkowski, R. L., & Rubin, P. E. (2001). On the bistability of sinewave analogs of speech. *Psychological Science, 12*, 24-29.

Remez, R. E., & Rubin, P. E. (1983). The stream of speech. *Scandinavian Journal of Psychology*, *24*, 63-66.

Remez, R. E., & Rubin, P. E. (1984). Perception of intonation in sinusoidal sentences. *Perception & Psychophysics*, *35*, 429-440.

Remez, R. E., & Rubin, P. E. (1990). On the perception of speech from time-varying attributes: Contributions of amplitude variation. *Perception & Psychophysics*, *48*, 313-325.

Remez, R. E., & Rubin, P. E. (1993). On the intonation of sinusoidal sentences: Contour and pitch height. *Journal of the Acoustical Society of America*, *94*, 1983-1988.

Remez, R. E., Rubin, P. E., Berns, S. M., Pardo, J. S., & Lang, J. M. (1994). On the perceptual organization of speech. *Psychological Review*, *101*, 129-156.

Remez, R. E., Rubin, P. E., Nygaard, L. C., & Howell, W. A. (1987). Perceptual normalization of vowels produced by sinusoidal voices. *Journal of Experimental Psychology: Human Perception and Performance*, *13*, 40-61.

Remez, R. E., Rubin, P. E., & Pisoni, D. B. (1983). Coding of the speech spectrum in three time-varying sinusoids. In C. Parkins and S. W. Anderson (Eds.), *Cochlear Prostheses* (pp. 485-489). New York: New York Academy of Sciences.

Remez, R. E., Van Dyk, J. L., Fellowes, J. M., & Rubin, P. E. (1998). On the perception of qualitative and phonetic similarities of voices. In P. K. Kuhl and L. A. Crum (Eds.) *Proceedings of the 16th International Congress on Acoustics and the 135th Meeting of the Acoustical Society of America, Volume 4* (pp. 2063-2064). New York: Acoustical Society of America.

Remez, R. E., Rubin, P. E., Pisoni, D. B., & Carrell, T. D. (1981). Speech perception without traditional speech cues. *Science*, *212*, 947-950.

Rubin, P.E. (1980). Sinewave synthesis. Internal memorandum. Haskins Laboratories, New Haven, Connecticut.

Rubin, P. (1992). *HyperSWS*. HyperCard stack.

Rubin, P.E. (1995). HADES: A Case Study of the Development of a Signal Analysis System. In A. Syrdal, R. Bennett, & S. L. Greenspan (Eds.), *Applied Speech Technology*. CRC Press, Boca Raton, 501-520.

Rubin, P., Baer, T., & Mermelstein, P. (1981). An articulatory synthesizer for perceptual research. *Journal of the Acoustical Society of America*, *70*, 321-328.

Rubin, P., Remez, R., & Pardo, J. (1996), SineWave Synthesis weblet. http://www.haskins.yale.edu/haskins/MISC/SWS/SWS.html

Rubin, P. & Vatikiotis-Bateson, E. (1998). Talking heads. In D. Burnham, J. Robert-Ribes, & E. Vatikiotis-Bateson (Eds.), *International Conference on Auditory-Visual Speech Processing - AVSP'98*, 231-235, Terrigal, Australia.

Rubin, P. & Vatikiotis-Bateson, E. (1998). Measuring and modeling speech production in humans. In S.L. Hopp, M. J. Owren & C.S. Evans (Eds.), *Animal Acoustic Communication*. Springer-Verlag, New York, 251-290.

Saldaña, H. M., Pisoni, D. B., Fellowes, J. M., & Remez, R. E. (1996). Audio-visual speech perception without speech cues. In *Proceedings of the International Conference on Spoken Language Processing—96*. (pp. 2187-2190). Philadelphia: ICSLP.

Saldaña, H. M., Fellowes J. M., Remez, R. E., & Pisoni, D. B. (1996) Audio-visual speech perception without speech cues: A first report. In D. G. Stork and M. E. Hennecke (Eds.), *Speechreading by Man and Machines: Models, Systems and Applications* (pp. 145-151). Berlin: Springer-Verlag.

Sheffert, S. M., Pisoni, D. B., Fellowes, J. M & Remez, R. E. (2002). Learning to recognize talkers from natural, sinewave and reversed speech samples. *Journal of Experimental Psychology: Human Perception and Performance*, 28, 1447-1469.

Whalen, D. H., Wiley, E. R., Rubin, P. E., & Cooper, F. S. (1990). The Haskins Laboratories' pulse code modulation (PCM) system. *Behavior Research Methods, Instruments, & Computers, 22(6),* 550-559.

