CDP FORMANTS functions

(with Command Line Usage)

Functions which Get and Use FORMANT Data

FORMANTS GET
Extract evolving formant envelope from an analysis file
FORMANTS GETSEE
Get formant data from an analysis file to a pseudo-soundfile to view in VIEWSF
FORMANTS PUT
Impose formants in a formant data file on data in an analysis file
FORMANTS SEE
Convert formant data in formant data file to a pseudo soundfile to fiew in VIEWSF
FORMANTS VOCODE
Impose spectral envelope of one sound on another
Technical Discussion
Formants and Spectral Envelopes

ALSO SEE:
COMBINE MAKE
Generate spectrum from pitch & formant data
ONEFORM Group
Operations with single formants

FORMANTS GET – extract evolving formant envelope from an analysis file

Usage

formants get *infile outfile* **-f***N* | **-p***N*

outfile is a binary formant file, which can be used in other formant applications

Parameters

infile an analysis file
outfile a binary formant file
-f extract formant envelope 'linear frequency-wise', using 1 point for every *N* equally-spaced frequency-channels
-p extract formant envelope 'linear pitchwise', using *N* equally-spaced pitch-bands per octave

Understanding the FORMANTS GET Process

This is a straightforward utility to extract and store the evolving spectral envelope of a sound. In earlier CDP systems it was called SPECFGET.

On Extracting Formants
The two **extraction options** (**-f***N* | **-p***N*) relate to resolution. Essential information:
the logarithmic nature of the pitch scale means that as one goes higher, an increasingly wider band of frequencies is encompassed by the 'octave'. The octave of a tone is double that tone's frequency. Thus the octave between 220Hz and 440Hz encompasses 220Hz, while the octave between 2200Hz and 4400Hz encompasses 2200Hz.

- **-f***N* – 'linear frequency-wise' is an 'equal spacing' comprising *equal differences*.
- **-p***N* – 'linear pitchwise' is an 'equal spacing' comprising *equal ratios*.

With 'linear frequency-wise', the analysis channels define frequency bands of identical bandwidth: each band will encompass the same frequency range.

With 'linear pitchwise', the frequency bands actually get wider as one goes higher. For example, if linear pitchwise is set at *N* = 12, the octave between 220Hz and 440 Hz is divided into 12 semitones; if *N* = 3, the octave is divided into 4 minor thirds (each of slightly increasing bandwidth). In the octave between 2200Hz and 4400Hz these same divisions will consist in considerably wider bandwidths. In other words, linear pitchwise is working via ratios with the logarithmic relationship of pitches.

Linear frequency-wise should therefore be chosen if the main formant area of interest is in the *higher frequencies*, because there will be better resolution here when extracting the spectral envelope. But if the main interest in the sound is in the *lower frequencies*, then the **linear pitchwise** option will be suitable: the frequency bandwidths relate to the octave bandwidths. You are also able more easily to think in terms of normal musical terminology for intervals. If in doubt, you'll need to experiment.

Linear frequency-wise and linear pitchwise

	Linear frequency-wise	Linear pitchwise
'Equal' means:	frequency spacing is the same *N* = **2**	octave divisor is the same *N* = **12**
E.g., band is 220Hz	86Hz (e.g,) encompassed by every **2** channels	**12** semitones within that octave
E.g., band is 2200Hz	exactly the same bandwidth	**12** semitones within that octave
Recommended application	better resolution with higher frequencies	effective with lower frequencies

The nature of the source, we should also observe, affects the results.

- With a noisy source, there is energy all over the signal. It is therefore easy to find the envelope of the source and extract the formants.
- With a pitched source, especially a high voice, there will be just a few partials with significant energy and, although these lie under the formant envelope, there is often not enough data to get a good indication of where the whole formant is.
- A sonogram is not really the best way to identify formants, says Richard Dobson. Better would be a per frame FFT-style display of amplitude/frequency along with a display of the spectral envelope. The Snack-based sonogram display available in *Sound Loom* does not have a facility for selecting a frequency band directly from it. We would recommend **Audacity** for inspecting the spectrum of a sound and picking out a formant region.
- We are currently exploring peak tracking and ConstQ Pvoc as ways to improve pitched partials and formants extraction.
- For more detailed information and a software option, please see **Prosoniq** .

Formant extraction and preservation is therefore a difficult process and ideal results are not obtained in every case.

Musical Applications

This process is essential when wanting to preseve the timbral qualities of a sound. They can be transferred to another sound, or they may be re-imposed on the original sound after it has undergone some other transformations. The re-imposing is done with **FORMANTS PUT**.

Also see: **FORMANTS PUT** and **FORMANTS GETSEE**.

