NAME

lame - create mp3 audio files

SYNOPSIS

lame [options] <infile> <outfile>

DESCRIPTION

LAME is a program which can be used to create compressed audio files. (Lame ain’t an MP3 encoder). These audio files can be played back by popular MP3 players such as mpg123 or madplay. To read from stdin, use "-" for <infile>. To write to stdout, use "-" for <outfile>.

OPTIONS

Input options:

-r Assume the input file is raw pcm. Sampling rate and mono/stereo/jstereo must be specified on the command line. For each stereo sample, LAME expects the input data to be ordered left channel first, then right channel. By default, LAME expects them to be signed integers with a bitwidth of 16 and stored in little-endian. Without **-r,** LAME will perform several *fseek()’s* on the input file looking for WAV and AIFF headers.
Might not be available on your release.

-x Swap bytes in the input file (or output file when using **--decode).**
For sorting out little endian/big endian type problems. If your encodings sounds like static, try this first.
Without using **-x,** LAME will treat input file as native endian.

-s *sfreq*

sfreq = 8/11.025/12/16/22.05/24/32/44.1/48

Required only for raw PCM input files. Otherwise it will be determined from the header of the input file.

LAME will automatically resample the input file to one of the supported MP3 samplerates if necessary.

--bitwidth *n*

Input bit width per sample.
n = 8, 16, 24, 32 (default 16)

Required only for raw PCM input files. Otherwise it will be determined from the header of the input file.

--signed
Instructs LAME that the samples from the input are signed (the default for 16, 24 and 32 bits raw pcm data).

Required only for raw PCM input files.

--unsigned
Instructs LAME that the samples from the input are unsigned (the default for 8 bits raw pcm data, where 0x80 is zero).

Required only for raw PCM input files and only available at bitwidth 8.

--little-endian
Instructs LAME that the samples from the input are in little-endian form.

Required only for raw PCM input files.

--big-endian
Instructs LAME that the samples from the input are in big-endian form.

Required only for raw PCM input files.

--mp1input
Assume the input file is a MPEG Layer I (ie MP1) file.
If the filename ends in ".mp1" LAME will assume it is a MPEG Layer I file. For stdin or Layer I files which do not end in .mp1 you need to use this switch.

--mp2input
Assume the input file is a MPEG Layer II (ie MP2) file.
If the filename ends in ".mp2" LAME will assume it is a MPEG Layer II file. For stdin or Layer II files which do not end in .mp2 you need to use this switch.

--mp3input
Assume the input file is a MP3 file.
Useful for downsampling from one mp3 to another. As an example, it can be useful for streaming through an IceCast server.
If the filename ends in ".mp3" LAME will assume it is an MP3. For stdin or MP3 files which do not end in .mp3 you need to use this switch.

--nogap *file1 file2* ...
gapless encoding for a set of contiguous files

--nogapout *dir*
output dir for gapless encoding (must precede --nogap)

--out-dir *dir*
If no explicit output file is specified, a file will be written at given path. Ignored when using piped/streamed input

Operational options:

-m *mode*
mode = s, j, f, d, m, l, r

Joint-stereo is the default mode for stereo files.

(s)imple stereo (Forced LR)
In this mode, the encoder makes no use of potentially existing correlations between the two input channels. It can, however, negotiate the bit demand between both channel, i.e. give one channel more bits if the other contains silence or needs less bits because of a lower complexity.

(j)oint stereo
In this mode, the encoder can use (on a frame by frame basis) either L/R stereo or mid/side stereo. In mid/side stereo, the mid (L+R) and side (L-R) channels are encoded, and more bits are allocated to the mid channel than the side channel. When there isn't too much stereo separation, this effectively increases the bandwidth, so having higher quality with the same amount of bits.

Using mid/side stereo inappropriately can result in audible compression artifacts. Too much switching between mid/side and regular stereo can also sound bad. To determine when to switch to mid/side stereo, LAME uses a much more sophisticated algorithm than the one described in the ISO documentation.

(f)orced MS stereo
Forces all frames to be encoded with mid/side stereo. It should be used only if you are sure that every frame of the input file has very little stereo separation.