Williams, D.R., Verbrugge, R. R. & Studdert-Kennedy, M. (1983). Judging sine wave speech as speech and nonspeech. *Journal of the Acoustical Society of America 74*, S66.

Wong, D., Miyamoto, R.T., Pisoni, D.B., Sehgal, M., & Hutchins, G. (1999). PET imaging of cochlear-implant and normal-hearing subjects listening to speech and nonspeech stimuli. *Hearing Research 132*, 34-42.

Wong, D., Pisoni, D.B., Learn, J., Gandour, J., Miyamoto, R.T., and Hutchins, G.D. (2002). PET imaging of differential cortical activation to monaural speech and nonspeech stimuli. *Hearing Research 166/1-2*, 9-23 (April).

Check these:

Hodsgon, P. & Miller, J. L. (1996). Internal structure of phonetic categories: Evidence for within-category trading relations. *Journal of the Acoustical Society of America 100*, 565-576.

Johnson, K. & Ralston, J. V. (1994). Automaticity in speech perception: Some speech/nonspeech comparisons. *Phonetica 51*, 195-209.

Makashay, M. J. (2003). Individual differences in speech and non-speech perception of frequency and duration. Unpublished doctoral dissertation, The Ohio State University.

Remez, R. E. (in press). The perceptual organization of speech. In D. B. Pisoni and R. E. Remez (Eds.), *The Handbook of Speech Perception.* (pp. 000-000). Oxford: Blackwell.

Remez, R. E. (in press). Three puzzles of multimodal speech perception. In E. Vatikiotis-Bateson, G. Bailly & P. Perrier (Eds.). *Audiovisual Speech Processing* (pp. 000-000). Cambridge, Massachusetts: MIT Press.

Remez, R. E., Fellowes, J. M., Blumenthal, E. Y., & Shoretz Nagel, D. (2003). Analysis and analogy in the perception of vowels. *Memory & Cognition*, 31, 1126-1135.

Remez, R. E. (2003). Establishing and maintaining perceptual coherence: Unimodal and multimodal evidence. *Journal of Phonetics*, 31, 293-304.

Liebenthal, E., Binder, J. R., Piorkowski, R. L., & Remez, R. E. (2003). Short-term reorganization of auditory analysis induced by phonetic experience. *Journal of Cognitive Neuroscience*, 15, 549-558.

Remez, R. E. (2001). The interchange of phonology and perception considered from the perspective of organization. In E. V. Hume and K. A. Johnson (Eds.), *The Role of Speech Perception Phenomena in Phonology* (pp. 27-52). San Diego: Academic Press.

Appendix I: Portion of an SWS input file (.SWI)

```
- - - - - - - - - - WHERE.SWI  - - - - - - - - - - -
 3
    0.00
    0.0000,0.000000
    0.0000,0.000000
    0.0000,0.000000
   10.00
    0.0000,0.000000
    0.0000,0.000000
    0.0000,0.000000
.
.
.
  200.00
  408.0000,0.468840
 1239.0000,0.237700
 2128.0000,0.129140
  210.00
  437.0000,0.610980
 1443.0000,0.229640
 2069.0000,0.227000
  220.00
  451.0000,0.544540
 1545.0000,0.272920
 2069.0000,0.269800
  230.00
  466.0000,0.458180
 1618.0000,0.452920
 2026.0000,0.359780
  240.00
  481.0000,0.394480
 1676.0000,0.468840
 2055.0000,0.452920
  250.00
  495.0000,0.335760
 1705.0000,0.508200
 2084.0000,0.432540
```