End of FORMANTS GET

FORMANTS GETSEE – extract formants from analfile and write as 'soundfile' for viewing

Usage

formants getsee *infile outsndfile* **-f***N* **-p***N* [**-s**]

Parameters

infile an analysis file
outsndfile a pseudo-soundfile ready for viewing. **Do not resynthesize or play this** (an error message will be returned).
-f*N* extract formant envelope linear frequency-wise, using 1 point for every *N* equally-spaced frequency-channels
-p*N* extract formant envelope linear pitchwise, using *N* equally-spaced pitch-bands per octave
-s semitone bands for display (Default: equal Hz bands)

The resulting logarithmically scaled display indicates formant shapes, but NOT the absolute amplitude values.

Understanding the FORMANTS GETSEE Function

See a detailed discussion of the two extraction options in the text for FORMANTS GET for further information on this choice.

Previously a part of SPECFSEE, this process produces a pseudo soundfile which can be viewed in the same way as a normal soundfile. In the display we see a series of spectral windows, separated by zeroes. The horizontal display represents the spectral frequencies and this may be displayed in a manner which increases regularly with frequency (linear frequency-wise) or in a manner which increases regularly with pitch (linear pitch-wise). The vertical display represents the amplitude of each spectral band. The range of amplitudes is very large indeed, and in order to display them in the soundfile format, the log of the amplitude values is used and scaled to utilise fully the range of sound-sample values (-32768 to 32767).

The display therefore indicates only the shape of the spectrum, and how that shape changes through time - no absolute amplitude values can be read off from it.

FORMNTS GETSEE uses linear interpolation in determining the spectral contour. The output pseudo-soundfile is viewed by loading it into a soundfile display program, such as CDP's VIEWSF.

SPECIAL NOTE – It is important to realise that the output 'soundfile' is in this case a 'pseudo-soundfile'. All pseudo-soundfiles and binary files are displayed by DIRSF, and on PC are automatically given a .wav extension. It is therefore recommended that you adopt a naming convention that indicates which type of file it is, such as 'fmnt...' for the formant file produced by FORMANTS GET, 'psnd...' for the output of GETSEE, and 'bpdf...' for the binary pitch data file produced by PITCH. The different types of file all have their own 'property strings' written in the header, so the system will know what they are and prevent inappropriate use; the naming conventions are for your own convenience.

When a program produces a textfile, it is NOT given a .wav extension and will not be displayed by DIRSF. Look for textfiles in the current directory, with the command 'dir'.

Also see: **FORMANTS GET** and **FORMANTS PUT**.

End of FORMANTS GETSEE

FORMANTS PUT – impose spectral envelope in a formant file on spectrum in a PVOC analysis file

Usage

formants put 1 *infile fmntfile outfile* [**-i**] [**-l***lof*] [**-h***hif*] [**-g***gain*]
formants put 2 *infile fmntfile outfile* [**-l***lof*] [**-h***hif*] [**-g***gain*]

outfile is an analysis file ready for resynthesis

Modes

1 New formant envelope REPLACES the sound's own formant envelope
2 New formant envelope is IMPOSED ON TOP OF the sound's own formant envelope

Parameters

infile and *outfile* PVOC analysis files
fmntfile a formant data file made with FORMANTS GET
-l*lof lof* = low frequency, below which spectrum is set to zero
-h*hif hif* = high frequency, above which spectrum is set to zero
-g*gain gain* = adjustment to spectrum loudness (normally < 1.0)
-i quicksearch for formants (less accurate)

Understanding the FORMANTS PUT Process

Previously SPECFPUT, FORMANTS PUT reads a spectral trajectory as stored in *formant file* and imposes it on *infile*. In many sounds, the amplitude peaks are focused around certain frequencies; these are called formant areas, and are a key factor in creating the timbral colouration of a sound. These formants may themselves change over time, forming a spectral trajectory. The spectral trajectory is extracted (prior to running FORMANTS PUT) with **FORMANTS GET**. This trajectory (i.e., spectral envelope) is the rising and falling amplitude pattern of the partials (frequencies).

When this pattern is imposed on another file, the precise spectral envelope changes and colourations are introduced into the receiving file.

Musical Applications

The presence of the spectral trajectory data in the *fmntfile* makes it possible to impose the same data on more than one file.

There are several musical purposes which can be achieved with this process:

- simply as a means to link/unify musical data
- to re-colour a sound with the timbral characteristics of another
- to give one sound the meanings associated with another sound (this applies especially to recognisable sounds from the world around us)

End of FORMANTS PUT

FORMANTS SEE – convert formant data in a binary formant data file to a 'soundfile' for viewing

Usage

formants see *infile outsndfile* [**-v**]

Parameters

infile a binary formant data file created with **FORMANTS GET**
outsndfile a 'pseudo-soundfile' suitable for display in a graphic soundfile editor (do not resynthesize or play (an error message will be returned)
-v display data about formant-band parameters

Understanding the FORMANTS SEE Function

The resulting logarithmically scaled display indicates formant shapes, but *not* the absolute amplitude values. This process used to form a part of the CDP program SPECFSEE. In the display we see a series of spectral windows, separated by zeroes. The horizontal display represents the spectral frequencies and this may be displayed in a manner which increases regularly with frequency (if linear frequency-wise was selected when using FORMANTS GET) or in a manner which increases regularly with pitch (if linear pitch-wise was selected).