(d)ual channel
In this mode, the 2 channels will be totally independently encoded. Each channel will have exactly

half of the bitrate. This mode is designed for applications like dual languages encoding (for example: English in one channel and French in the other). Using this encoding mode for regular stereo files will result in a lower quality encoding.

(m)ono
The input will be encoded as a mono signal. If it was a stereo signal, it will be downsampled to mono. The downmix is calculated as the sum of the left and right channel, attenuated by 6 dB. Also note that, if using a stereo RAW PCM stream, you need to use the -a parameter.

(l)eft channel only
The input will be encoded as a mono signal. If it was a stereo signal, the left channel will be encoded only.

(r)ight channel only
The input will be encoded as a mono signal. If it was a stereo signal, the right channel will be encoded only.

-a Mix the stereo input file to mono and encode as mono.
The downmix is calculated as the sum of the left and right channel, attenuated by 6 dB.

This option is only needed in the case of raw PCM stereo input (because LAME cannot determine the number of channels in the input file). To encode a stereo RAW PCM input file as mono, use **lame -a -m m**

For WAV and AIFF input files, using **-m m** will always produce a mono .mp3 file from both mono and stereo input.

--freeformat
Produces a free format bitstream. With this option, you can use **-b** with any bitrate higher than 8 kbps.

However, even if an mp3 decoder is required to support free bitrates at least up to 320 kbps, many players are unable to deal with it.

Tests have shown that the following decoders support free format:
in_mpg123 up to 560 kbps
l3dec up to 310 kbps
LAME up to 640 kbps
MAD up to 640 kbps

--decode

Uses LAME for decoding to a wav file. The input file can be any input type supported by encoding, including layer II files. LAME uses a fork of mpglib known as HIP for decoding.

If **-t** is used (disable wav header), LAME will output raw pcm in native endian format. You can use **-x** to swap bytes order.

This option is not usable if the MP3 decoder was **explicitly** disabled in the build of LAME.

-t Disable writing of the INFO Tag on encoding.

This tag is embedded in frame 0 of the MP3 file. It includes some information about the encoding options of the file, and in VBR it lets VBR aware players correctly seek and compute playing times of VBR files.

When **--decode** is specified (decode to WAV), this flag will disable writing of the WAV header. The output will be raw pcm, native endian format. Use **-x** to swap bytes.

--comp *arg*

Instead of choosing bitrate, using this option, user can choose compression ratio to achieve.

--scale *n*
--scale-l *n*
--scale-r *n*

Scales input (every channel, only left channel or only right channel) by *n*. This just multiplies the PCM data (after it has been converted to floating point) by *n*.

n > 1: increase volume
n = 1: no effect
n < 1: reduce volume

Use with care, since most MP3 decoders will truncate data which decodes to values greater than 32768.

--replaygain-fast

Compute ReplayGain fast but slightly inaccurately.

This computes "Radio" ReplayGain on the input data stream after user-specified volume-scaling and/or resampling.

The ReplayGain analysis does *not* affect the content of a compressed data stream itself, it is a value stored in the header of a sound file. Information on the purpose of ReplayGain and the algorithms used is available from **http://www.replaygain.org/.**

Only the "RadioGain" Replaygain value is computed, it is stored in the LAME tag. The analysis is performed with the reference volume equal to 89dB. Note: the reference volume has been changed from 83dB on transition from version 3.95 to 3.95.1.

This switch is enabled by default.

See also: **--replaygain-accurate, --noreplaygain**

--replaygain-accurate

Compute ReplayGain more accurately and find the peak sample.

This computes "Radio" ReplayGain on the decoded data stream, finds the peak sample by decoding on the fly the encoded data stream and stores it in the file.

The ReplayGain analysis does *not* affect the content of a compressed data stream itself, it is a value stored in the header of a sound file. Information on the purpose of ReplayGain and the algorithms used is available from **http://www.replaygain.org/.**

By default, LAME performs ReplayGain analysis on the input data (after the user-specified volume scaling). This behavior might give slightly inaccurate results because the data on the output of a lossy compression/decompression sequence differs from the initial input data. When **--replaygain-accurate** is specified the mp3 stream gets decoded on the fly and the analysis is performed on the decoded data stream. Although theoretically this method gives more accurate results, it has several disadvantages:

* tests have shown that the difference between the ReplayGain values computed on the input data and decoded data is usually not greater than 0.5dB, although the minimum volume difference the human ear can perceive is about 1.0dB

* decoding on the fly significantly slows down the encoding process

The apparent advantage is that:

* with **--replaygain-accurate** the real peak sample is determined and stored in the file. The knowledge of the peak sample can be useful to decoders (players) to prevent a negative effect called ’clipping’ that introduces distortion into the sound.