Appendix II: FORTRAN code

```
C****
C
C    Note:
C
C      What follows is slightly modified FORTRAN source code
C      that is being used as part of the SWS weblet.
C      This source code is intended only as an illustration.
C      Included is the SWS sinewave sound generation code subroutines.
C      Modifications have been made for purposes of illustration.
C      (c) 1980-1996, Haskins Laboratories, New Haven, CT.
C
C****
C
C
          SUBROUTINE SWS_GENV1
C
C         P. RUBIN  2/26/80, 2/5/81
C                   6/16/81    FOR VAX
C                   2/13/82
C                  10/27/82    CORRECT HEADER WRITE
C                   5/26/83    HANDLE PREEMPHASIS,CONTIG
C                   6/19/84    CHANGES FOR AMP_TYPE
C
C    PURPOSE:
C
C      GENERATE PCM DATA FROM ARRAYS OF TIME SLICE (TIMSL),
C      SINE WAVE FREQUENCY (SWF) AND SINE WAVE AMPLITUDE (SWA)
C      INFORMATION.
C
C      VOICE 1 : SINE WAVE
C
C    ARGUMENTS PASSED IN COMMON:
C
C      OUTCHN    LOGICAL      OUTPUT CHANNEL FLAG, WHERE:
C                               .TRUE. = CHANNEL ASSIGNED
C                               .FALSE.= CHANNEL NOT ASSIGNED
C
C      ERR       INTEGER      ERROR FLAG, WHERE:
C                               0 = NO ERROR
C                              <0 = ERROR
C
C    NOTES:
C
C      FREQUENCY IS CONSTRAINED TO BE BETWEEN 1 HZ AT THE MINIMUM
C        AND THE NYQUIST LIMIT ( (SAMPLING FREQ./2) - 1  ) AT THE MAX.
C
C=====================================================================

     PARAMETER MXRECL = 64         ! MAX. RECORD LENGTH
     PARAMETER HEADER = 4          ! # HEADER RECORDS
     PARAMETER MIDSCL = 2048       ! PCM MIDSCALE
     PARAMETER PI     = 3.14159    ! PI
     PARAMETER TWOPI  = 6.28319    ! 2. * PI


C=====================================================================
=======
```

```
C
C*****
C
C    INCLUDE 'SWS.INC/NOLIST'      ! MAIN SWS COMMON
C
C*****

C*****
C    S W S . I N C
C
C         P. RUBIN  9/16/80
C                   7/21/81  FOR VAX
C                   6/15/84
C                   7/18/84  CHANGES FOR MAC AND INC FEATURES
C                    8/8/84
C                   9/19/85  V4.1 WITH MANY MORE SLICES
C                    3/4/88  V4.2 supports DATA TRANSLATION output
C                  10/15/91  changes for additional output systems
C                    6/4/92  changes for AUDIO flag
C
C    SINEWAVE SYNTHESIZER COMMON
C*****

     PARAMETER MAXSW = 50               ! MAX # SINE WAVES
     PARAMETER MAXSL = 2000             ! MAX # TIME SLICES
     PARAMETER MAXDISP= 999             ! MAX. # TIME SLICES TO DISPLAY
     PARAMETER CTRLZ = -99              ! CONTROL-Z CODE

     INTEGER    MAXMDEF                 ! MAX. # OF MACRO DEFINITIONS
     PARAMETER  ( MAXMDEF = 100 )

     COMMON /SWSM/   LIST,TTO,TTI,LP,CFLUN,SPFIL,DTFIL,PCFIL,
     1         SWF(0:MAXSL,MAXSW),SWA(0:MAXSL,MAXSW),
     2         TIMSL(0:MAXSL),NSLIS,NSW,SR,DEF_SR,
     3         NRECS,NHEDER,TEXT1,LTEX1,TEXT2,LTEX2,
     4         NCODE, CODEC, CODEV, PROMPT, DEFV, ICHAN,
     5         SWDAT, SPOPEN, DISPLA, OUTCHN, PCMDAT,
     6         LNSP, FAXIS, TAXIS, NCKPAG, TIK_FLAG, ERR,
     7         IVOICE, VOIARG(12), EMPH, LPCMEXT, SC_CHAN,
     8         AMP_MAX, NAMESP, PCMEXT, AMP_TYPE, AUDIO,
     9         INI_LUN, ARGS, FPARS, NUMMDEF, MDEF, LMDEF,
     1         PCM_DT


     INTEGER   LIST                ! LISTING DEVICE LUN
     INTEGER   TTO                 ! TERMINAL OUT LUN
     INTEGER   TTI                 ! TERMINAL IN  LUN
     INTEGER   LP                  ! LINEPRINTER  LUN
