Analyzer Documentation

Prepared by: Tristan Jehan, CSO
David DesRoches, Lead Audio Engineer

January 7, 2014
Analyzer Version: 3.2

The Echo Nest Corporation 48 Grove St. Suite 206, Somerville, MA 02144 (617) 628-0233 tristan@echonest.com http://the.echonest.com

Introduction

"Analyze" is a music audio analysis tool available as a free public web API (visit developer.echonest.com) and a stand-alone command-line binary program for commercial partners (contact biz@echonest.com). The program takes a digital audio file from disk (e.g. mp3, m4a, wav, aif, mov, mpeg, flv), or audio data piped in on the command line. It generates a JSON-formatted text file that describes the track's structure and musical content, including rhythm, pitch, and timbre. All information is precise to the microsecond (audio sample).

Analyze is the world's only "music listening" API. It uses proprietary machine listening techniques to simulate how people perceive music. It incorporates principles of psychoacoustics, music perception, and adaptive learning to model both the physical and cognitive processes of human listening. The output of analyze contains a complete description of all musical events, structures, and global attributes such as key, loudness, time signature, tempo, beats, sections, harmony. It allows developers to create applications related to the way people hear and interact with music.

The output data allows developers to

1) *interpret*: understand, describe, and represent music.
   Applications include music similarity, playlisting, music visualizers, and analytics.

2) *synchronize*: align music with other sounds, video, text, and other media.
   Applications include automatic soundtrack creation and music video games.

3) *manipulate*: remix, mashup, or process music by transforming its content.
   An example is the automatic ringtone application mashtone for the iPhone.

Output Data

- **meta data**: analyze, compute, and track information.
- **track data**
  - **time signature**: an estimated overall time signature of a track. The time signature (meter) is a notational convention to specify how many beats are in each bar (or measure).
  - **key**: the estimated overall key of a track. The key identifies the tonic triad, the chord, major or minor, which represents the final point of rest of a piece.
  - **mode**: indicates the modality (major or minor) of a track, the type of scale from which its melodic content is derived.
  - **tempo**: the overall estimated tempo of a track in beats per minute (BPM). In musical terminology, tempo is the speed or pace of a given piece and derives directly from the average beat duration.
  - **loudness**: the overall loudness of a track in decibels (dB). Loudness values in the Analyzer are averaged across an entire track and are useful for comparing relative loudness of segments and tracks. Loudness is the quality of a sound that is the primary psychological correlate of physical strength (amplitude).
  - **duration**: the duration of a track in seconds as precisely computed by the audio decoder.
  - **end of fade in**: the end of the fade-in introduction to a track in seconds.
  - **start of fade out**: the start of the fade out at the end of a track in seconds.
  - **codestring, echoprintstring**: these represent two different audio fingerprints computed on the audio and are used by other Echo Nest services for song identification. For more information on Echoprint, see http://echoprint.me.
  - **synchstring**: a synchronization code that allows a client player to synchronize the analysis data to the audio waveform with sample accuracy, regardless of its decoder type or version. See Synchstring section.

- **rhythmstring**: a representation of spectro-temporal transients as binary events. This temporal data distributed on 8 frequency channels aims to be independent of timbre and pitch representations. See Rhythmstring section.

