CDP Functions to COMBINE Spectral Data

(with Command Line Usage)

Functions to COMBINE spectra from different sounds

COMBINE **CROSS**
    Replace spectral amplitudes of 1st file with those of 2nd
COMBINE **DIFF**
    Subtract one spectrum from another
COMBINE **INTERLEAVE**
    Interleave (groups of) windows of several spectra
COMBINE **MAKE**
    Generate spectrum from pitch & formant data
COMBINE **MAKE2**
    Generate spectrum from pitch, formant & envelope data
COMBINE **MAX**
    In each analysis channel, in each window, take the maximum value amongst the input files
COMBINE **MEAN**
    Generate the spectral 'mean' of 2 sounds
SPECROSS
    Interpolate partials of pitched *inanalfile1* towards those of pitched *inanalfile2*
SPECSPHINX
    Impose the channel amplitudes of inanalfile2 onto the channel frequencies of inanalfile1
SPECTWIN
    Combine the formant and/or total spectral envelopes of two spectra
COMBINE **SUM**
    Find the sum of two spectra

Technical Discussion
    Interacting Spectra

SEE ALSO:
[FASTCONV]
    Multi-channel FFT-based convolution

COMBINE CROSS – Replace spectral amplitudes of 1st file with those of 2nd

Usage

combine cross *infile infile2 outfile* [**-i***interp*]

Parameters

infile, infile2 – input analysis files made with PVOC
outfile – output analysis file
-i*interp* – the degree of replacement (Range: 0 to 1)

interp may vary over time

Understanding the COMBINE CROSS Process

Formerly SPECCROS, this function makes a weighted substitution of the amplitude of one spectral envelope for that of the other. It is therefore a fairly straightforward way to achieve a crossover between the timbral characteristics of two different sounds.

Musical pplications

The spectral envelope contains the data for spectral peaks/formants. Thus the transfer of timbral characteristics. However, it is not a full substitution of the whole envelope, but only a (weighted) crossover. It can therefore be used, for example, to prepare two sounds for morphing, especially when the timbral characteristics differ to a considerable extent. By moving one sound gradually into the timbral field of the other prior to morphing, a smoother morph transition can be achieved.

End of COMBINE CROSS

COMBINE DIFF – Find difference of two spectra

Usage

combine diff *infile infile2 outfile* [**-c***crossover*] [**-a**]

Parameters

infile, infile2 – input analysis files made with PVOC
outfile – output analysis file
-c*crossover* – the amount of the 2nd spectrum subtracted from the 1st (Range: 0 to 1)

crossover may vary over time

-a retain any subzero amplitudes produced (Default: set these to zero)

Understanding the COMBINE DIFF Process

In COMBINE DIFF one spectrum is subtracted from another on a channel-by- channel basis (weighted by *crossover*). If amplitude in any channel goes negative, you can opt whether or not to retain the negative amplitudes.

By default, the setting of any amplitudes < 0 to 0 protects against negative overflow, but you can use the **-a** flag and see what happens.

Musical Applications

This is a way to change a sound on a very experimental basis.

Also see: **COMBINE SUM**.

End of COMBINE DIFF

COMBINE INTERLEAVE – Interleave (groups of) windows of several spectra

Usage

combine interleave *infile infile2 [infile3 ...] outfile leafsize*

Parameters

infile(s) – an number of input files made with PVOC (at least two), which will be processed sequentially in the same order as given on the command line
outfile – output analysis file
leafsize – number of analysis windows in each leaf (group of windows)

Understanding the COMBINE INTERLEAVE Process

COMBINE INTERLEAVE interleaves *N* spectra from *N* analysis files, thereby constructing a new analysis file as output. The spectral characteristics of several sounds are thus amalgamated into one new sound.

The degree to which the original sounds are retained depends on the number of analysis windows used.

A *leafsize* is simply a specified group of analysis windows. If *leafsize* is 1, then the windows from each file are interleaved one at a time. If *leafsize* is 2, then the windows from each file are interleaved two at a time, etc. Larger *leafsize* retains more recognisable chunks of the original sounds.

Musical Applications

With very small values for *leafsize*, the various sounds are mingled quite thoroughly and relatively smoothly – relative to the nature of the input sounds, though a somewhat grainy effect may result. With larger values for *leafsize*, say 8, 10, 20 or more, more of each component sound is heard in each leaf and the result is more like a churning or pulsating effect as the new soundfile oscillates among the sound material in the various source soundfiles. To churn means to shake, stir or agitate violently, as when making butter from milk, which is nicely descriptive of what happens when several sound sources, each fairly strong, are interleaved in this way. [It's one of my favourites! – A.E].