     INTEGER   CFLUN               ! COMMAND FILE LUN
     INTEGER   SPFIL               ! SPEECH FILE  LUN
     INTEGER   DTFIL               ! SW DATA FILE LUN
     INTEGER   PCFIL               ! PCM DATA FILE LUN
     REAL      SWF                 ! SINE WAVE FREQUENCIES
     REAL      SWA                 ! SINE WAVE AMPLITUDES
     REAL      TIMSL               ! TIME SLICES
     INTEGER   NSLIS               ! # SLICES
     INTEGER   NSW                 ! # SINE WAVES
     REAL      SR                  ! SAMPLING RATE
     REAL      DEF_SR              ! SAMPLING RATE DEFAULT
     INTEGER   NRECS               ! # RECS IN PCM FILE
     INTEGER   NHEDER              ! # HEADER BLOCKS IN PCM
     CHARACTER*100 TEXT1           ! TEXT 1
     INTEGER   LTEX1               ! LENGTH OF TEXT 1
```

```
      CHARACTER*100 TEXT2           ! TEXT 2
      INTEGER   LTEX2               ! LENGTH OF TEXT 2
      INTEGER   NCODE               ! # INPUT CODES
      BYTE      CODEC(2)            ! CHARACTER CODES
      REAL      CODEV(2)            ! CODE VALUES
      REAL      PROMPT              ! PROMPT VALUE
      REAL      DEFV                ! DEFAULT VALUE
      INTEGER   ICHAN               ! CHANNEL NUMBER
      LOGICAL   SWDAT               ! SINEWAVE DATA FLAG
      LOGICAL   SPOPEN              ! SPEECH FILE OPENED ?
      LOGICAL   DISPLA              ! DISPLAY ALLOWED ?
      LOGICAL   OUTCHN              ! OUTPUT CHANNEL ASSIGNED?
      LOGICAL   PCMDAT              ! PCM DATA EXIST ?
      INTEGER   LNSP                ! LENGTH OF NAMESP
      REAL      FAXIS               ! FREQ. AXIS SCALE DEF.
      REAL      TAXIS               ! TIME  AXIS SCALE DEF.
      INTEGER   NCKPAG              ! LINE # FOR CKPAGE
      LOGICAL   TIK_FLAG            ! TIME TICKS FOR DISPLAY
      INTEGER   ERR                 ! ERROR FLAG
      INTEGER   IVOICE              ! VOICE QUALITY
      REAL      VOIARG              ! VOICE QUALITY ARGUMENTS
      LOGICAL*1 EMPH                ! PRE-EMPHASIS FLAG
      INTEGER   LPCMEXT             ! LENGTH OF PCM EXTENSION
      INTEGER*2 SC_CHAN             ! SCA0 DEVICE CHANNEL
      REAL      AMP_MAX             ! MAX. AMPLITUDE VALUE
      CHARACTER*100 NAMESP          ! NAME OF SWS PCM FILE
      CHARACTER*4   PCMEXT          ! PCM EXTENSION
      CHARACTER*1   AMP_TYPE        ! AMPLITUDE: LINEAR OR DB
      CHARACTER*4   AUDIO           ! AUDIO SYSTEM
                                    !  NONE, DT, GRAD or SIOP
      INTEGER       INI_LUN         ! INITIALIZATION FILE LUN
      LOGICAL*1     ARGS            ! .TRUE. IF AT LEAST 1 ARGUMENT
      REAL          FPARS(12)           ! F.P. ARGUMENTS
      INTEGER       NUMMDEF             ! # OF MACROS CURRENTLY DEFINED
      CHARACTER     MDEF(MAXMDEF)*100   ! MACRO DEFINITIONS
      INTEGER       LMDEF(MAXMDEF)      ! LENGTH OF EACH MACRO DEF.
      LOGICAL   PCM_DT                  ! Data Translation PCM flag


C.....
C
C     End of SWS.INC
C
C=================================================================================
=======

      EXTERNAL CONTIG

      REAL FREQW(MAXSW)             ! working frequencies
      REAL AMPLW(MAXSW)             ! working amplitudes
      REAL RADIAN(MAXSW)            ! angle of sine waves in radians
      REAL FNYQL                    ! Nyquist freq. limit
      INTEGER   LEN                 ! total # samples
      INTEGER   NSMPIN              ! total # samples
      INTEGER   FOR_STAT
      INTEGER*2 IB(MXRECL)          ! record buffer
