

## NAME

`sfam` – apply amplitude modulation to a soundfile

## SYNOPSIS

`sfam` [-t][-a][-r][-m MI][-A dB][-o output] soundfile

## DESCRIPTION

`sfam` applies amplitude modulation (or its variants, tremolo and ring modulation) to a soundfile.

*Amplitude modulation* is implemented as

$$(1 + m_i * \cos(2 * \pi * f / sr * t)) * \text{inputData}[t]$$

*Ring modulation* is implemented as

$$(m_i * \cos(2 * \pi * f / sr * t)) * \text{inputData}[t]$$

In both cases,  $m_i$  is restricted to values between 0 and 2.

*Tremolo* is implemented as

$$((\sin(2 * \pi * f * t / sr) + 1.0) / 2.0) * \text{inputData}[t]$$

This preserves the sign of the input signal.

## USAGE

`sfam -a -f 40 -m 1.5 -A 72 -o out.wav inputSound.aiff`

applies amplitude modulation with a frequency of 40 Hz., a modulation index of 1.5, and an output level of 72 dB to the input sound *inputSound.aiff* and places the output in the file *out.wav*

## OPTIONS

- t Apply *tremolo* to the input sound.
- a Apply *amplitude modulation* to the input sound.
- r Apply *ring modulation* to the input sound.
- f n Frequency  $n$  (in Hertz) for amplitude or ring modulation.
- m n The "modulation index".  $n$  must be between 0-2.
- A n Desired output amplitude  $n$  given in decibels.
- o file Place the output in *file*.

## BUGS

This program will only work with WAVE, AIFF, AIFC, and the old NeXT/Sun "snd" format soundfiles, that are 16-bit PCM encoded, with 1 or 2 channels.

## AUTHOR

Arun Chandra <arunc@evergreen.edu>

`sfam` is free software.

**NAME**

sfamplify – amplify or attenuate a soundfile.

**SYNOPSIS**

**sfamplify** [-d dB][-n][-w][-s][-o output] inputSoundfile

**DESCRIPTION**

**sfamplify** amplifies or attenuates a soundfile, or reports the range of its minimum and maximum amplitudes.

**USAGE**

sfamplify -d 80 -o out.aiff inputSound.wav

reads in *inputSound.wav* and either amplifies or attenuates the sound so that the final level is 80 dB, and places the output in *out.aiff*

**OPTIONS**

- d *n* Set output amplitude to *n* decibels. Maximum value is exactly  $20 * \log_{10}(2^{16})$ , which is about 96 dB. Minimum value is 0.
- n "Normalize" the soundfile; amplify all values to maximum.
- w "Wyatt" normalization. Sets maximum amp to -12 dB. (Named after Scott Wyatt, director of the Experimental Music Studios at the University of Illinois.
- s Scan soundfile, and report values.
- o output.wav  
Place the output in *output.wav* .

**BUGS**

This program will only work with WAVE, AIFF, AIFC, and the old NeXT/Sun "snd" format soundfiles, that are 16-bit PCM encoded, with 1 or 2 channels.

**AUTHOR**

Arun Chandra <arunc@evergreen.edu>

**sfamplify** is free software.

## NAME

`sfcats` – concatenates a sequence of soundfiles.

## SYNOPSIS

`sfcats` [-o *output*][-e *n*][-g *n*][-s] *snd1.wav snd2.wav ...*

## DESCRIPTION

`sfcats` concatenates a sequence of soundfiles. The soundfiles must have the same number of channels, the same sampling rate, and must all be 16-bit PCM samples.

## USAGE

`sfcats` -o *sequence.wav snd1.wav snd2.aiff snd3.wav ...*

Reads in *snd1.wav snd2.aiff snd3.wav* and places the output in *sequence.wav*

## OPTIONS

-o *output.wav*

Place the resulting sequence in the soundfile *output.wav*

-e *n* "Elide" the soundfiles together, where *n* is given in samples. This will cause the end of the 1st soundfile to be ramped down to 0 over *n* samples, the beginning of the 2nd soundfile ramped up from 0 over *n* samples. The last *n* samples of the 1st soundfile and the first *n* samples of the 2nd soundfile will be mixed together. This process will be done for all the soundfiles in the sequence, i.e., all will be "elided" together.