End of COMBINE INTERLEAVE

COMBINE MAKE – Generate spectrum from pitch & formant data only

Usage

combine make *pitchfile formantfile outfile*

Parameters

pitchfile – a binary pitch data file (**.frq**) as produced by REPITCH GETPITCH, REPITCH GENERATE, REPITCH COMBINE Mode **2** or BRKTOPI
formantfile – a binary formant data file (**.for**) as produced by FORMANTS GET
outfile – analysis file (**.ana**) which can be resynthesized with PVOC

Understanding the COMBINE MAKE Process

MAKE combines a binary pitch data file (**.frq**) with a formant (spectral envelope) file (**.for**) to make a spectral file (**.ana**): **.frq** + **.for** = **.ana**.
Both input files must have been derived from the same PVOC settings.

COMBINE MAKE provides the standard route back to sound, following the processes of the REPITCH suite. REPITCH GETPITCH extracts time-varying pitch data from a spectral analysis file and the REPITCH functions offer a number of ways to manipulate this information. Pitch-breakpoint files can also be edited manually and converted to binary pitch files (**.frq**) via the function BRKTOPI, or the latter can be synthesised by REPITCH GENERATE. To get back to sound, the new pitch data file must be combined with a formant file (**.for**).

COMBINE MAKE is also an interesting way to combine the pitch data and the timbral (formant) data from any two arbitrarily chosen sounds.

End of COMBINE MAKE

COMBINE MAKE2 – Generate a spectrum from only pitch, formant and envelope data

Usage

combine make2 *pitchfile formantfile envfile outfile*

Parameters

pitchfile – a binary pitch data file (**.frq**) as produced by REPITCH GETPITCH, REPITCH GENERATE or REPITCH COMBINE, Mode **2**
formantfile – a binary formant data file (**.for**) as produced by FORMANTS GET
envfile – a binary envelope file (**.evl**) produced / modified in the Time Domain or derived from an analysis file (.ana) by REPITCH ANALENV
outfile – an analysis file **.ana** that can be resynthesised with PVOC

Understanding the COMBINE MAKE2 Process

.frq + **.for** + **.evl** = **.ana**
This function can be used to combine the pitch contour, formant envelope, and loudness envelope of different sounds, thereby transferring the characteristics of one sound onto another. MAKE2 extends the functionality of MAKE by enabling (spectral) envelope data to be used as well.

The source sound could also be recreated from the data in the pitch, formant and envelope files extracted from its own spectrum. This provides a way to check on the accuracy of the extraction – small details, especially of noise sounds, often get changed.

Musical Applications

New sounds can be engineered with this procedure by combining three different features of up to three different sounds. The pitch data file contains an extracted pitch contour, the formant data file contains the strongest, most resonant, frequency bands of the spectrum, and the envelope file contains the contour of the spectral envelope, the (changing) amplitude pattern made by the frequencies. Given that you are alert to the distinctive characteristics of your sounds, this function provides a tremendously powerful tool with which to construct new sonic material.

This is important not just from a purely sonic point of view. It is also an important tool for creating structural relationships between sounds, that is, the duplication of identical or similar shapes within a composition. This can create obvious relationships or it can be used to create subtle, hardly perceptible connections. Overall, the result is to achieve a sense of unity within the composition, even when diverse materials are being used. We are familiar with this in note-based music, where motifs and themes can have similar components, contours etc. COMBINE MAKE and COMBINE MAKE2 therefore provide a vital compositional tools in the spectral realm.

Recreating the original sound from its own pitch, formant and envelope data provides a diagnostic tool for assessing how the analysis is coping with the frequency contents of a sound.

End of COMBINE MAKE2

COMBINE MAX – In each analysis channel, in each window, take the maximum value amongst the input files

Usage

combine max *infile infile2 [infile3 ...] outfile*

Parameters

infile, infile2 [infile3 ...]– arbitrary number of input analysis files
outfile– output analysis file

Understanding the COMBINE MAX Process

COMBINE MAX studies the data in all the input analysis files. For each channel it retains only the single loudest spectral component *among all the input files*. In spite of the loss of the other components, the sound often remains the same to a remarkable degree. But how several sounds will merge in this way will be a little unpredictable: because data from the files which DO NOT have the loudest spectral component will simply be omitted.