      LOGICAL RAMPA(MAXSW)          ! amplitude ramp (true or false)
      LOGICAL RAMPF(MAXSW)          ! frequency ramp (true or false)


      EQUIVALENCE (IB(2),LEN)       ! put len in IB(2) and (3)

C:::::
```

```
C
C    Begin routine
C
C:::::

     ERR = 0                            ! error flag
     FNYQL = ( SR/2. ) - 1.             ! Nyquist frequency

     IF ( .NOT. SWDAT ) THEN
       WRITE (TTO,15)
15     FORMAT ('0 *** SINE WAVE DATA DOES NOT EXIST !! ***',/)
       ERR = -1
       GO TO 9000
     END IF

C.....
C    Open new disk file to write pcm data to.
C.....

     LEN = TIMSL(NSLIS) * ( SR/1000. ) + .5       ! length
     NRECS = ( LEN + (MXRECL-1) ) / MXRECL        ! # PCM records
     NRECS = NRECS + 1                            ! extra leeway
     NDB = ( NRECS+3 ) / 4  + 1                   ! # disk blocks

     OPEN  (UNIT=PCFIL,NAME=NAMESP,TYPE='NEW',ACCESS='DIRECT',
     1       USEROPEN=CONTIG,
     2       RECORDSIZE=32,INITIALSIZE=NDB,ERR=50)
     SPOPEN = .TRUE.
     GO TO 100

C.....
C      FILE OPEN ERROR
C.....

50     WRITE (TTO,60)
60     FORMAT ('0 *** WRITE PCM OPEN DISK FILE ERROR !! ***')
       WRITE (TTO,62) PCFIL,NAMESP(1:LNSP)
62     FORMAT ('  *** FILE ON LUN ',I2,' IS: ',A,' ***',/)
       ERR = -1
       CLOSE (UNIT=PCFIL)
       GO TO 9000

C.....
C    Generate the sine waves from time slice to time slice
C.....

100  IBPTR = 0                          ! Buffer pointer
     NRECS = 0                          ! # of PCM records
     NSMPIN = 0

     IF ( NSLIS .LT. 2 ) THEN      ! Too few slices to work with
       WRITE (TTO,110)
110    FORMAT ('0 *** TOO FEW TIME SLICES !! ***',/)
       ERR = -1
       GO TO 9000
     END IF

     DO I = 1, NSW                 ! NSW is the # of sinewaves
       RADIAN(I) = 0.              ! Initialize radians
     END DO

C.....
C    If the first time slice does not start at 0. msec.,
```

```
C    output midscale data.
C.....

     IF ( TIMSL(1) .EQ. 0. ) GO TO 250

     NSMSL = TIMSL(1) * ( SR/1000. )
     IF ( NSMSL .LT. 1 ) GO TO 250

       DO J = 0, NSMSL-1
         IBPTR = IBPTR + 1
         IB(IBPTR) = MIDSCL

         IF ( IBPTR .GE. MXRECL ) THEN
           NRECS = NRECS + 1
           WRITE (PCFIL' NRECS+HEADER) IB
           IBPTR = 0
         END IF
       END DO

       NSMPIN = NSMPIN + NSMSL


250  DO 1000 I = 1,NSLIS-1

       IF ( TIMSL(I+1) .LE. TIMSL(I) ) GO TO 1000
       NSMSL = ( TIMSL(I+1) - TIMSL(I) ) * (SR/1000.)

       IF ( NSMSL .LT. 1 ) THEN
         WRITE (TTO,270) TIMSL(I),TIMSL(I+1)
270      FORMAT ('0 *** DURATION FROM ',F10.3,' MSEC TO ',F10.3,
     1        ' MSEC ')
         WRITE (TTO,272)
272      FORMAT ('      IS TOO SMALL FOR THIS SAMPLING RATE ',
     1       14X,'***',/)
         ERR = -1
         GO TO 1100
       END IF

       DO J = 1, NSW
         FREQW(J)  = SWF(I,J)      ! working frequency
         AWORK     = SWA(I,J)      ! working amplitude

C.....
C        Amplitude data is specified, as linear from 0 to 1.0.
C        A flag, called AMP_TYPE, can be set so that the
C        amplitude values are specified in DB.
C        If this flag is set, the routine SWS_DB is called.
C.....

         IF ( AMP_TYPE .EQ. 'D' ) CALL SWS_DB ('L',AWORK,AWORK)
         AMPLW(J)  = AWORK              ! working amplitude
         RAMPA(J)  = .FALSE.
         RAMPF(J)  = .FALSE.
         IF ( NSMSL .GE. 2 ) THEN
           IF ( SWF(I,J) .NE. SWF(I+1,J) ) RAMPF(J) = .TRUE.
           IF ( SWA(I,J) .NE. SWA(I+1,J) ) RAMPA(J) = .TRUE.
         END IF
       END DO


       DO J = 0, NSMSL-1

         XSUM = 0.
```

