Audio Engineering Society

Convention Paper

Presented at the 137th Convention
2014 October 9–12 Los Angeles, USA

An Evaluation of Chromagram Weightings for Automatic Chord Estimation

Zhengshan Shi[1] and Julius Smith[1]

[1] *Center for Computer Research in Music and Acoustics, Stanford University, Stanford, CA 94305-8180, USA*

Correspondence should be addressed to Zhengshan Shi (kittyshi@ccrma.stanford.edu)

ABSTRACT

Automatic Chord Estimation (ACE) is a central task in Music Information Retrieval. Generally, audio files are parsed into chroma-based features for further processing in order to estimate the chord being played. Much work has been done to improve the estimation algorithm by means of statistical models for chroma vector transitions, but not as much attention has been given to the loudness model during the feature extraction stage. In this paper, we evaluate the effect on chord-recognition accuracy due to the use of various nonlinear transformations and loudness weightings applied to the power spectrum that is "folded" to form the chromagram in which chords are detected. Nonlinear spectral transformations included square-root magnitude, magnitude, magnitude-squared (power spectrum), and dB magnitude. Weightings included A-weighted dB and Gaussian-weighted magnitude.

1. INTRODUCTION

Automatic Chord Estimation (ACE) has been widely explored since Fujishima [Fuj99] introduced a twelve-dimensional Pitch Class Profile, now called the *chromagram*, using 12 semitones to represent the energy distribution versus semitone in one octave. An excellent state-of-the-art review was published in 2014 [MSND14], and a recent evaluation of various methods appears in [CB14]. In particular, the winner of the 2012 MIREX (Music Information Retrieval Evaluation eXchange) chord estimation evaluation task was the Harmony Progression Analyzer (HPA) system [NMSD12].[1] The HPA system uses a beat-synchronous loudness chromagram (A-weighted dB) as a feature vector along with a rich Hidden Markov Model (HMM).

The basic framework of an automatic chord estimation system is usually comprised of two stages: the chroma feature calculation, and the chord estimation

[1] https://patterns.enm.bris.ac.uk/hpa-software-package/

classifier. Each chord estimation system is developed independently with different combinations of modules and data sets, and thus it is hard to tell which module makes the best contribution to the final results. Also, the computational efficiency of chord estimation systems is not normally considered. It is therefore unfair to declare a best state-of-the-art system based on limited evaluation results.

Cho & Bello [CB14] did a thorough evaluation of the interactions among different chord estimation techniques, comparing various chroma features, filtering strategies, and stochastic chord chroma models systematically. Their evaluation experiments conclude that the performance of such systems is predominantly affected by the initial feature extraction stage, especially features that use a combination of overtone removal and timbre homogenization. This motivates us to investigate further the feature extraction stage, especially regarding the chromagram formation.

Since music is written, recorded, and mixed by people, the dynamic range and intensity level of a song are based mainly on auditory perception, i.e., by how loud we hear different layers of instruments and notes. Thus a loudness-weighted spectrogram should be more accurate in representing perception of audio spectral content. However, most of the currently existing chromagram computations do not take audio perception into account, and little emphasis has been placed on the importance of loudness-based feature extractions.

In this paper, we compare the results using different types of spectral weightings during the chromagram formation stage, and address the importance of forming a loudness-based chromagram in the front end feature extraction stage.

2. SYSTEM IMPLEMENTATION

The software system used for these tests is available on GitHub[2] and was configured as follows:

1. The input signal is downsampled, if necessary, to a desired sampling rate which here is chosen to be $f_s = 44100/6 = 7350$ Hz. A range of higher sampling rates was tried and found not to make an appreciable difference in the results. It is assumed that the input songs are tuned to standard pitch, but this was not always the case (e.g., "Lovely Rita" by the Beatles).

2. The parameter $f_{\min} = 65.4$ Hz (pitch C_2) determines the center-frequencies of the constant-Q filter bank. The filters are tuned to standard equal-tempered semitone frequencies, 12 per octave, up to a maximum determined by the sampling rate such that all octaves are complete. (I.e., any top partial octave is discarded.) This gave us five octaves starting at C_2. The Q of each filter in the filter bank, defined as the center-frequency divided by 3dB-bandwidth, is set to $1/(2^{1/12} - 1)$.

