COMPARISON OF TEMPO ESTIMATION ALGORITHMS ON ACAPELLA SINGING VOICE

Benedikt Wimmer
UPF Barcelona
benedikt.wimmer01@estudiant.upf.edu

ABSTRACT

Tempo estimation is a fundamental discipline in Music Information Retrieval and has such been subject to extensive research in the last decades. In competitions like MIREX [1], the performance of systems for tempo estimation and related tasks like beat tracking have mainly been measured on musical mixtures. However, there are use cases such as automatic accompaniment generation or multimedia applications, where the assumption of musical mixtures does not hold true. This work investigates on the application of tempo estimation specifically on acapella voice. Three distinct algorithms – two of which are part of open-source software projects – have been evaluated on a number of 148 acapella vocal tracks and are discussed on their ability to provide correct information for the task of automatic accompaniment. An accompanying code repository to this research is also provided [2].

1. INTRODUCTION

There are multiple disciplines that build up on the information retrieved through tempo estimation as a baseline, one of them being automatic accompaniment as in [10]. Automatic accompaniment means that given an audio signal, a second audio signal is automatically created that is musically matching to the first signal. Therefore, a computational perceptive understanding of musical descriptors such as harmony, beat locations and meter is needed. This work aims to elaborate on tempo estimation as an baseline descriptor for rhythmic understanding. In this, the tempo estimation will evaluated as a global metric rather than as a temporal evolution.

In the past, there have been a number of significant innovations in the field of tempo estimation. For example, tempograms can be executed in various ways such as Fourier-transform, Autocorellation Cross-correlation, cyclic tempograms or even replaced by comb filter banks [6, 12, 14]. The so-called novelty curve for onset detection can be retrieved by means of traditional spectrograms, phase spectrograms or spectral convolution [6, 15]. In this paper, three architectures each using a different of these approaches to tempo estimation are evaluated on acapella singing voice. While we keep referencing the use case of automatic singing accompaniment for discussion, the results to this study do not lose their informational value for more general applications.

[1] https://www.music-ir.org/mirex/wiki/MIREX_HOME
[2] https://github.com/wimmerb/singing_tempo_detection

2. METHODOLOGY

Three algorithms, two of which are current tempo estimation implementations in the popular open source projects *librosa* [7] and *madmom* [2] and another implementation that is used in the time-stretching and pitch-correction system *VoAlign* proprietary to the company *Voctro Labs, S.L.* [3] have been used to create tempo estimations on 148 monoaural recordings of acapella singing voice from the *musdb* [13] database. After utilizing a computer-aided approach to achieve valid ground truth tempo annotations, the resulting estimates have then been evaluated against these reference annotations with common traditional accuracy (aswell as perceptual) metrics.

2.1 Musdb [13] with Tempo Annotation

The musdb dataset was first released in 2017 for the *sixth community-based Signal Separation Evaluation Campaign* (SiSEC 2018) [17] and such is originally created for the task of learning source separation. It consists of 150 songs that are between 13 seconds and $10\frac{1}{2}$ minutes, close to 4 minutes on average, in length. The songs are available as multitrack files, each consisting of separate stereo tracks for the each the musical mixture, vocals, drums, bass and other accompaniment [13].

To our knowledge, there exists no tempo annotation for this dataset at the time being for this study. However, we had planned to do tempo estimations on the vocal tracks of this dataset. An efficient way of annotating those tracks could be found by first applying two of the open source project *Essentia*'s [3] tempo estimation algorithms [3, 11] on the drum tracks that accompany the respective vocal tracks for each song in musdb and then – based on the output – manually curating the tempo estimation for each song. Incorrect estimations have been manually corrected and two songs were disregarded for the study as they contain multiple tempo changes.

[3] https://www.voctrolabs.com/

2.2 Algorithms

The first algorithm tested is the tempo algorithm implemented in librosa. A novelty curve based on spectral envelope is transformed to cyclic tempograms as in [6]. In this case, the temporal evolution of multiple such cyclic tempograms is aggregated into a single tempogram by averaging over time. The BPM value corresponding to the tempogram entry with the highest tempogram envelope would then be picked as the resulting tempo estimation.

The second algorithm tested is is the implementation of [1] featured in madmom. Here, the audio is transformed to three spectrograms of different per-frame window sizes. These spectrograms are then rescaled to logarithmic scale, which is motivated by the human perception of loudness [1]. The three spectrograms of different time resolutions are then stacked (similar to multiresolution spectrograms, but keeping the full frequency band for all three) and fed into an LSTM-based recurrent neural network as in [4]. The neural network is pre-trained with supervised training to predict beat onsets, such providing an output that is similar to a spectral novelty curve [5] as in the librosa algorithm, but rather depicting probabilities for beat occurrences over time. This so-called "beat activation" is now not transformed into a tempogram, but rather played through a bank of comb filters and a histogram of the most resonant comb filters for each point in time is created. Peak picking on this histogram leads to the final BPM estimate.

The third algorithm used – this will later be referred to as **VoAlign** – is proprietary to Voctro Labs S.L. and such information on it is protected. However, the specialty here is that – aiming towards correctly capturing the tempo of a voice over time – an approach that is related to what has been described for the algorithm featured in librosa is used on moving time windows of about 8 seconds in length. Tempo estimations are made individually for each of these moving (distance about 0.5 seconds) time windows and the global estimate is retrieved by averaging the BPM estimate for each of these windows.