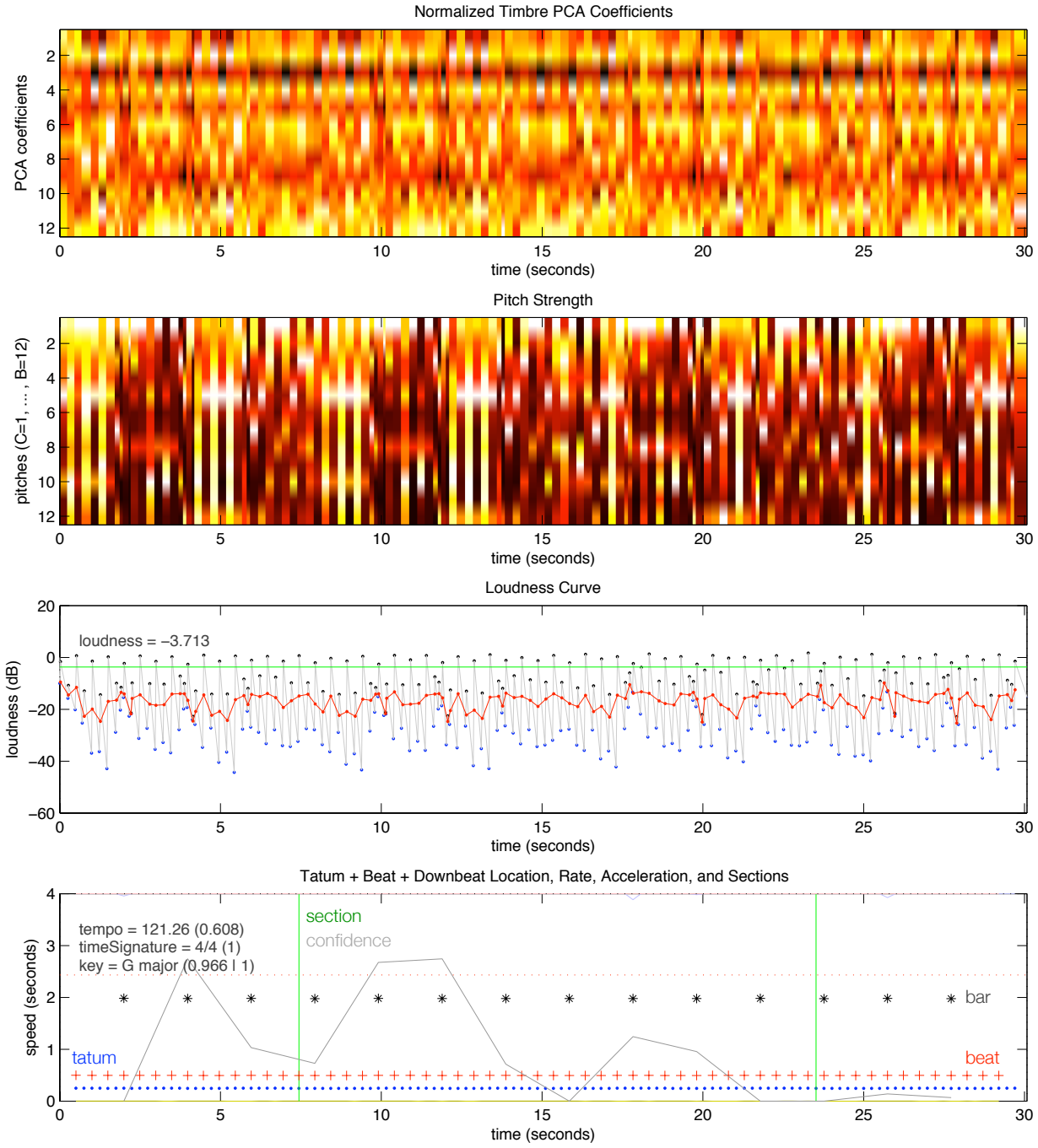
- **timbre**, **pitch**, and **loudness** are described in detail as part of the *segments* interpretation below.
- **sequenced data**: the Analyzer breaks down the audio into musically relevant elements that occur sequenced in time. From smallest to largest those include:
  - **segments**: a set of sound entities (typically under a second) each relatively uniform in timbre and harmony. Segments are characterized by their perceptual onsets and duration in seconds, loudness (dB), pitch and timbral content.
    - **loudness_start:** indicates the loudness level at the start of the segment
    - **loudness_max_time:** offset within the segment of the point of maximum loudness
    - **loudness_max:** peak loudness value within the segment
  - **tatums**: list of tatum markers, in seconds. Tatums represent the lowest regular pulse train that a listener intuitively infers from the timing of perceived musical events (segments).
  - **beats**: list of beat markers, in seconds. A beat is the basic time unit of a piece of music; for example, each tick of a metronome. Beats are typically multiples of tatums.
  - **bars**: list of bar markers, in seconds. A bar (or measure) is a segment of time defined as a given number of beats. Bar offsets also indicate downbeats, the first beat of the measure.
  - **sections**: a set of section markers, in seconds. Sections are defined by large variations in rhythm or timbre, e.g. chorus, verse, bridge, guitar solo, etc. Each section contains its own descriptions of tempo, key, mode, time_signature, and loudness.