3. The constant-Q filter bank is implemented in the usual way as a weighted sum of squared-magnitude FFT samples. The FFT window length is set to give better than semitone resolution at $f_{\min}$, yielding about a half-second long window, and the FFT size is taken to be a higher power of 2. The window type is Hamming, and its hop size is set for 75% overlap (0.13 s). At low frequencies this is a robust choice, while at high frequencies the spectrogram becomes effectively undersampled in time due to the synthesis of much wider constant-Q bands. The spectral weighting used in forming each constant-Q filter band consists of two half-Hann windows, the left half spanning from the band's center frequency to the next lower center frequency, and the right half spanning the longer distance from the band's center frequency to the next higher center frequency. The filter bands are normalized for unity peak gain, so that higher bands are generally more energetic. This can of course be compensated in any desired way by spectral weighting.

4. The result of the preceding step is a vector of constant-Q filter-bank output-power, sampled once per FFT frame. We evaluated the effects of the following spectral weightings:

   $\sqrt{|X|}$ — square root of the magnitude spectrum
   $|X|$ — magnitude spectrum
   $|X|^2$ — power spectrum
   dB — power spectrum in decibels

[2] https://github.com/josmithiii/ace/

dB-A — A-weighted dB power spectrum (see below)

$|X|$-G — Gaussian-weighted magnitude spectrum (see below)

5. After the weighting is applied in the previous step, the *chromagram* is formed by summing all of the octaves down to one. This is conveniently carried out by the following matlab statement:

```
chromagram = ...
 kron(ones(1,noctaves),eye(12)) ...
  * weightedPowerSpectrum;
```

where `noctaves` is the number of octaves spanned by the constant-Q filter bank (5 in our case).

6. An odd-length median filter spanning a specified duration is passed over the each row of the chromagram to eliminate outliers caused by percussion hits and transition frames.

7. A simple frame classification is performed to distinguish between "silence" and "music". In the future, we would like to extend this to include a preliminary spectrum analysis that ensures at least three tonal spectral peaks are present. A frame declared to be silence is marked 'N' for "No Chord" and not processed further.

8. As is common practice, chords are detected in the chromagram by means of "matched filtering" each chromagram by a binary "chord template" vector having a 1 at each semitone in the chord, and 0 at all other semitones. For example, the template vector ("matched spectral filter") for a major C triad is $[1, 0, 0, 0, 1, 0, 0, 1, 0, 0, 0, 0]$, which selects 'C', 'E', and 'G' out of the twelve semitones starting at 'C'. In addition to searching for the 12 possible major triads starting on any semitone, we also test for the 12 minor triads, giving 24 test-chords all together. In another case, we include seventh chords, yielding a total of 48 chords (48 rows) in the template matrix. For seventh chords, since there are four tones, we use 3/4 instead of 1 for the template value at each included semitone so that triads are not superseded by seventh chords; in other words, if all chord-notes are at amplitude 1 in the chromagram, then both a seventh chord and a triad will obtain a score of 3.

   Let C denote the $12 \times N_t$ matrix of chromagrams, where N_t is the number of time frames (FFT hops) present. Then the chord-score matrix S is given by

$$S = T \cdot C$$

   where '$\cdot$' denotes normal matrix multiplication, and T is a 24×12 or 48×12 matrix of chord templates, i.e., the ith row is a "matched filter" for the ith chord. The detected chord for each time step (frame center time) is taken to be the one corresponding to the maximum chord-score at each time step. In matlab notation,

```
[~, Chords] = max(S);
```

   where we note that `max(S)` for a matrix `S` returns a row vector containing the maximum from each column.

9. Normally, a statistical model is used for chord estimation improvement. A Hidden Markov Model (HMM) is most often used to find the maximum-likelihood chord sequence based on a transition-probability matrix obtained from annotated music data. However, since the aim of our system was to study the impact of different spectral weightings in the front end on chord-recognition accuracy, we didn't include the statistical model.