-g *n* Place a gap of *n* seconds between each soundfile.

-s Suppress the normal output messages. Useful for running from a scripting language, such as Python, etc.

## BUGS

This program will only work with WAVE, AIFF, AIFC, and the old NeXT/Sun "snd" format soundfiles, that are 16-bit PCM encoded, with 1 or 2 channels.

## AUTHOR

Arun Chandra <arunc@evergreen.edu>

`sfcats` is free software.

**NAME**

sfcheck – checks for inconsistencies between the size of a soundfile and the data written in its header.

**SYNOPSIS**

**sfcheck** [-f] soundfile(s)

**DESCRIPTION**

**sfcheck** checks for inconsistencies between the actual size of a soundfile (as reported by the OS), and the size that is written in the header.

**USAGE**

sfcheck snd1.wav snd2.aiff snd3.wav ...

Checks the soundfiles *snd1.wav snd2.aiff snd3.wav* and makes sure the size written to their headers is the correct one.

If a discrepancy is found, the user is asked whether the header should be changed to reflect the actual file-size.

**OPTIONS**

-f "Force" the correction of the header if a discrepancy is found, and don't ask the user for confirmation.

**BUGS**

This program will only work with WAVE, AIFF, AIFC, and the old NeXT/Sun "snd" format soundfiles, that are 16-bit PCM encoded, with 1 or 2 channels.

**AUTHOR**

Arun Chandra <arunc@evergreen.edu>

**sfcheck** is free software.

## NAME

`sfconvert` – applies various stereo/mono transformations.

## SYNOPSIS

`sfconvert` [options] soundfile(s)

## DESCRIPTION

`sfconvert` changes the channel format of the input soundfile(s) to a new one.

## OPTIONS

`sfconvert -j snd1.wav snd2.aiff` [or `sfconvert --join snd1.wav snd2.aiff`]

"Joins" the two mono soundfiles *snd1.wav* and *snd2.aiff* into one stereo file. The output will be placed in the file *snd1.J.wav* where the "J" stands for "joined."

`sfconvert -s snd3.wav` [or `sfconvert --split snd3.wav`]

"splits" the stereo file into two mono ones, and place the output in *snd3.L.wav* and *snd3.R.wav* , for "left" and "right" respectively.

`sfconvert -f snd4.aiff` [or `sfconvert --flip snd4.aiff`]

"flips" the channels of a stereo soundfile. The output is placed in *snd4.F.aiff*

`sfconvert --s2m snd5.wav` [or `sfconvert --stereo2mono snd5.wav`]

converts a stereo file to a mono file by summing the left and right channels together. The output is placed in *snd5.M.wav* .

`sfconvert --m2s snd6.aifc` [or `sfconvert --mono2stereo snd6.aifc`]

converts a mono file to a stereo file by placing the single stream of samples into both left and right channels. The output is placed in *snd6.S.aifc* .

`sfconvert --m2l snd7.wav` [or `sfconvert --mono2left snd7.wav`]

converts a mono file to a stereo file, and places the single input stream of samples into the left channel of a stereo file. The right channel is filled with zeros. The output is placed in *snd7.L.wav* .

`sfconvert --m2r snd8.aif` [or `sfconvert --mono2right snd8.aif`]

converts a mono file to a stereo file, and places the single input stream of samples into the right channel of a stereo file. The left channel is filled with zeros. The output is placed in *snd8.R.aif* .

## BUGS

This program will only work with WAVE, AIFF, AIFC, and the old NeXT/Sun "snd" format soundfiles, that are 16-bit PCM encoded, with 1 or 2 channels.

## AUTHOR

Arun Chandra <arunc@evergreen.edu>

`sfconvert` is free software.

**NAME**

`sfcopy` – copies one soundfile into another, changing soundfile formats.