Only the "RadioGain" ReplayGain value is computed, it is stored in the LAME tag. The analysis is performed with the reference volume equal to 89dB. Note: the reference volume has been changed from 83dB on transition from version 3.95 to 3.95.1.

This option is not usable if the MP3 decoder was **explicitly** disabled in the build of LAME. (Note: if LAME is compiled without the MP3 decoder, ReplayGain analysis is performed on the input data after user-specified volume scaling).

See also: **--replaygain-fast, --noreplaygain --clipdetect**

--noreplaygain

Disable ReplayGain analysis.

By default ReplayGain analysis is enabled. This switch disables it.

See also: **--replaygain-fast, --replaygain-accurate**

--clipdetect

Clipping detection.

Enable **--replaygain-accurate** and print a message whether clipping occurs and how far in dB the waveform is from full scale.

This option is not usable if the MP3 decoder was **explicitly** disabled in the build of LAME.

See also: **--replaygain-accurate**

--preset type | [cbr] kbps

Use one of the built-in presets.

Have a look at the PRESETS section below.

--preset help gives more infos about the the used options in these presets.

--noasm type

Disable specific assembly optimizations (**mmx / 3dnow / sse**). Quality will not increase, only speed will be reduced. If you have problems running Lame on a Cyrix/Via processor, disabling mmx optimizations might solve your problem.

Verbosity:

--disptime *n*

Set the delay in seconds between two display updates.

--nohist

By default, LAME will display a bitrate histogram while producing VBR mp3 files. This will disable that feature.
Histogram display might not be available on your release.

-S
--silent
--quiet

Do not print anything on the screen.

--verbose

Print a lot of information on the screen.

--help

Display a list of available options.

Noise shaping & psycho acoustic algorithms:

-q *qual*

0 <= *qual* <= 9

Bitrate is of course the main influence on quality. The higher the bitrate, the higher the quality. But for a given bitrate, we have a choice of algorithms to determine the best scalefactors and Huffman encoding (noise shaping).

For CBR and ABR, the following table applies:

-q 0:
Use the best algorithms (Best Huffman coding search, full outer loop, and the highest precision of several parameters).

-q 1 to q 4:
Similar to -q 0 without the full outer loop and decreasing precision of parameters the further from q0. -q 3 is the default.

-q 5 and -q 6:
Same as -q 7, but enables noise shaping and increases subblock gain

-q 7 to -q 9:
Same as -f. Very fast, OK quality. Psychoacoustics are used for pre-echo and mid/side stereo, but no noise-shaping is done.

For the default VBR mode since LAME 3.98, the following table applies :

-q 0 to -q 4:
include all features of the other modes and additionally use the best search when applying Huffman coding.

-q 5 and -q 6:
include all features of -q7, calculate and consider actual quantisation noise, and additionally enable subblock gain.

-q 7 to -q 9
This level uses a psymodel but does not calculate quantisation noise when encoding: it takes a quick guess.

-h Alias of **-q 2**

-f Alias of **-q 7**

CBR (constant bitrate, the default) options:

-b *n*
For MPEG-1 (sampling frequencies of 32, 44.1 and 48 kHz)
n = 32, 40, 48, 56, 64, 80, 96, 112, 128, 160, 192, 224, 256, 320

For MPEG-2 (sampling frequencies of 16, 22.05 and 24 kHz)
n = 8, 16, 24, 32, 40, 48, 56, 64, 80, 96, 112, 128, 144, 160

For MPEG-2.5 (sampling frequencies of 8, 11.025 and 12 kHz)
n = 8, 16, 24, 32, 40, 48, 56, 64

Default is 128 for MPEG1 and 64 for MPEG2 and 32 for MPEG2.5
(64, 32 and 16 respectively in case of mono).