   For detailed plots illustrating internal operations and intermediate spectra in our ACE system, one may execute the available GitHub software in either Octave or Matlab with the variable `doPlot` set to 1. (All of the control variables are near the beginning of the main script `ace/ACE.m`).

3. SPECTRAL WEIGHTINGS

The initial motivation for this work was to investigate the use of high quality loudness models in the formation of the chromagram.

Loudness can be described as the level of sensation due to a sound. As is well known, perceptual audio coding (e.g., .mp3, .m4a audio files) exploits auditory masking to compress the original audio file by eliminating masked spectral content that we don't hear.

Fletcher & Munson introduced the equal-loudness contour [FM33], giving us a picture of how amplitudes are perceived by the ear as a function of frequency across the audio range. For example, the ear is more sensitive in the 2-4 kHz frequency range.

The use of loudness weighting (A-weighted dB) was apparently introduced for ACE in [NMSD12]. A-weighting is based on the 40-phon equal loudness curve of Fletcher and Munson. We wanted to take this farther by going on to the definition of *specific loudness* (perceptual loudness versus time and frequency) introduced by Moore, Glasberg, and Baer (MGB) [GM02].

The MGB specific-loudness distribution was expected to provide the most psychoacoustically accurate definition and implementation of a "loudness spectrogram". It was expected that summing loudness from all octaves would provide the ideal psychoacoustic chromagram. For this we used the Genesis implementation of MGB.[3] However, while the MGB specific-loudness is known to sum across frequency to provide a time-varying loudness estimate that performs very well in predicting perceived loudness versus time, it was found not to serve as a good basis for ACE, due to insufficient resolution of fundamental frequencies in chords at low registers, and even the relative loudness of tones in a chord is not always well predicted in accordance with what is heard. Finally, the computational complexity is very high in comparison to the weighted short-time power spectra used here. As far as we know, it remains a research opportunity to define "specific loudness" in a way that is appropriate for automatic chord estimation and related problems.

Absent an appropriate and available loudness spectrogram, we decided to investigate the effect of various spectral weightings and nonlinear transformations, including some we had seen implemented in the literature.

[3] `http://www.genesis.fr`

In this paper, we study the effects on ACE of the weightings/transformation listed in Section 2, Item 4 above. The square-root magnitude spectrum is commonly used as similar to a log, but without the difficulty of dealing with zeros mapping to $-\infty$ dB. The magnitude spectrum is commonly used in the ACE community. Magnitude squared is the classic power spectrum and tests whether exaggerated variance in the spectral magnitude might be beneficial. It also is most appropriate physically to sum power when forming the chromagram. These three cases are analogous to L_p norms for $p = 1/2$, $p = 1$, and $p = 2$. It is well known that L_2 error norms are more sensitive to outliers than L_1 error norms, and an $L_{1/2}$ norm is less sensitive than L_1.

We used two dB cases: (1) Unweighted dB has been extensively used in audio since the early days of Bell Telephone Laboratories. (2) A-weighted dB is used in [NMSD12] and provides approximate equal-loudness at the relatively low listening level of 40 phons. Finally, the Gaussian-weighted magnitude case is summarized in [CB14]. It de-emphasizes high frequencies as an "overtone-removal method", and reduces low frequencies in a manner similar to an equal-loudness weighting. We initially implemented Gaussian-weighted dB, but G-weighted magnitude performed slightly better.

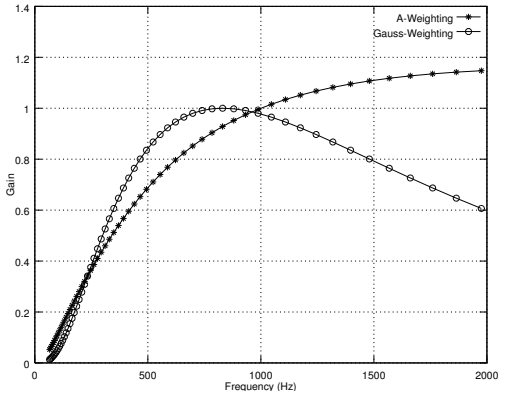

Fig. 1: Overlay of A-weighting and Gaussian-weighting.