```
         DO K = 1, NSW

C.....          PERCNT is the percent advance into the current slice

           IF ( RAMPA(K) .OR. RAMPF(K) ) PERCNT = FLOAT (J) /
     1                                   FLOAT (NSMSL-1)

C.....          If changing, compute new frequency and amplitude.


           IF ( RAMPA(K) ) THEN
             AWORK  = SWA(I,K)
             AWORK2 = SWA(I+1,K)
             IF ( AMP_TYPE .EQ. 'D' ) THEN
               CALL SWS_DB ('L',AWORK,AWORK)
               CALL SWS_DB ('L',AWORK2,AWORK2)
             END IF
             AMPLW(K) = AWORK + PERCNT * ( AWORK2 - AWORK )
           END IF

           IF ( RAMPF(K) ) FREQW(K) = SWF(I,K) + PERCNT *
     1                         ( SWF(I+1,K)-SWF(I,K) )

C.....
C          Window frequency: MIN = 1 HZ; MAX = NYQUIST limit
C.....

           IF ( FREQW(K) .LT. 1. ) FREQW(K) = 1.
           IF ( FREQW(K) .GE. FNYQL ) FREQW(K) = FNYQL

C.....
C          Generate
C.....

           RADIAN(K) = RADIAN(K) + TWOPI * FREQW(K) / SR
           IF ( RADIAN(K) .GE. TWOPI ) RADIAN(K) =
     1                           AMOD(RADIAN(K),TWOPI)

           X = AMPLW(K) * SIN( RADIAN(K) )

           XSUM = XSUM + X

         END DO

         IBPTR = IBPTR + 1
         IOUT = IFIX ( XSUM/NSW * 2047. ) + MIDSCL
         IB(IBPTR) = IOUT

         IF ( IBPTR .GE. MXRECL ) THEN
           NRECS = NRECS + 1
           WRITE (PCFIL' NRECS+HEADER) IB
           IBPTR = 0
         END IF

       END DO

       NSMPIN = NSMPIN + NSMSL


1000 CONTINUE

1100 IF ( IBPTR .NE. 0 ) THEN
       DO I = IBPTR+1, MXRECL
```

```
          IB(I) = MIDSCL
        END DO
        NRECS = NRECS + 1
        WRITE (PCFIL' NRECS+HEADER) IB
      END IF

C.....
C     Rewrite header block.
C.....

      DO I = 1, MXRECL
        IB(I) = 0
      END DO

      DO IREC = 2, 4
        WRITE (PCFIL' IREC) IB
      END DO

      LEN = NSMPIN
      IB(1) = 1
      IB(4) = IFIX(SR)
      IF ( .NOT. EMPH ) IB(5) = 1
      IB(62) = IB(4)
      IB(63) = -32000

      NHEDER = HEADER

      WRITE (PCFIL' 1) IB


9000  IF ( SPOPEN ) CLOSE (UNIT=PCFIL)
      SPOPEN = .FALSE.


      RETURN
      END




      SUBROUTINE SWS_DB (TYPE,FIN,FOUT)
C
C          P. RUBIN  6/19/84
C
C
C     PURPOSE:
C
C       Convert between linear (0-1.) and
C       db (0-100db) values.
C
C
C     ARGUMENTS:
C
C       TYPE    CHARACTER*1    type of conversion, where:
C                               'L' = convert to linear
C                               'D' = convert to DB
C       FIN     REAL           value to be converted
C       FOUT    REAL           converted value
C
C================================================
```

```
      PARAMETER SCALE = 100000.          ! scale for 0 to 1
      PARAMETER FACDB = 50.              ! constant factor

      REAL        FIN                    ! value to be converted
      REAL        FOUT                   ! converted value
      CHARACTER*1 TYPE                   ! type of conversion


C:::::
C
C     BEGIN ROUTINE
C
C:::::

      IF ( TYPE .EQ. 'D' ) THEN

        FIN = FIN * SCALE                ! scale value up
        IF ( FIN .EQ. 0. ) THEN
          FOUT = 0.                      ! convert to db
        ELSE
          FOUT = 20. * ALOG10( FIN )
          FOUT = FOUT - FACDB            ! subtract constant
          IF ( FOUT .LT. 0. ) FOUT = 0.
        END IF

      ELSE

        FTMP = FIN + FACDB               ! add constant factor
        FOUT = 10.** (FTMP/20.)          ! convert to linear
        FOUT = FOUT / SCALE
      END IF

      RETURN
      END
```