JSON Schema Example


```
{
  “meta”:
      {
        “analyzer_version”:”3.08b”, "detailed_status":"OK", "filename":"/Users/Jim/
        Desktop/file.mp3", "artist":"Michael Jackson", "album":"Thriller",
        "title":"Billie Jean", "genre":"Rock", "bitrate":192, "sample_rate":44100,
        "seconds":294, "status_code":0, "timestamp":1279120425, "analysis_time":
        3.83081
      },
  “track”:
      {
        "num_samples":6486072, "duration":294.15293,
        "sample_md5":"0a84b8523c00b3c8c42b2a0eaabc9bcd", "decoder":"mpg123",
        "offset_seconds":0, "window_seconds":0, "analysis_sample_rate":22050,
        "analysis_channels":1, "end_of_fade_in":0.87624, "start_of_fade_out":
        282.38948, "loudness":-7.078, "tempo":117.152, "tempo_confidence":0.848,
        "time_signature":4, "time_signature_confidence":0.42, "key":6,
        "key_confidence":0.019, "mode":1, "mode_confidence":0.416,
        "codestring":"eJwdk8U7m4Rz9Pej...tbtSnk8U7m4Rz980uF", "code_version":3.15,
        "echoprintstring":"eJzFnQuyrTquZbsENjamOf5A_5t...pDF6eF__7eH_D9MWE8p",
        "echoprint_version": 4.12, "synchstring": "eJxlWwmWJCsOu0ocIWz2-1-
        ssSRDZP...nUf0aeyz4=", "synch_version": 1.00, "rhythmstring":
        “eJxVnAmyHDcM...9v71b_92-5V-fmWuX2n”, "rhythm_version": 1.00
```

```
        },
    “bars”:
        [{"start":1.49356, "duration":2.07688, "confidence":0.037}, ...],
    "beats":
        [{"start":0.42759, "duration":0.53730, "confidence":0.936}, ...],
    "tatums":
        [{"start":0.16563, "duration":0.26196, "confidence":0.845}, ...],
    "sections":
        [{"start":0.00000, "duration":8.11340, "confidence":1.000,
        "loudness": -15.761, "tempo": 135.405, "tempo_confidence": 0.938,
        "key": 0, "key_confidence": 0.107, "mode": 1, "mode_confidence": 0.516,
        "time_signature": 4, "time_signature_confidence": 1.000}, ...],
    "segments":
        [{
          "start":0.00000, "duration":0.31887, "confidence":1.000,
          "loudness_start":-60.000, "loudness_max_time":0.10242,
          "loudness_max":-16.511, "pitches":[0.370, 0.067, 0.055, 0.073, 0.108, 0.082,
          0.123, 0.180, 0.327, 1.000, 0.178, 0.234], "timbre":[24.736, 110.034, 57.822,
          -171.580, 92.572, 230.158, 48.856, 10.804, 1.371, 41.446, -66.896, 11.207]
         }, ...]
}
```

Interpretation



Plot of the JSON data for a 30-second excerpt of “around the world” by Daft Punk.

Rhythm

Beats are subdivisions of bars. Tatums are subdivisions of beats. That is, bars always align with a beat and ditto tatums. Note that a low confidence does not necessarily mean the value is inaccurate. Exceptionally, a *confidence* of -1 indicates "no" value: the corresponding element must be discarded. A *track* may result with no *bar*, no *beat*, and/or no *tatum* if no periodicity was detected. The *time signature* ranges from 3 to 7 indicating time signatures of 3/4, to 7/4. A value of -1 may indicate no time signature, while a value of 1 indicates a rather complex or changing time signature.

Pitch

The *key* is a track-level attribute ranging from 0 to 11 and corresponding to one of the 12 keys: C, C#, D, etc. up to B. If no key was detected, the value is -1. The *mode* is equal to 0 or 1 for "minor" or "major" and may be -1 in case of no result. Note that the major key (e.g. C major) could more likely be confused with the minor key at 3 semitones lower (e.g. A minor) as both keys carry the same pitches. Harmonic details are given in *segments* below.