Figure 1 shows an overlay of the A-weighting and Gaussian-weighting used as a function of constant-Q filter-bank center frequency. We see that A-weighting reduces low-frequency amplitudes, in ac-

cordance with perception, but the high frequencies do not extend far enough to see the reduced sensitivity at high frequencies above 4 kHz. Gaussian weighting, on the other hand, does significantly reduce amplitudes above 1 kHz. The purpose of this is to reduce overtones, under the assumption that the fundamental frequencies lie below 1 kHz or so. The low-frequency weightings are comparable in the two cases.

All dB scales were normalized to the range 0–100 dB. While this is arbitrary, it does make a difference in the results. It would be preferable to use true dB SPL levels, with zero dB mapped to the threshold of hearing. This requires assuming a presentation level for each song and working with calibrated amplitudes. In any case, the lower dB limit must be clipped below at some reasonable level, and the audibility threshold is probably ideal.

4. EXPERIMENT AND EVALUATION

For our automatic chord estimation experiments, we used Christopher Harte's Beatles dataset[4] which includes 180 Beatles songs from 12 albums comprising more than 8 hours of music. We performed raw chord estimation without any training models, or post-filtering techniques, matching against binary chord templates of 24 chords (major and minor triads), and 48 chords (adding a minor seventh to each triad). The chord estimation accuracy is calculated using the standard MIREX Chord Symbol Recall (CSR) measure,[5] defined by

$$\mathrm{CSR} = \frac{D_{\mathrm{Correct}}}{D_{\mathrm{Total}}}$$

where D_{Correct} denotes the total duration of segments where annotation equals estimation, and D_{Total} is the total duration of the data set. The CSR gives insight into how the predicted chord matched the ground-truth chord labels among all songs.

In addition to detecting both 24 and 48 chords, we look at three different median-filter lengths: 0, 1.2 s, and 1.7 s. Length 0 corresponds to no median filtering and provides a baseline for comparison. The length 1.7 s corresponds to 13 frames at our sampling rate and hop size, which was the optimal length determined in [CWB10]. Since they reported their hop size and sampling rate to be 4096 samples (93 ms) and 44100 Hz, respectively, we also tried a median-smoother length of 1.2 s which equals the time span in their study. Interestingly, as shown in the results below, 1.7 s gave the best performance, indicating that the absolute order of the median smoother (13 frames) is more important than its time span. We believe this observation merits further investigation.

On a Linux computer with an Intel i7-3820K processor (4 cores, 8 threads, 1.44 GHz clock) and 32GB of RAM, the algorithm runs (in Matlab) 215 times faster real-time for each setting of the median smoother length. Thus, 8 hours of music can be processed to generate the data matrix given in the Results section below in less than 7 minutes. On a 2012 MacBook Pro running Octave, the real-time ratio is typically near 60.

5. RESULTS

s	#c	$\sqrt{\lvert X\rvert}$	$\lvert X\rvert$	$\lvert X\rvert^2$	dB	dB-A	$\lvert X\rvert$-G
0	24	48.4	46.1	41.3	**48.6**	45.7	45.7
0	48	43.2	41.2	37.0	**43.6**	40.4	40.2
1.2	24	55.4	53.3	48.9	**56.2**	51.2	50.5
1.2	48	49.6	47.8	43.9	**50.4**	45.1	44.2
1.7	24	56.0	54.1	49.8	**56.9**	51.2	50.3
1.7	48	50.1	48.5	44.7	**51.0**	45.0	44.0

Fig. 2: Chord Symbol Recall CSR (total time duration of correct chord identification divided by the total duration of the dataset) for 180 Beatles songs comprising more than 8 hours of music. The 's' column gives the chromagram median-smoother length in seconds. The '#c' column gives the number of chords being estimated.