The spectral characteristics of the input sounds are just merged, producing an ultra-seamless mix. The input sounds remain recognisable, just like an ordinary mix. Note that this is different from the programs which transfer the spectrum or spectral envelope of a sound. With the latter, the audible characteristics of one of the sounds are noticeably changed.

Rob Waring's wonderful *hornwhivoc* sound (included on the *Future Music* cover CD for December 1993), which we often use for demos, was made with this function. It merges a horn, a whistle and a voice – timbrally very different sounds.

End of COMBINE MAX

COMBINE MEAN – Generate the spectral 'mean' of 2 sounds

Usage

combine mean mode *infile infile2 outfile* [**-l***lofrq*] [**-h***hifrq*] [**-c***chans*] [**-z**]

Modes

1 mean channel amp of 2 files : mean of two pitches
2 mean channel amp of 2 files : mean of two frequencies
3 channel amp from file 1 : mean of two pitches
4 channel amp from file 1 : mean of two frequencies
5 channel amp from file 2 : mean of two pitches
6 channel amp from file 2 : mean of two frequencies
7 max channel amp of 2 files : mean of two pitches
8 max channel amp of 2 files : mean of two frequencies

Parameters

infile, infile2 – input analysis files made with PVOC
outfile – output analysis file
-l*lofrq* – low frequency limit of channels to look at
-h*hifrq* – high frequency limit of channels to look at
-c*chans* – number of significant channels to compare. Default: ALL within range.
-z zeroes channels OUTSIDE the frequency range specified

Understanding the COMBINE MEAN Process

This program takes 2 input sounds and calculates the average of the 2 signals, spectral window by spectral window. The average in loudness between, for example, two speech sounds, should reach some vague mean between the formants of the speech components being used. The frequency average is altogether more strange. It may, for example, convert harmonic into inharmonic spectra. Seriously wierd.

Musical Applications

It is not really possible to predict the results of this way to combine sounds, as they are so dependent on the nature of the sources. It is best to try it with the type of source file you favour and try to build up some experimental data about what happens.

End of COMBINE MEAN

SPECROSS PARTIALS – Interpolate partials of pitched *inanalfile1* towards those of pitched *inanalfile2*

Usage

specross partials *inanalfile1 inanalfile2 outanalfile tuning minwin signois harmcnt lo hi thresh level interp*

Example command line to interpolate pitched partials between analysis files:

```
specross partials p3c.ana horn2.ana p3xh2.ana 4 5 80 5 50 2000 0.2 1 0.2
```

Parameters

inanalfile1 – first input analysis file towards which the partials are interpolated
inanalfile2 – second input analysis file from which the partials are derived
outanalfile – output analysis file with the partials from the second file interpolated towards those of the first
tuning – the range in semitones within which the harmonics are 'in tune'; higher values are more 'forgiving' and use more of the second file (Range: 0.0 to 6.0, Default: 1)
minwin – the minimum number of adjacent windows that must be pitched, for a pitch-value to be registered (Default: 2)
signois – the signal-to-noise ration, in decibels (Default 80dB). Windows that are greater than *signois*dB below the maximum level in the sound are assumed to be noise, and any detected pitch is assumed to be spurious.
harmcnt – The number of the 8 loudest peaks in the spectrum which much be harmonics in order to confirm that the sound is pitched (Default 5)
lo – the lowest acceptable frequency for a pitch (Default: 10Hz)
hi – the highest acceptable frequency for a valid pitch (Default: Nyquist/8, i.e., 2756.25 at a sample rate of 44100: sample rate divided by 2 and then divided by 8: 44100 ÷ 2 = 22050 ÷ 8 = 2756.25. The program adjusts *hi* according to the sample rate of the input file.)
thresh – the minimum acceptable level of any partial found, if it is to be used in reconstructed spectrum (level relative to loudest partial). A lower value is more 'forgiving' and uses more of the second file. A higher value accepts only the loudest partials.
level – the level of the output (Default: 1.0). Use if reapplying *inanalfile1* to several *inanalfile2s*, whose relative level is important.
interp – the interpolation between *inanalfile2* and *inanalfile1*, which can vary through time. Breakpoint time values will be scaled to the duration if *inanalfile1*. A lower value reduces the effect of *inanalfile2*, i.e., it 'lets through' more of a recognisable second file, rather than just its partials. (Default? Range?)
-a – retain the loudness contour of *inanalfile2*, under the contour of *inanalfile1*. What does this mean?
-p – extend the first stable pitch of *inanalfile1* to the start of the *outanalfile*

Understanding the SPECROSS PARTIALS Process

The importance of SPECROSS PARTIALS is that it is focusing on **pitched** sounds. An interpolation is being between one sound and another: in this case, of the harmonic partials, i.e., those that are integer multiples. These are heard as pitches because the partials fuse into a single (pitched) entity. This is a welcome addition to the CDP transition programs, because the others deal with any partial, whether harmonic or inharmonic.