**SYNOPSIS**

`sfcopy` *old.wav* *new.aiff*

**DESCRIPTION**

`sfcopy` copies one soundfile into another, changing formats as understood by the file extension.

**USAGE**

`sfcopy` *old.wav* *new.aiff*

copies *old.wav* and places the output in *new.aiff*. The WAVE format (which is little-endian) is converted into an AIFF format (which is big-endian).

**BUGS**

This program will only work with WAVE, AIFF, AIFC, and the old NeXT/Sun "snd" format soundfiles, that are 16-bit PCM encoded, with 1 or 2 channels.

**AUTHOR**

Arun Chandra <arunc@evergreen.edu>

`sfcopy` is free software.

## NAME

`sffilters` – applies a number of filters to a soundfile.

## SYNOPSIS

`sffilters` [-t n][-f n][-w n][-N n][-o output][-b][-info] inputSoundfile

## DESCRIPTION

`sffilters` applies a variety of filters to a soundfile.

## OPTIONS

- t 0 Butterworth low-pass, specify cutoff frequency.  
1 Butterworth high-pass, specify cutoff frequency.  
2 Butterworth band-pass, specify center frequency and bandwidth.  
3 Butterworth band-reject, specify center frequency and bandwidth.  
4 2nd order recursive filter, specify center and bandwidth.  
5 2nd order FIR filter, specify center and bandwidth.  
6 N-order FIR filter (lowpass), specify number of taps (N) and cutoff frequency.  
7 two-pole IIR filter, specify center and bandwidth.
- f c Cutoff frequency or center frequency, depending on type of filter chosen.
- w b Bandwidth for bandpass and band-reject filters.
- N n Number of taps for the N-order FIR filter.
- o outputSoundfile  
Write the output to the *outputSoundfile* .
- b Boost sound to maximum after filtering. *-info* Print the filter equations to the screen.

## EQUATIONS

Filters 0-3

Butterworth filters:  $y[n] = a_0 * x[n] + a_1 * x[n-1] + a_2 * x[n-2] - b_1 * y[n-1] - b_2 * y[n-2]$

Filter 4 2nd order Infinite Impulse Response (IIR) filter:  $y[n] = a_0 * x[n] - b_1 * y[n-1] - b_2 * y[n-2]$

Filter 5 2nd order Finite Impulse Response (FIR) filter:  $y[n] = a_0 * x[n] + a_1 * x[n-1] + a_2 * x[n-2]$

Filter 6 N-order FIR filter (lowpass):  $y[n] = a_0 * x[n] + a_1 * x[n-1] + a_2 * x[n-2] + \dots + a_N * x[n-N]$

Filter 7 Two-pole IIR filter:  $y[n] = G * (x[n] - R * x[n-2]) + b_1 * y[n-1] + b_2 * y[n-2]$

## USAGE

`sffilters -t 0 -f 1000 -o output.wav input.aiff`

Apply a Butterworth lowpass filter (-t 0) with a cutoff frequency of 1000 Hertz (-f 1000) to the file *input.aiff* and place the output in the file *output.wav* (-o output.wav).

`sffilters -t 6 -N 15 -f 800 -o output.aiff input.wav`

Apply an N-order FIR filter (-t 6) with 15 taps (-N 15) and a cutoff frequency of 800 Hertz (-f 800) to the input file *input.wav* and place the output in the file *output.aiff* .

## BUGS

This program will only work with WAVE, AIFF, AIFC, and the old NeXT/Sun "snd" format soundfiles, that are 16-bit PCM encoded, with 1 or 2 channels.

## AUTHOR

Arun Chandra <arunc@evergreen.edu>

`sffilters` is free software.

**NAME**

sfflip – reverses the channels in a stereo soundfile.

**SYNOPSIS**

**sfflip** stereoSoundfile [outputSoundfile]

**DESCRIPTION**

**sfflip** reverses the channels of a stereo soundfile. If the original is A (left) and B (right), the output will be B (left) and A (right).