Figure 2 shows the MIREX Chord Symbol Recall (CSR) across 180 Beatles songs for each spectral weighting, for three different median-smoother lengths (0, 1.2 s, and 1.7 s), and for two chord template matrices, one for 24 chords and the other for 48 chords. We see that *the consistent winner is unweighted dB, followed closely by square-root magnitude*. Unweighted dB hits the accuracy rate of 48.6% for raw audio chord estimation, using no pre- or post-filtering [CB14], while the power-spectrum gives the least-accurate result. Unweighted dB outperforms the power spectrum by $\approx$ 7.3%. Also,

[4] http://www.isophonics.net/datasets

[5] http://www.music-ir.org/mirex/wiki/-2014:Audio_Chord_Estimation

we can see from the table that the G-weighted magnitude spectrum performs almost as well as A-weighted dB. Among the three magnitude-to-a-power cases, the magnitude square-root gives the best results, followed by magnitude, with squared-magnitude last. This ordering is in line with their respective outlier sensitivities. That is, squared-magnitude exaggerates the spectrum fluctuations more compared with magnitude, which in turn has wider variance than the magnitude square-root.

6. CONCLUSIONS

Log and square-root spectral-magnitude transformations outperformed the weighted magnitude- and log-transformations studied in this paper. Both the A and G weightings down-weight low frequencies, as does loudness perception, and the G weighting additionally attenuates high frequencies above the fundamental range. However, neither weighting proved to be helpful in our experiments. It appears that *the value of more psychoacoustically accurate weighting is outweighed by the wider-band information-emphasis given by uniform weighting.* At the same time, *the dynamic range reductions provided by the dB and square-root transformations of spectral magnitude appear to generally improve chord-classification using binary chord templates.* One can imagine that compressing the spectral dynamic range partially compensates for the mismatch in shape between the binary chord template and the more varied spectral profile occurring in the chromagram for each chord; the binary template's shape is invariant to spectral dynamic-range compression, while natural variations among observed chromagram peaks are reduced. For future work, we would like to find the optimal value of $p \in (0, 1)$ that provides the best performance using spectral magnitude to the pth power. In the other direction, binary chord templates can be replaced by more natural expected shapes for matched filtering, e.g. "stochastic chord models" [CB14].

7. ACKNOWLEDGMENTS

Thanks to Juan-Pablo Bello for initializing our efforts with some excellent references and pointers. Thanks also to Jonathan Abel for fruitful discussions and for suggesting the addition of the square-root magnitude case, which did almost as well as unweighted dB. Finally, thanks to Regina Collecchia for assistance with the ACE software.

8. REFERENCES

[CB14] T. Cho and J.P. Bello. On the relative importance of individual components of chord recognition systems. *Audio, Speech, and Language Processing, IEEE/ACM Transactions on*, 22(2):477–492, 2014.

[CWB10] T. Cho, R.J. Weiss, and J.P. Bello. Exploring common variations in state of the art chord recognition systems. In *Proceedings of the Sound and Music Computing Conference (SMC)*, pages 1–8, Barcelona, Spain, July 2010.

[FM33] H. Fletcher and W.A. Munson. Loudness, its definition, measurement and calculation. *Bell System Technical Journal*, 12(4):377–430, 1933.

[Fuj99] Takuya Fujishima. Realtime chord recognition of musical sound: a system using common lisp music. In *Proc. 1999 Int. Computer Music Conference, Beijing*, 1999.

[GM02] Brian R. Glasberg and Brian C. J. Moore. A model of loudness applicable to time-varying sounds. *Journal of the Audio Engineering Society*, 50(5):331–342, May 2002.

[MSND14] M. McVicar, R. Santos-Rodriguez, Y. Ni, and T. De Bie. Automatic chord estimation from audio: A review of the state of the art. *IEEE Transactions on Audio, Speech and Language Processing*, 22(2):556–575, 2014.

[NMSD12] Y. Ni, M. McVicar, R. Santos-Rodriguez, and T. De Bie. An end-to-end machine learning system for harmonic analysis of music. *IEEE Transactions on Audio, Speech and Language Processing*, 20(5):1771–1783, 2012. arxiv.org/pdf/1107.4969.