The harmonically-related partials taken from the pitched source in effect form a colouration, not unlike a formant. This colouration is taken from the second sound and imposed on the first, thus re-colouring it. If the *thresh* and *interp* parameters are set to high values, the partials-colouration is so strong that there is virtually nothing left audible from the rest of sound-2, only the colouration comes across to sound-1. If the *thresh* and *interp* parameters are on the low side, some of the rest of the sound will come across as well. The example command line uses lower values.

Other parameters such as *tuning* provide ways to fine-tune just how the harmonic qualities of sound2 are picked up. A lower value here is more 'forgiving' and is likely to use more partials.

Musical Applications

This program provides a tool to work with recolouring pitched sounds. The various parameters enable you to define how intensely this is done, how focused the result is on the harmonically related partials, and how much else of the second sound is used, making its presence more audible in the output sound.

Although designed for use with two pitched sounds, I tried recolouring the vocal file *count.ana* with the sound of a horn (*horn2.ana*, which was *horn.ana* + *horn.ana* joined with ANALJOIN JOIN). The result was a highly modulated vocal sound with a significant amount of horn colouration – a rather interesting result.

Also see a number of other programs that deal with data transfers between sounds:

- MORPH MORPH – to make gradual transitions between the spectrum of one file and that of another.
- NEWMORPH/2 – to make gradual transitions between the spectral peaks of one file and that of another.
- MORPH BRIDGE – to make a gradual transition from one time-specified analysis window to another.
- MORPH GLIDE – to make a gradual transition from a single analysis window to another.
- FORMANTS VOCODE – to impose the formants from the second sound onto the first.
- COMBINE CROSS – to replace the channel amplitudes of the first file with those of the second.

It may be useful to compare the results obtained with SPECROSS PARTIALS with those of HILITE TRACE, which retains the *N* loudest partials – any partial, without constraining them to harmonically related (i.e., pitched) partials.

End of SPECROSS PARTIALS

SPECSPHINX – Impose channel amplitudes of *inanalfile2* onto the channel frequencies of *inanalfile1* OR Multiply the spectra

Usage

specsphinx specsphinx 1 *inanalfile1 inanalfile2 outanalfile* [**-a***ampbalance*] [**-f***frqbalance*]
OR:
specsphinx specsphinx 2 *inanalfile1 inanalfile2 outanalfile* [**-b***bias*] [**-g***gain*]

Example command line to impose channel amplitudes:

```
specsphinx specsphinx 1 in1.ana in2.ana out.ana -b0.5 -f0.5
```

Modes

1 Impose channel amplitudes of *inanalfile2* onto the channel frequencies of *inanalfile1*
2 Multiply the spectra

Parameters

inanalfile1 – input analysis file 1
inanalfile2 – input analysis file 2
outanalfile – output analysis file
-a*ampbalance* – proportion of *inanalfile1*'s amplitudes that are retained. Default = 0
-f*frqbalance* – proportion of *inanalfile2*'s frequencies injected into the output spectrum. Default = 0
-b*bias* – When non-zero, it adds a proportion of the original files to the output:
< 1: add *inanalfile1* to *outanalfile* in the proportion determined by the formula `-bias/(1 + bias)`
> 1: add *inanalfile2* to *outanalfile* in the proportion determined by the formula `bias/(1 - bias)`
-g*gain* – overall gain applied to the output

ampbalance, *frqbalance*, *bias* and *gain* may vary over time

Understanding the SPECSPHINX Process

SPECSPHINX is one of three experimental programs designed to create spectra that are intermediate between two given spectra. (The other two are NEWMORPH and SPECTWIN.

Mode 1 is related to COMBINE CROSS, which replaces the channel amplitudes with those of Infile2. (Here, *ampbalance* sets the proportion of amplitudes to be retained.)