**USAGE**

sfflip snd1.wav

Flips the channels, and puts the output into *flip.wav*, which is the default output soundfile.

sfflip snd1.wav out.aiff

Flips the channels, and puts the output into *out.aiff*.

**BUGS**

This program will only work with WAVE, AIFF, AIFC, and the old NeXT/Sun "snd" format soundfiles, that are 16-bit PCM encoded, with 1 or 2 channels.

**AUTHOR**

Arun Chandra <arunc@evergreen.edu>

**sfflip** is free software.

**NAME**

`sfinfo` – shows the information in a soundfile header.

**SYNOPSIS**

`sfinfo` [-s][-1][-v] soundfile(s)

**DESCRIPTION**

`sfinfo` displays information about the soundfile, as read in from its header.

**OPTIONS**

- s Sum the durations of a number of soundfiles, and print the result.
- 1 Display one-line summation of the header information.
- v Verbosely show the information in the soundfile file, showing chunk names, sizes, etc.

**USAGE**

`sfinfo snd1.wav`

Display the information in the soundfile header.

`sfinfo -s *.aiff`

Sum the durations of all the ".aiff" files in the directory.

`sfinfo -1 *.wav`

Give a 1-line synopsis of the information in all the "\*.wav" files in the directory.

**BUGS**

This program will only work with WAVE, AIFF, AIFC, and the old NeXT/Sun "snd" format soundfiles, that are 16-bit PCM encoded, with 1 or 2 channels.

**AUTHOR**

Arun Chandra <arunc@evergreen.edu>

`sfinfo` is free software.

**NAME**

sfkaiser – Apply a "Kaiser" filter to a soundfile.

**SYNOPSIS**

**sfkaiser inputSound.wav outputSound.wav**

**DESCRIPTION**

**sfkaiser** reads in *inputSound.wav* then writes out a filtered version to *outputSound.wav* .

**OPTIONS**

There are no options to the **sfkaiser** program, since it prompts the user for all the information it needs.

**USAGE**

sfkaiser snd1.wav snd2.aiff

Apply the filter to *snd1.wav* and put the results in *snd2.aiff* .

The program will prompt the user for

the type of filter desired (lp [lowpass], hp [highpass], bp [bandpass], bs [bandstop]);

the desired stopband attenuation in dB;

the passband edge frequency in Hertz;

the stopband edge frequency in Hertz.

The program will then calculate the coefficients according to the Kaiser filter formulas, filter the file, and save the result in *snd1.aiff* .

**BUGS**

This program will only work with WAVE, AIFF, AIFC, and the old NeXT/Sun "snd" format soundfiles, that are 16-bit PCM encoded, with 1 or 2 channels.

**AUTHOR**

Arun Chandra <arunc@evergreen.edu>

**sfkaiser** is free software.

## NAME

`sfmix` – mixes "N" number of one-channel soundfiles together.

## SYNOPSIS

`sfmix` [-a amp][-b][-s][-o outputSoundfile] inputSoundfile(s)

## DESCRIPTION

`sfmix` mixes the soundfiles given on the command-line, and places the output in a stereo soundfile.

The input soundfiles (which must be single-channel) are spread equally across the stereo field in the following manner: the more soundfiles, the "closer together they are" in the stereo field. "Equal power" panning is used to avoid "holes" in the center.

## USAGE

`sfmix mono1.wav mono2.wav`

The soundfile *mono1.wav* will be in the left channel only, and *mono2.wav* will be in the right channel only. The output will be put in the default output soundfile *sfmixOut.snd*.

`sfmix -a 80 mono1.wav mono2.wav mono3.wav`

The soundfile *mono1.wav* will be in the left channel only, *mono2.wav* will be in the center, and *mono3.wav* will be in the right channel only. The output will be boosted or attenuated to be no more than 80 dB.

`sfmix -b mono1.wav mono2.wav mono3.wav mono4.wav`

The soundfile *mono1.wav* will be in the left channel only, *mono2.wav* will be 1/3 of the way to the right channel, *mono3.wav* will be 2/3 of the way to the right channel, and *mono4.wav* will be in the right channel only. The output soundfile will be boosted to the maximum 16-bits will allow (about 96 dB).