The vertical display represents the amplitude of each spectral band. The range of amplitudes is very large indeed, and in order to display them in the soundfile format, the log of the amplitude value is used and scaled to utilise fully the range of sound-sample values (-32768 to 32767).

End of FORMANTS SEE

FORMANTS VOCODE – impose spectral envelope of 2nd sound onto 1st sound

Usage

formants vocode *infile infile2 outfile* **-f***N* | **-p***N* [**-l***lof*] [**-h***hif*] [**-g***gain*]

Parameters

infile and *infile2* analysis files
outfile resultant analysis file
-f*N* extract formant envelope linear frequency-wise, using 1 point for every *N* equally spaced frequency channels
-p*N* extract formant envelope linear pitchwise, using *N* equally-spaced pitch-bands per octave
-l*lof lof* is low frquency, below which data is filtered out
-h*hif hif* is high frquency, above which data is filtered out
-g*gain gain* is the amplitude adjustment to the signal (normally < 1.0)

Understanding the FORMANTS VOCODE Process

FORMANTS VOCODE achieves 'cross-synthesis'; it was formerly called SPECVOCO, but here has been re-written extensively.

Cross-synthesis takes the time-evolving spectral envelope of one sound and imposes it on another sound. For example, vocal shouts will have a strong spectral profile. If this is imposed onto the sound of running water, we should hear the water shout.

Strictly speaking, this is not 'morphing', which is more of a gradual transition from one sound to another. But in a broader sense, it certainly is the reshaping of one sound with the characteristics of another.

Musical Applications

The difference between frequency-wise and pitch-wise could be important. As discussed in **FORMANTS GET** above, linear frequency-wise divides into absolutely equal frequency bands, while linear pitchwise divides octaves (of varying frequency widths) into equal-ratio bands, but these bands will vary in size from octave to octave, wider as one goes higher. It was recommended above to use frequency-wise if the sound is concentrated in the higher frequency areas (to get better resolution), and pitchwise for lower frequency areas. Experimentation is still the name of the game.

Frequencies below *lof* and above *hif* are simply ignored when the new spectrum is reconstructed.

Gain is not predictable because it depends on the level of the signals you put in, their relative levels, and the way those levels are distributed over the spectrum. A process that boosts level a great deal at the low end of the spectrum can cause distortion. Try it with no gain first.

End of FORMANTS VOCODE

Technical Discussion of formants and spectral envelopes

About Formants

Resonance is the transmission of vibrations from one vibrating body to another. A *resonator* is a body which reacts with sounds of any frequency or pitch, such as the sounding body of a piano or the belly and back of a violin, or the cavities associated with the vocal tract.

The effect of a resonator is to amplify certain frequency regions. These regions remain constant, unless the shape of the resonator is altered. Whatever partials are present in one of these regions will be amplified. The result is a perceptibly similar tone colour (timbre), even when the fundamental pitch of the vibrations is changing.

The human voice provides one of the best examples of this phenomenon. The vowels produced by the human voice maintain a constant 'oo' or 'ah' while the fundamental pitch changes, as long as the shape of the mouth etc. remains constant: the *same* frequency regions are amplified even though the pitch is going up or down. The specific formant (frequency regions) associated with each vowel have been identified – and can be used to impose a given vowel sound onto any other sound.

A **formant**, then, is an absolute frequency region in which the (changing) partials contained in any given sound are amplified, resulting in a constant timbre amidst changing pitches.

While this phenomenon occurs naturally through resonance effects, formants can also be produced on the computer by boosting the amplitudes of the partials in the specific frequency regions associated with a given vowel or tone.

The *spectral envelope* is the shape made by the amplitude levels of the partials contained in a sound. Note that this is *not* a fixed shape lasting throughout the duration of the sound. Rather, which partials are present at any given moment (i.e., the *spectrum*) and the amplitude level of these partials is changing all the time. Some sounds are very steady, with not so much change, while other sounds are constantly changing a great deal. The latter tend to be the more useful sounds for a variety of spectral manipulations: there is simply more to work with.

References

Also see the more detailed discussions and examples in:

- The discussion above on extracting formants.
- Dodge & Jerse, *Computer Music*, pp. 54-55 (Schirmer, NY & London, 1985).
- Pierce, John R., *The Science of Musical Sound*, p. 49 (Scientific American Books, NY & San Francisco, 1983)
- Ed. Boulanger, Richard, *The Csound Book*, pp. 653-656 for a chart of formant frequency regions for the various vowels for Soprano, Tenor, Alto, Bass and Countertenor (MIT Press, Cambridge Mass., 2000).

End of Technical Discussion

Last Updated 2 Oct 2016 -- HTML5 version
Documentation: Archer Endrich, revised R. Fraser