Appendix III: MATLAB code

Dan Ellis created a *MATLAB* version of *SWS* in 1996 while he was at the International Computer Science Institute, Berkeley, CA. Steve Frost and Philip Rubin, of Haskins Laboratories, have created a *MATLAB* version of *SWS*, based on Dan's *MATLAB* routines. Dan has modified his *MATLAB* version of *SWS* to include an integrated analysis component. Additional information can be found on his website (http://www.ee.columbia.edu/~dpwe/). We reproduce a portion of the Dan Ellis *MATLAB* code here.

```
The Haskins site includes several example analysis files that you can
download.  These files contain, in a compact form, all the data you need to
resynthesize the sinewave speech.  The MATLAB routines below do this for you.

*      README - usage details
*      synthtrax.m - the main synthesis routine
*      slinterp.m - subsidiary linear interpolation routine
*      readswi.m - function to read the SWI-format data files into Matlab
*      s1pars.swi, s6pars.swi - example paramters files from the Haskins site.
```

```
README for ~dpwe/matlab/sws-1996aug

dpwe@icsi.berkeley.edu 1996aug23

This directory contains functions for regenerating sinewave speech
from the parameters published by Philip Rubin of Haskins Labs at

  http://www.haskins.yale.edu/Haskins/MISC/SWS/sentences.html

Sinewave speech is an unexpectedly intelligible analog of speech
where three or four formant tracks are reproduced by sine tones
alone.  That these combinations evoke a phonetic perception raises
very deep questions about the nature of auditory organization.

You can reproduce the sound examples on the web page, as well as
experimenting with manipulations such as removing or altering
certain components, using the functions in this directory.
The *.swi files define the frequency and amplitudes for the
formant tones, and are downloaded from the web site (there are
nine examples there, S1pars.swi - S9pars.swi).

This directory contains three matlab scripts:

  X = synthtrax(F, M, SR, SUBF, DUR)      Reconstruct a sound from track rep'n.
        Each row of F and M contains a series of frequency and magnitude
        samples for a particular track.  These will be remodulated and
        overlaid into the output sound X which will run at sample rate SR,
        although the columns in F and M are subsampled from that rate by
        a factor SUBF.  If DUR is nonzero, X will be padded or truncated
        to correspond to just this much time.

  Y = slinterp(X,F)  Simple linear-interpolate X by a factor F
         Y will have ((size(X)-1)*F)+1 points i.e. no extrapolation
```

```
  [F,M] = readswi(NAME)  Read a Haskins-format sinewave speech data file
      NAME is the name of a text data file containing the frequency
      and magnitude parameters for sinewave synthesis.  Result is
      F and M matrices suitable for synthtrax.m

You use them like this (within matlab):

>>   % Read in arrays defining the frequency and magnitude of the oscillators:
>> [F,M] = readswi('S1pars.swi');
>>   % Each row of F and M defines a single oscillator.  Columns are uniformly
>>   % spaced time samples.
>>   % Synthesize an audio signal based on the parameters:
>> X = synthtrax(F,M,8000,80);    % Takes 1.05s on Sparc5, 17s on Duo270c
>>   % X is the output of oscillators controlled by F and M.  Its sampling
>>   % rate is 8000 Hz, and the control columns were interpolated by a
>>   % factor of 80 before synthesis (i.e. 100 Hz control rate).
>>   % Play the sound at 8000 Hz SR (on a sun, mac, sgi...):
>> sound(X,8000)
>>   % We can also synthesize each track separately by selecting rows
>>   % in F and M:
>> T1 = synthtrax(F(1,:), M(1,:), 8000, 80);
>> T2 = synthtrax(F(2,:), M(2,:), 8000, 80);
>> T3 = synthtrax(F(3,:), M(3,:), 8000, 80);
>>   % Listen to combinations of the sine tones:
>> sound(T1, 8000);
>> sound(T1+T2, 8000);
>> sound(T1+T3, 8000);
>> sound(T1+T2+T3, 8000);
>>   % etc...
>>   % We can even look at the oscillator tracks:
>> plot(F')
>>   % Maybe force the frequency to zero when magnitude is zero
>> plot((F.*(M~=0))')

This code has been tried under Matlab 4.2c running on a Solaris 2.5
SPARCstation and Matlab 4.1 running on a MacOS 7.5.1 Powerbook Duo 270c.

Robert Remez, Philip Rubin and others have published a large series of
papers on the nature of this strange phenomenon (the references are at
the web site mentioned above).  You will be able to reproduce their
stimuli, including reversed formant tracks, dichotic presentation
(given a stereo playback extension such as SoundMex4.1) etc.  Have fun.
```

```
function X = synthtrax(F, M, SR, SUBF, DUR)
% X = synthtrax(F, M, SR, SUBF, DUR)      Reconstruct a sound from track rep'n.
%      Each row of F and M contains a series of frequency and magnitude