`sfmix -o out.wav mono1.wav mono2.wav mono3.wav mono4.wav mono5.wav`

The output will be put in *out.wav* and the rest of the soundfiles will be equally panned across the stereo field, from left to right.

## OPTIONS

-a N Set the maximum level of the output soundfile to *N* decibels.

-b Boost the output soundfile to the maximum amplitude, about 96 dB.

-s Suppress normal screen output. Useful for calling from scripting programs, such as Python.

-o outputSoundfile

Place the output in the *outputSoundfile*.

## BUGS

This program will only work with WAVE, AIFF, AIFC, and the old NeXT/Sun "snd" format soundfiles, that are 16-bit PCM encoded, with 1 or 2 channels.

## AUTHOR

Arun Chandra <arunc@evergreen.edu>

`sfmix` is free software.

**NAME**

sfnoise – A "white" noise generator.

**SYNOPSIS**

**sfnoise** [-a dB][-t type][-d seconds][-r samplingRate][-o outputSoundfile]

**DESCRIPTION**

**sfnoise** creates "white" noise, either "uniform" (the default) or "gaussian", at a variety of durations, amplitudes, and sampling rates.

**USAGE**

sfnoise -d 10

Generate 10 seconds of "uniform" white noise (the default), and put the output into *noise.wav*, which is also the default.

**OPTIONS**

-a dB Set the output amplitude level to *dB* decibels.

-t type *type* can either be *gaussian* or *uniform*.

-o file Place the output in *file*.

-d n Create the noise for *n* seconds. This variable must be given by the user.

-r srate Set the sampling rate to be *srate*. The default is 44100 samples per second.

**BUGS**

This program will only work with WAVE, AIFF, AIFC, and the old NeXT/Sun "snd" format soundfiles, that are 16-bit PCM encoded, with 1 or 2 channels.

**AUTHOR**

Arun Chandra <arunc@evergreen.edu>

**sfnoise** is free software.

**NAME**

sfpitches – Creates and plays a pitch or frequency

**SYNOPSIS**

**sfpitches** [-s samples][-r srate][-c] frequency|pitchName

**DESCRIPTION**

**sfpitches** plays a tone at the specified pitch. The pitch can be given in "pitch-class" notation (c#3, ab1, b7), or as a frequency in Hertz.

**USAGE**

sfpitches c#4

Play the pitch c#4 (which is 554 Hertz).

sfpitches 1640

Play the frequency 1640 (which is between g5 and g#5).

sfpitches -s 100

Displays a small chart showing the frequency created by a period of 100 samples (at a sampling rate of 44100 sps), and the pitch name(s) it is close to. (It's between a4 and a#4, closer to a4.)

sfpitches -c

Display a chart of pitch/frequency/sample equivalents at the default sampling rate of 44100.

sfpitches -c -r 48000

Display a chart of pitch/frequency/sample equivalents at the sampling rate of 48000.

**AUTHOR**

Arun Chandra <arunc@evergreen.edu>

**sfpitches** is free software.

**NAME**

`sfplay` – Plays a mono or stereo 16-bit PCM soundfile

**SYNOPSIS**

`sfplay` [-v][-c] soundfile(s)

**DESCRIPTION**

`sfplay` plays a soundfile or a sequence of soundfiles.

**USAGE**

`sfplay` soundfile

Plays a single soundfile.

`sfplay -c snd1.wav snd2.aiff`

Plays the two soundfiles *snd1.wav* and *snd2.aiff* without any break between them.

**OPTIONS**

`-c` Continuous. Play a sequence of soundfiles with no break between them.

**BUGS**

This program will only work with WAVE, AIFF, AIFC, and the old NeXT/Sun "snd" format soundfiles, that are 16-bit PCM encoded, with 1 or 2 channels.

**AUTHOR**

Arun Chandra <arunc@evergreen.edu> This program uses PortAudio 18.1, a freely available set of programs whose main authors are Ross Bencina and Phil Burk, to handle the low-level work of communicating with the hardware. I wrote the code for opening and closing the soundfiles, and for feeding the data to the PortAudio routines.