--cbr

enforce use of constant bitrate. Used to disable VBR or ABR encoding even if their settings are enabled.

ABR (average bitrate) options:

--abr *n*

Turns on encoding with a targeted average bitrate of n kbits, allowing to use frames of different sizes. The allowed range of *n* is 8 **-** 310, you can use any integer value within that range.

It can be combined with the **-b** and **-B** switches like: **lame --abr** *123* **-b** *64* **-B** *192 a.wav a.mp3* which would limit the allowed frame sizes between 64 and 192 kbits.

The use of **-B** is NOT RECOMMENDED. A 128 kbps CBR bitstream, because of the bit reservoir, can actually have frames which use as many bits as a 320 kbps frame. VBR modes minimize the use of the bit reservoir, and thus need to allow 320 kbps frames to get the same flexibility as CBR streams.

VBR (variable bitrate) options:

-v use variable bitrate **(--vbr-new)**

--vbr-old

Invokes the oldest, most tested VBR algorithm. It produces very good quality files, though is not very fast. This has, up through v3.89, been considered the "workhorse" VBR algorithm.

--vbr-new

Invokes the newest VBR algorithm. During the development of version 3.90, considerable tuning was done on this algorithm, and it is now considered to be on par with the original **--vbr-old.** It has the added advantage of being very fast (over twice as fast as **--vbr-old**). This is the default since 3.98.

-V *n*

0 <= *n* <= 9.999
Enable VBR (Variable BitRate) and specifies the value of VBR quality (default = 4). Decimal

values can be specified, like 4.51.
0 = highest quality.

ABR and VBR options:

-b *bitrate*

For MPEG-1 (sampling frequencies of 32, 44.1 and 48 kHz)
n = 32, 40, 48, 56, 64, 80, 96, 112, 128, 160, 192, 224, 256, 320

For MPEG-2 (sampling frequencies of 16, 22.05 and 24 kHz)
n = 8, 16, 24, 32, 40, 48, 56, 64, 80, 96, 112, 128, 144, 160

For MPEG-2.5 (sampling frequencies of 8, 11.025 and 12 kHz)
n = 8, 16, 24, 32, 40, 48, 56, 64

Specifies the minimum bitrate to be used. However, in order to avoid wasted space, the smallest frame size available will be used during silences.

-B *bitrate*

For MPEG-1 (sampling frequencies of 32, 44.1 and 48 kHz)
n = 32, 40, 48, 56, 64, 80, 96, 112, 128, 160, 192, 224, 256, 320

For MPEG-2 (sampling frequencies of 16, 22.05 and 24 kHz)
n = 8, 16, 24, 32, 40, 48, 56, 64, 80, 96, 112, 128, 144, 160

For MPEG-2.5 (sampling frequencies of 8, 11.025 and 12 kHz)
n = 8, 16, 24, 32, 40, 48, 56, 64

Specifies the maximum allowed bitrate.

Note: If you own an mp3 hardware player build upon a MAS 3503 chip, you must set maximum bitrate to no more than 224 kpbs.

-F Strictly enforce the **-b** option.
This is mainly for use with hardware players that do not support low bitrate mp3.

Without this option, the minimum bitrate will be ignored for passages of analog silence, i.e. when the music level is below the absolute threshold of human hearing (ATH).

Experimental options:

-X *n*

0 <= *n* <= 7

When LAME searches for a "good" quantization, it has to compare the actual one with the best one found so far. The comparison says which one is better, the best so far or the actual. The **-X** parameter selects between different approaches to make this decision, **-X0** being the default mode:

-X0

The criteria are (in order of importance):
* less distorted scalefactor bands
* the sum of noise over the thresholds is lower
* the total noise is lower

-X1

The actual is better if the maximum noise over all scalefactor bands is less than the best so far.

-X2

The actual is better if the total sum of noise is lower than the best so far.

-X3

The actual is better if the total sum of noise is lower than the best so far and the maximum noise over all scalefactor bands is less than the best so far plus 2dB.

-X4

Not yet documented.