%      samples for a particular track.  These will be remodulated and
%      overlaid into the output sound X which will run at sample rate SR,
%      although the columns in F and M are subsampled from that rate by
%      a factor SUBF (default 128).  If DUR is nonzero, X will be padded or
%      truncated to correspond to just this much time.
% dpwe@icsi.berkeley.edu 1994aug20, 1996aug22

if(nargin<4)
  SUBF = 128;
end

if(nargin<5)
  DUR = 0;
end
```

```
rows = size(F,1);
cols = size(F,2);

opsamps = round(DUR*SR);
if(DUR == 0)
  opsamps = 1 + ((cols-1)*SUBF);
end

X = zeros(1, opsamps);

for row = 1:rows
%  fprintf(1, 'row %d.. \n', row);
  mm = M(row,:);
  ff = F(row,:);
  % Where mm = 0, ff is undefined.  But interp will care, so find points
  % and set.
  % First, find onsets - points where mm goes from zero (or NaN) to nzero
  % Before that, even, set all nan values of mm to zero
  mm(find(isnan(mm))) = zeros(1, sum(isnan(mm)));
  ff(find(isnan(ff))) = zeros(1, sum(isnan(ff)));
  nzv = find(mm);
  firstcol = min(nzv);
  lastcol = max(nzv);
  % for speed, chop off regions of initial and final zero magnitude -
  % but want to include one zero from each end if they are there
  zz = [max(1, firstcol-1):min(cols,lastcol+1)];
  mm = mm(zz);
  ff = ff(zz);
  nzcols = prod(size(zz));
  mz = (mm==0);
  mask = mz & (0==[mz(2:nzcols),1]);
  ff = ff.*(1-mask) + mask.*[ff(2:nzcols),0];
  % Do offsets too
  mask = mz & (0==[1,mz(1:(nzcols-1))]);
  ff = ff.*(1-mask) + mask.*[0,ff(1:(nzcols-1))];
  % Ok. Can interpolate now
  % This is actually the slow part
%  % these parameters to interp make it do linear interpolation
%  ff = interp(ff, SUBF, 1, 0.001);
%  mm = interp(mm, SUBF, 1, 0.001);
%  % chop off past-the-end vals from interp
%  ff = ff(1:((nzcols-1)*SUBF)+1);
%  mm = mm(1:((nzcols-1)*SUBF)+1);
  % slinterp does linear interpolation, doesn't extrapolate, 4x faster
  ff = slinterp(ff, SUBF);
  mm = slinterp(mm, SUBF);
  % convert frequency to phase values
  pp = cumsum(2*pi*ff/SR);
  % run the oscillator and apply the magnitude envelope
  xx = mm.*cos(pp);
  % add it in to the correct place in the array
  base = 1+SUBF*(zz(1)-1);
  sizex = prod(size(xx));
  ww = (base-1)+[1:sizex];
  X(ww) = X(ww) + xx;
end
```

```
function Y = slinterp(X,F)
```

```
% Y = slinterp(X,F)  Simple linear-interpolate X by a factor F
%        Y will have ((size(X)-1)*F)+1 points i.e. no extrapolation
% dpwe@icsi.berkeley.edu  fast, narrow version for SWS

% Do it by rows

sx = prod(size(X));

% Ravel X to a row
X = X(1:sx);
X1 = [X(2:sx),0];

XX = zeros(F, sx);

for i=0:(F-1)
  XX((i+1),:) = ((F-i)/F)*X + (i/F)*X1;
end

% Ravel columns of X for output, discard extrapolation at end
Y = XX(1:((sx-1)*F+1));
```

```
function [F,M] = readswi(NAME)
% [F,M] = readswi(NAME)  Read a Haskins-format sinewave speech data file
%     NAME is the name of a text data file containing the frequency
%     and magnitude parameters for sinewave synthesis.  Result is
%     F and M matrices suitable for synthtrax.m
% dpwe@icsi.berkeley.edu 1996aug22

% SWI files are downloaded from
%    http://www.haskins.yale.edu/Haskins/MISC/SWS/sentences.html
% and have the format:
%    Number of oscillators
%      Time0
%         frq,mag   for 1st oscillator
%         frq,mag   for 2nd oscillator
%         .. for as many oscillators as specified
%      Time1
%         frq,mag  ... etc.
% Times are in ms, frq in Hz, mag in linear units
%
% Here, we're assuming the times are uniformly spaced, and it is
% up to the user to know the correct interpolation factor to
% give to synthtrax.
%
% BE SURE TO TRIM OFF THE TEXT AT THE TOP AND BOTTOM OF THE FILES
% IF YOU SAVE DIRECTLY FROM THE WEB PAGES!

colchunk = 100;
col = 0;

fid = fopen(NAME, 'r');
if (fid == -1)
  fprintf(1, 'readswi: unable to read %s\n', NAME);
else
  nOscs = fscanf(fid, '%d', 1);
  % Increase the arrays in chunks of colchunk cols to avoid slow
  % matrix growing.
  emptyF = zeros(nOscs, colchunk);
  F = emptyF;
```

```
  M = emptyF;
  Fcols = colchunk;

  endoffile = 0;
  while (endoffile == 0)
    [time,count] = fscanf(fid, '%f', 1);
    if (count == 0)
      endoffile = 1;
    else
      col = col+1;
      if(col > Fcols)
       % We ran out of empty columns - grow the matrices
       F = [F, emptyF];
       M = [M, emptyF];
       Fcols = Fcols + colchunk;
      end
      for osc = 1:nOscs
       F(osc,col) = fscanf(fid, '%f,', 1);
       M(osc,col) = fscanf(fid, '%f',1);
      end
    end
  end
  fclose(fid);
  % Trim off excess empty columns
  F = F(:,1:col);
  M = M(:,1:col);
end
```

Philip Rubin, Haskins Laboratories, New Haven, CT 06511
rubin@haskins.yale.edu, 203-865-6163
and Yale University School of Medicine, Department of Surgery, Otolaryngology