`sfplay` is free software, as is PortAudio.

## NAME

`sfpv` – Applies a phase vocoder to a soundfile.

## SYNOPSIS

`sfpv [-d n][-c n][-C n][-L n][-t n][-o outputSoundfile][-S s] inputSoundfile`

## DESCRIPTION

`sfpv` Applies a phase vocoder to a soundfile. This program is primarily used for changing the pitch of a sound without changing its duration, or for changing the duration of a sound without changing its pitch.

## OPTIONS

- `-d n` Desired duration of output sound in seconds.
- `-c n` Change of pitch in cents (where 100 cents is equal to a semi-tone).
- `-C n` Change of ending pitch in cents. The "-c" option sets the initial change of pitch in cents, and the "-C" option sets the final change of pitch in cents.
- `-L n` Length of the FFT window, in samples. The default value is 4096. (This probably never has to be changed.)
- `-t n` Amplitude threshold. All amplitudes that are below this threshold are ignored, speeding up the phase vocoder significantly. (This probably never has to be changed.)
- `-o outputSoundfile`  
Where to put the resulting sound.
- `-S dur` Add "dur" seconds of silence to the end of the sound.

## USAGE

`sfpv -d 10 soundfile`

Stretch (or shrink) the duration of *soundfile* to be 10 seconds long without changing its pitch. The output is placed in the default output soundfile *sfpvout.wav*.

`sfpv -c +100 snd1.wav`

Raise the pitch of *snd1.wav* by 100 cents. Place the output in *sfpvout.wav*.

`sfpv -c -100 -C +100 snd1.wav`

At the beginning of the sound, shift the pitch by -100 cents, and keep raising the pitch, until by the end of the sound it is +100 above the original. Put the output in *sfpvout.wav*.

## BUGS

This program will only work with WAVE, AIFF, AIFC, and the old NeXT/Sun "snd" format soundfiles, that are 16-bit PCM encoded, with only 1 channel.

## AUTHOR

Arun Chandra <arunc@evergreen.edu>

This program is heavily indebted to F. Richard Moore, who published the code for a complete phase vocoder in his book "Elements of Computer Music."

`sfpv` is free software.

**NAME**

`sframp` – Ramps the beginning and/or ending of a soundfile.

**SYNOPSIS**

`sframp` [-u *n*][-d *n*][-o *outputSoundfile*] *inputSoundfile*

**DESCRIPTION**

`sframp` creates an S-shaped ramp at the beginning and/or ending of a soundfile, where the amplitudes are ramped from 0-1 at the beginning and 1-0 at the end.

**OPTIONS**

-u *n*     Upramp (initial) length. Default value for *n* is 500 samples.

-d *n*     Downramp (final) length. Default value for *n* is 500 samples.

-o *outputSoundfile*

Where to write the newly-ramped soundfile. The default output soundfile is *sframpOut.wav*.

**USAGE**

`sframp -d 10 soundfile`

Apply a "down" ramp to the soundfile at the end, and write the output to the default soundfile *sframpOut.wav*.

`sframp -o out.wav inputSoundfile.aiff`

Apply the default ramps (500 samples up and 500 samples down) to *inputSoundfile.aiff* and place the output in *out.wav*.

**BUGS**

This program will only work with WAVE, AIFF, AIFC, and the old NeXT/Sun "snd" format soundfiles, that are 16-bit PCM encoded, with 1 or 2 channels.

**AUTHOR**

Arun Chandra <arunc@evergreen.edu>

`sframp` is free software.

**NAME**

sfrose – Creates a "classic" tone matrix.

**SYNOPSIS**

**sfrose**

**DESCRIPTION**

**sfrose** Creates a "classic" tone matrix of the kind first invented in the time of Schoenberg, Berg, and Webern. The transpositions of the row are given horizontally, and the inversions of the row are given vertically.

It can take a maximum length of 25 tones as input.