-X5

The criteria are (in order of importance):
* the sum of noise over the thresholds is lower
* the total sum of noise is lower

-X6

The criteria are (in order of importance):
* the sum of noise over the thresholds is lower
* the maximum noise over all scalefactor bands is lower
* the total sum of noise is lower

-X7

The criteria are:
* less distorted scalefactor bands
or
* the sum of noise over the thresholds is lower

-Y lets LAME ignore noise in sfb21, like in CBR

MP3 header/stream options:

-e *emp*
emp = n, 5, c

n = (none, default)
5 = 0/15 microseconds
c = citt j.17

All this does is set a flag in the bitstream. If you have a PCM input file where one of the above types of (obsolete) emphasis has been applied, you can set this flag in LAME. Then the mp3 decoder should de-emphasize the output during playback, although most decoders ignore this flag.

A better solution would be to apply the de-emphasis with a standalone utility before encoding, and then encode without **-e.**

-c Mark the encoded file as being copyrighted.

-o Mark the encoded file as being a copy.

-p Turn on CRC error protection.
It will add a cyclic redundancy check (CRC) code in each frame, allowing to detect transmission errors that could occur on the MP3 stream. However, it takes 16 bits per frame that would otherwise be used for encoding, and then will slightly reduce the sound quality.

--nores
Disable the bit reservoir. Each frame will then become independent from previous ones, but the quality will be lower.

--strictly-enforce-ISO
With this option, LAME will enforce the 7680 bit limitation on total frame size.
This results in many wasted bits for high bitrate encodings but will ensure strict ISO compatibility.

This compatibility might be important for hardware players.

Filter options:

--lowpass *freq*
Set a lowpass filtering frequency in kHz. Frequencies above the specified one will be cutoff.

--lowpass-width *freq*
Set the width of the lowpass filter. The default value is 15% of the lowpass frequency.

--highpass *freq*
Set an highpass filtering frequency in kHz. Frequencies below the specified one will be cutoff.

--highpass-width *freq*
Set the width of the highpass filter in kHz. The default value is 15% of the highpass frequency.

--resample *sfreq*
sfreq = 8, 11.025, 12, 16, 22.05, 24, 32, 44.1, 48
Select output sampling frequency (only supported for encoding).
If not specified, LAME will automatically resample the input when using high compression ratios.

ID3 tag options:

--tt *title*
audio/song title (max 30 chars for version 1 tag)

--ta *artist*
audio/song artist (max 30 chars for version 1 tag)

--tl *album*
audio/song album (max 30 chars for version 1 tag)

--ty *year*
audio/song year of issue (1 to 9999)

--tc *comment*
user-defined text (max 30 chars for v1 tag, 28 for v1.1)

--tn *track[/total]*
audio/song track number and (optionally) the total number of tracks on the original recording. (track and total each 1 to 255. Providing just the track number creates v1.1 tag, providing a total forces v2.0).

--tg *genre*
audio/song genre (name or number in list)

--tv *id=value*
Text or URL frame specified by id and value (v2.3 tag). User defined frame. Syntax: --tv "TXXX=description=content"

--add-id3v2
force addition of version 2 tag

--id3v1-only
add only a version 1 tag

--id3v2-only
add only a version 2 tag

--id3v2-latin1
add following options in ISO-8859-1 text encoding.

--id3v2-utf16
add following options in unicode text encoding.

--space-id3v1
pad version 1 tag with spaces instead of nulls

--pad-id3v2
same as --pad-id3v2-size 128

--pad-id3v2-size num
adds version 2 tag, pad with extra "num" bytes

--genre-list
print alphabetically sorted ID3 genre list and exit

--ignore-tag-errors

ignore errors in values passed for tags, use defaults in case an error occurs

Analysis options:

-g run graphical analysis on <infile>. <infile> can also be a .mp3 file. (This feature is a compile time option. Your binary may for speed reasons be compiled without this.)

ID3 TAGS

LAME is able to embed ID3 v1, v1.1 or v2 tags inside the encoded MP3 file. This allows to have some useful information about the music track included inside the file. Those data can be read by most MP3 players.

Lame will smartly choose which tags to use. It will add ID3 v2 tags only if the input comments won't fit in v1 or v1.1 tags, i.e. if they are more than 30 characters. In this case, both v1 and v2 tags will be added, to ensure reading of tags by MP3 players which are unable to read ID3 v2 tags.