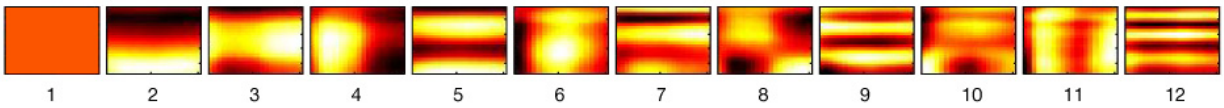
Segments

Beyond timing information (start, duration), segments include *loudness*, *pitch*, and *timbre* features.

- *loudness* information (i.e. attack, decay) is given by three data points, including dB value at onset (*loudness_start*), dB value at peak (*loudness_max*), and segment-relative offset for the peak loudness (*loudness_max_time)*. The dB value at onset is equivalent to the dB value at offset for the preceding *segment*. The last segment specifies a dB value at offset (*loudness*_end) as well.

- *pitch* content is given by a "chroma" vector, corresponding to the 12 pitch classes C, C#, D to B, with values ranging from 0 to 1 that describe the relative dominance of every pitch in the chromatic scale. For example a C Major chord would likely be represented by large values of C, E and G (i.e. classes 0, 4, and 7). Vectors are normalized to 1 by their strongest dimension, therefore noisy sounds are likely represented by values that are all close to 1, while pure tones are described by one value at 1 (the pitch) and others near 0.

- *timbre* is the quality of a musical note or sound that distinguishes different types of musical instruments, or voices. It is a complex notion also referred to as sound color, texture, or tone quality, and is derived from the shape of a segment's spectro-temporal surface, independently of pitch and loudness. The Echo Nest Analyzer's *timbre* feature is a vector that includes 12 unbounded values roughly centered around 0. Those values are high level abstractions of the spectral surface, ordered by degree of importance. For completeness however, the first dimension represents the average loudness of the segment; second emphasizes brightness; third is more closely correlated to the flatness of a sound; fourth to sounds with a stronger attack; etc. See an image below representing the 12 basis functions (i.e. template segments). The actual timbre of the segment is best described as a linear combination of these 12 basis functions weighted by the coefficient values: timbre = c1 x b1 + c2 x b2 + ... + c12 x b12, where c1 to c12 represent the 12 coefficients and b1 to b12 the 12 basis functions as displayed below. Timbre vectors are best used in comparison with each other.



12 basis functions for the timbre vector: x = time, y = frequency, z = amplitude

Confidence Values

Many elements at the track and lower levels of analysis include confidence values, a floating-point number ranging from 0.0 to 1.0. *Confidence* indicates the reliability of its corresponding attribute. Elements carrying a small confidence value should be considered speculative. There may not be sufficient data in the audio to compute the element with high certainty.

Synchstring

With Analyzer v3.08, a new data string was introduced. It works with a simple synchronization algorithm to be implemented on the client side, which generates offset values in numbers of samples for 3 locations in the decoded

waveform, the beginning, the middle, and the end. These offsets allow the client application to detect decoding errors (when offsets mismatch). They provide for synching with sample accuracy, the JSON timing data with the waveform, regardless of which mp3 decoder was used on the client side (quicktime, ffmpeg, mpg123, etc.) Since every decoder makes its own signal-dependent offset and error correction, sample accuracy isn't manageable by other means, such as decoder type and version tracking. For implementation examples of the synchronization algorithm, please go to the github repository at http://github.com/echonest/synchdata.

Rhythmstring

With Analyzer v3.2 was introduce the rhythmstring, a binary representation of rhythmic impulses, or transients, over 8 frequency channels. The encoded format goes as follows:

Fs Hop Nch <Nos Oi do_1 ... do_n> ... <Nos Oi do_1 ... do_n> where:

Fs: sampling rate
Hop: hop size in samples
Nch: number of channels
Nos: number of onsets
Oi: initial onset frame
do_n: number of frames to the next onset

See http://github.com/echonest/synchdata for relevant decoding code.