**USAGE**

sfrose c# g f# eb

This gives the result of

c# g f# eb

g c# c a

g# d c# bb

b f e c#

**AUTHOR**

Arun Chandra <arunc@evergreen.edu>

**sfrose** is free software.

**NAME**

sfstereomix – Mixes stereo soundfiles together

**SYNOPSIS**

**sfstereomix** will mix the stereo soundfiles given on the command line.

**DESCRIPTION**

**sfstereomix** allows you to mix an arbitrary number of stereo soundfiles from the command line. The soundfiles must all have the same sampling rate, and must all have 16-bit samples.

The duration of the resulting sound will be the maximum duration of the input soundfiles.

The amplitude of the resulting sound will be the maximum amplitude in all the soundfiles.

**USAGE**

```
sfstereomix snd1.aif snd2.wav snd3.snd
```

This mixes together the three soundfiles, and places the output in the default soundfile *stereomix.wav*.

```
sfstereomix -o out.aiff snd1.aif snd2.wav snd3.snd
```

This mixes together the three soundfiles, and places the output in *out.aiff*.

**AUTHOR**

Arun Chandra <arunc@evergreen.edu>

**sfstereomix** is free software.

**NAME**

sfsweep – Creates a sine-tone sweep.

**SYNOPSIS**

**sfsweep** creates a sine-tone sweep from a specified initial frequency and amplitude, to a final specified frequency and amplitude. The duration of the sweep can also be altered.

**DESCRIPTION**

**sfsweep** allows you to create a sine-tone sweep, and prompts the user for the initial and final frequencies, amplitudes, and sweep duration. The resulting glissando is then written to a file *sweep.wav*.

The default values 440 Hz. and 880 Hz. for the frequencies, 84 dB for the amplitude, and 1 second for the duration.

**USAGE**

sfsweep

**AUTHOR**

Arun Chandra <arunc@evergreen.edu>

**sfsweep** is free software.

## NAME

`sftesttone` – Creates a test tone with user settable properties.

## SYNOPSIS

`sftesttone` creates a sine tone of a specified frequency, amplitude, duration, and channel placement.

## DESCRIPTION

`sftesttone` allows you to create a sine-tone sweep, and prompts the user for the initial and final frequencies, amplitudes, and sweep duration. The resulting glissando is then written to a file *sweep.wav*.

The default values 440 Hz. and 880 Hz. for the frequencies, 84 dB for the amplitude, and 1 second for the duration.

Remember to use the "-c" option when running the program, otherwise no soundfile will be created.

## OPTIONS

- a db Amplitude in decibels. Default: 84 dB.
- f hz Frequency in Hertz. Default: 1000 Hertz.
- d sec Duration in seconds. Default: 30 seconds.
- o outputsoundfile  
Name of the output soundfile. Default: test.wav.
- l Left channel only.
- r Right channel only.
- c Flag to "create" the soundfile.

## USAGE

`sftesttone -c`

Creates a sine tone with the default values of 1000 Hz., 84 dB, 30 seconds duration, and put it in the default soundfile *test.wav*.

`sftesttone -a 96 -f 440 -d 10 -l -o out.wav -c`

Create a sine tone with an amplitude of 96 dB, frequency of 440 Hertz, a duration of 10 seconds, left-channel only, and put it in the soundfile *out.wav*.

## AUTHOR

Arun Chandra <arunc@evergreen.edu>

`sftesttone` is free software.

**NAME**

`sftonerow` – Creates a sequence of tones.

**SYNOPSIS**

`sftonerow` creates a sequence of enveloped sine tones from a list of pitch names or frequencies.

**DESCRIPTION**

`sftonerow` is useful for hearing a sequence of tones, particularly if one is interested in those acoustic properties which do not reinforce tonal relationships.

**USAGE**

`sftonerow c#4 f5 eb3 d6`

will create a sequence of sine tones at the specified pitches, each with the default duration of 1 second.

`sftonerow -d 0.5 -o out.wav 440 880 660 1000`

This will create a sequence of the frequencies "440 880 660 1000", give each tone a duration of 0.5 seconds, and place the output in the file *out.wav* .