ENCODING MODES

LAME is able to encode your music using one of its 3 encoding modes: constant bitrate (CBR), average bitrate (ABR) and variable bitrate (VBR).

Constant Bitrate (CBR)

This is the default encoding mode, and also the most basic. In this mode, the bitrate will be the same for the whole file. It means that each part of your mp3 file will be using the same number of bits. The musical passage being a difficult one to encode or an easy one, the encoder will use the same bitrate, so the quality of your mp3 is variable. Complex parts will be of a lower quality than the easiest ones. The main advantage is that the final files size won't change and can be accurately predicted.

Average Bitrate (ABR)

In this mode, you choose the encoder will maintain an average bitrate while using higher bitrates for the parts of your music that need more bits. The result will be of higher quality than CBR encoding but the average file size will remain predictable, so this mode is highly recommended over CBR. This encoding mode is similar to what is referred as vbr in AAC or Liquid Audio (2 other compression technologies).

Variable bitrate (VBR)

In this mode, you choose the desired quality on a scale from 9 (lowest quality/biggest distortion) to

0 (highest quality/lowest distortion). Then encoder tries to maintain the given quality in the whole file by choosing the optimal number of bits to spend for each part of your music. The main advantage is that you are able to specify the quality level that you want to reach, but the inconvenient is that the final file size is totally unpredictable.

PRESETS

The **--preset** switches are aliases over LAME settings.

To activate these presets:

For VBR modes (generally highest quality):

--preset medium
This preset should provide near transparency to most people on most music.

--preset standard
This preset should generally be transparent to most people on most music and is already quite high in quality.

--preset extreme
If you have extremely good hearing and similar equipment, this preset will generally provide slightly higher quality than the **standard** mode.

For CBR 320kbps (highest quality possible from the **--preset** switches):

--preset insane
This preset will usually be overkill for most people and most situations, but if you must have the absolute highest quality with no regard to filesize, this is the way to go.

For ABR modes (high quality per given bitrate but not as high as VBR):

--preset kbps
Using this preset will usually give you good quality at a specified bitrate. Depending on the bitrate entered, this preset will determine the optimal settings for that particular situation. While this approach works, it is not nearly as flexible as VBR, and usually will not attain the same level of quality as VBR at higher bitrates.

cbr If you use the ABR mode (read above) with a significant bitrate such as 80, 96, 112, 128, 160, 192, 224, 256, 320, you can use the **--preset cbr kbps** option to force CBR mode encoding instead of the standard ABR mode. ABR does provide higher quality but CBR may be useful in situations such as when streaming an MP3 over the Internet may be important.

EXAMPLES

Fixed bit rate jstereo 128kbs encoding:

lame -b *128 sample.wav sample.mp3*

Fixed bit rate jstereo 128 kbps encoding, highest quality:

lame -q 0 -b *128 sample.wav sample.mp3*

To disable joint stereo encoding (slightly faster, but less quality at bitrates <= 128 kbps):

lame -m *s sample.wav sample.mp3*

Variable bitrate (use -V n to adjust quality/filesize):

lame -V *2 sample.wav sample.mp3*

Streaming mono 22.05 kHz raw pcm, 24 kbps output:

cat *inputfile* **| lame -r -m** *m* **-b** *24* **-s** *22.05* **- - >** *output*

Streaming mono 44.1 kHz raw pcm, with downsampling to 22.05 kHz:

cat *inputfile* **| lame -r -m** *m* **-b** *24* **--resample** *22.05* **- - >** *output*

Encode with the **standard** preset:

lame --preset standard *sample.wav sample.mp3*

BUGS

Probably there are some.

SEE ALSO

mpg123(1)**, madplay**(1)**, sox**(1)

AUTHORS

LAME originally developed by Mike Cheng and now maintained by Mark Taylor, and the LAME team.

GPSYCHO psycho-acoustic model by Mark Taylor.
(See http://www.mp3dev.org/).

mpglib by Michael Hipp

Manual page by William Schelter, Nils Faerber, Alexander Leidinger, and Rogèrio Brito.