End of SPECSPHINX

SPECTWIN – Combine the formant and/or total spectral envelopes of 2 spectra

Usage

spectwin spectwin 1-4 *inanalfile1 inanalfile2 outanalfile* [**-f***frqint*] [**-e***envint*] [**-d***dupl* **-s***step* **-r***dec*]

Example command line to create combined spectra:

```
spectwin spectwin 1 in1.ana in2.ana out.ana
```

Modes

1 Formant envelope of *inanalfile1* with the formant envelope of *inanalfile2*
2 Formant envelope of *inanalfile1* with the total envelope of *inanalfile2*
3 Total envelope of *inanalfile1* with the formant envelope of *inanalfile2*
4 Total envelope of *inanalfile1* with the total envelope of *inanalfile2*

The **formant envelope** traces out the envelope formed by the peaks in the spectrum.
The **total envelope** traces out the envelope formed by EVERY channel in the spectrum.

Parameters

inanalfile1 – input analysis file 1
inanalfile2 – input analysis file 2
outanalfile – output analysis file
-f*frqint* – dominance of the spectral frequencies of *inanalfile2*. (Range: 0 to 1. Default: 1.0)
-e*envint* – dominance of the spectral envelope of *inanalfile2*. (Range: 0 to 1. Default: 1.0)

Note that the next three optional parameters are linked.:
-d*dupl* – duplicate the sound of *inanalfile1 dupl* times, at higher pitches (Range: 0 to 8)
-s*step* – the transposition step in (fractions of) semitones at each duplication (Range: 0 to 48)
-r*dec* – the amplitude level multiplier from one transposition to the next when creating the transposed duplications (Range: 0 to 1)

Understanding the SPECTWIN Process

SPECTWIN creates hybrid sounds by combining the formant envelopes and/or total spectral envelopes of two files. *frqint* and *envint* specify the dominance of *inanalfile2*'s spectral frequencies and spectral envelope, respectively. Various hybrids can be formed by reducing the effect of one of these. Roughly speaking, if *envint* is at maximum and *frqint* is at minimum, the result is a transformation of *inanalfile1* (heavily influenced by *inanalfile2*); if the reverse, the effect is a transformation of *inanalfile2*.

You can optionally duplicate Infile 1 *dupl* times at higher pitches; if chosen, the interval is set as *step* semitones and *dec* determines how much each duplicate should be attenuated (1 = no attenuation).

Musical Applications

This function can create a wide variety of hybrids between two sounds. If both sounds are the same, the duplication feature on its own is virtually the spectral equivalent of **MODIFY STACK**: in Modes 3 or 4, set *frqint* and *envint* to 0, and *dec* to a high value. This method can be used to add extra brightness to the timbre.

If one source file is longer than the other, the outfile length is that of the shorter file. (R.F.)

End of SPECTWIN

COMBINE SUM – Find the sum of two spectra

Usage

combine sum *infile infile2 outfile* **-c***crossover*

Parameters

infile – input analysis file made with PVOC
outfile – output analysis file
-c*crossover* *crossover* is the amount of 2nd spectrum which is added to the 1st (Range 0 to 1)

crossover may vary over time

Understanding the COMBINE SUM Process

COMBINE SUM examines the amplitude levels in each channel of each of the two input analysis files. Where they differ, it **adds** this difference to the 1st file, as weighted by *crossover*. In other words, the result is *infile* plus the **difference** between it and *infile2*. When resynthesising, watch out for clipping.

Musical Applications

This is a way to adjust the timbre of a sound by adding to it some part of of the timbral features which belong to another sound. Very often, the change is both major and relatively unpredictable.

Also see: **COMBINE DIFF**.

End of COMBINE SUM

Technical Discussion of Interacting Spectra

As mentioned in the text for MAKE2, there are both sonic and compositional reasons for combining the features of different sounds.

Another application is for morphing, to create intermediate stages when morphing from one sound to another. This presumes that a morph is meant to be perceptually different from a transition. The latter implies a smooth changeover from the first to the second. A morph, however, can imply a gradual and perceptible *reshaping* of the first with the features of the second. All of the COMBINE functions can be explored with this in mind: sums (SUM), differences (DIFF), means (MEAN) of sounds, unpredictable mixes of the loudest frequencies (MAX), combinations of pitch traces, resonant frequencies and spectral amplitude contours MAKE and MAKE2), and even an intermediate stage that interleaves the two files (INTERLEAVE).

End of Technical Discussion