**AUTHOR**

Arun Chandra <arunc@evergreen.edu>

`sftonerow` is free software.

**NAME**

`sfwaves` – Sums a sequence of frequencies and plays them.

**SYNOPSIS**

`sfwaves` [-d dur][-a amp][-t type][-o outputSoundfile] f1 f2 f3 ...  
Sum together the frequencies  $f1 + f2 + f3$  and play them.

**DESCRIPTION**

`sfwaves` is useful for exploring what the combination of frequencies can do. Frequencies are given in Hertz, and the generating waveform can be set on the command line, along with duration and amplitude.

**USAGE**

`sfwaves 440 660 770`

Sum together the frequencies 440+660+770, using default amplitude and duration, and place the output in the file *sfwaveout.wav* which is the default output soundfile.

`sfwaves -d 10 -a 20000 -t tri 440 660 770 880 990`

Sum together the frequencies on the command line, use a triangle wave to create them, have the total duration be 10 seconds, and the linear amplitude be 20000. The output is again placed in *sfwaveout.wav*.

**BUGS**

`sfwaves` will only write WAVE, AIFF, or SND files, based on the file extension.

**AUTHOR**

Arun Chandra <arunc@evergreen.edu>

`sfwaves` is free software.

**NAME**

sndfft – Applies a Fast Fourier Transform to a soundfile

**SYNOPSIS**

**sndfft** soundfile

**DESCRIPTION**

**sndfft** applies an FFT to the soundfile, and puts its output into a textfile with x/y pairs in columns, "x" representing the frequency and "y" representing the amplitude in decibels.

The output is written to a file named "fft.data". This is useful for reading into programs such as "gnuplot", or "octave" for graphic analysis.

**USAGE**

sndfft [-l fftlen][-w winType][-x xmax] soundfile

**OPTIONS**

- l Length of fft (must be power of 2). Default: 1024
- w Window type: 0=noWindow, 1=Hamming, 2=Hanning, 3=Hartlett,4=Blackman, 5=Blackman-Harris. Default: Hamming
- x Highest frequency (xvalue) displayed. Default: Nyquist frequency (SamplingRate/2)

**BUGS**

This program will only work with WAVE, AIFF, AIFC, and the old NeXT/Sun "snd" format soundfiles, that are 16-bit PCM encoded, with 1 or 2 channels.

**AUTHOR**

Arun Chandra <arunc@evergreen.edu>

**sndfft** is free software.

## NAME

sndrms – Reports the Root Mean Square (RMS) level of a soundfile.

## SYNOPSIS

**sndrms** *snd1.wav snd2.wav ...*

## DESCRIPTION

**sndrms** concatenates a sequence of soundfiles. The soundfiles must have the same number of channels, the same sampling rate, and must all be 16-bit PCM samples.

## USAGE

sndrms -o *sequence.wav* *snd1.wav snd2.aiff snd3.wav ...*

Reads in *snd1.wav snd2.aiff snd3.wav* and places the output in *sequence.wav*

## OPTIONS

-o *output.wav*

Place the resulting sequence in the soundfile *output.wav*

-e *n* "Elide" the soundfiles together, where *n* is given in samples. This will cause the end of the 1st soundfile to be ramped down to 0 over *n* samples, the beginning of the 2nd soundfile ramped up from 0 over *n* samples. The last *n* samples of the 1st soundfile and the first *n* samples of the 2nd soundfile will be mixed together. This process will be done for all the soundfiles in the sequence, i.e., all will be "elided" together.

-g *n* Place a gap of *n* seconds between each soundfile.

-s Suppress the normal output messages. Useful for running from a scripting language, such as Python, etc.

## BUGS

This program will only work with WAVE, AIFF, AIFC, and the old NeXT/Sun "snd" format soundfiles, that are 16-bit PCM encoded, with 1 or 2 channels.

## AUTHOR

Arun Chandra <arunc@evergreen.edu>

**sndrms** is free software.