2.3 Evaluation Metrics

The current standard for evaluating tempo estimation algorithms are perceptual metrics aswell as more traditional accuracy metrics [8]. Perceptual metrics as proposed by McKinney et al. [8, 9] include *one-correct, both-correct* and *P-Score*. The idea is that within a group of human annotators, there would naturally exist disagreement on the correct tempo estimation for a given piece. So with these perceptually motivated metrics, those disagreements would be reflected in scores that take into account two distinct tempo estimations together with their percentual support in a group of annotators. The algorithms that are to be evaluated would then need to produce a guess for two tempo estimates that should – in the best case – reflect both human annotated tempi for a given sound excerpt. The algorithms performance is then evaluated on its ability to guess:

- **one-correct:** if at least one of the two estimates matches with one of the human annotated references
- **both-correct:** if both estimates match with both human annotations
- **P-Score:** based on the percentages of annotators supporting the two reference tempi, how many percent of annotators would be matched correctly by the algorithm's estimates (always 100% if both correct)

For these scores, a number of human annotators would be required to annotate the given musical data. This requirement of multiple human annotations on the reference side aswell as algorithm estimate side could unfortunately not be fulfilled in this study, so the solution was to experiment with multiple heuristics, before finally choosing to use *half* the tempo of the given reference/estimate as an artificially constructed second tempo candidate.

Furthermore, in ACC_1 and ACC_2, two metrics were used for evaluation that only require a single tempo estimate and ground truth reference per song. ACC_1 is calculated by checking for each estimate if the relative difference between estimate and ground truth is within a certain tolerance rate (1) or not (0). The score is computed by the mean of this evaluation over all given datapoints (songs). In related works, these tolerance rates vary from 4% – which is assumed to be the human *just noticeable difference* for tempo – to 8% [16]. ACC_2 does follow the same principle, but provides compensation for so-called "octave errors". Octave errors occur when the tempo estimation is deviating from the reference tempo by a factor of 2 (binary meter) or 3 (ternary meter), such – more mathematically – deviating by a factor of $\frac{1}{2}$, 2, $\frac{1}{3}$ or 3. Those octave errors would indeed be matching to the ground truth tempo from a musical perspective, either building subdivisions or superdivisions to the annotated tempo.

For comparability with competitions such as MIREX, the aforementioned metrics were computed for tolerance thresholds of 4% aswell as 8%. Next to the proposed evaluation scores, error plots (estimation relative to reference) were used to facilitate interpretation of results.

Conjoined, all the mentioned score metrics have been implemented in the open-source projects *tempo_eval* [4] and *mir_eval* [5], implementations which were utilized in this study.

3. RESULTS

Figure 1 shows the results for aforementioned metrics evaluated on the given algorithms with a tolerance of 8%. We can notice that the heuristic (this problem would hold true for other heuristics aswell) used to obtain a second candidate for the scores *one-correct, both-correct* and *P-Score* leads to biased results. For example, one-correct tends to degenerate to ACC_2, as dividing results on both sides (estimate and reference) by a factor of 2 ends up working very similar to the octave error compensation described for ACC_2. A degeneration of both-correct to ACC_1 is also noticeable. This is why **the perceptual scores P-Score, one-correct and both-correct are disregarded in further reports and discussion**.

[4] https://tempoeval.github.io/tempo_eval/

[5] https://craffel.github.io/mir_eval/

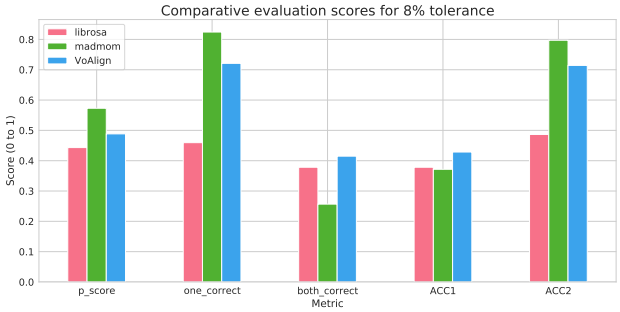

Figure 1. Score metrics with 8% tolerance. Algorithms: **Red:** librosa tempo, **Green:** madmom TempoDetect, **Blue:** VoAlign. Metrics from left to right: P-Score, One-Correct, Two-Correct, ACC1, ACC2

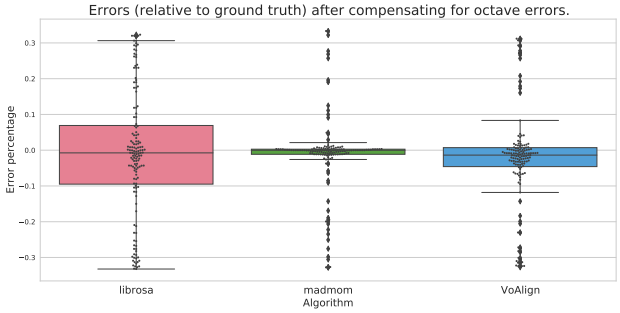

Figure 2. Percentage of error in estimated tempo versus ground truth after compensating for potential octave errors. Algorithms: **Red:** librosa tempo, **Green:** madmom TempoDetect, **Blue:** VoAlign.

We notice that all three algorithms would produce poor results in estimating the exact tempo as postulated by ACC_1. VoAlign would be the only algorithm surpassing a score of 40%. ACC_2 logically yields much butter results for all three algorithms. All algorithms score over 40% in this metric, VoAlign and madmom scoring above 70% with madmom scoring about 79%.

Figure 2 shows the distribution of errors with compensation for octave errors. With this property, the error distribution shown here can be related to ACC_2 rather than ACC_1. Here, the estimates of madmom show to be the most precise while the estimates of librosa are the least precise.

Figure 3, where the described score evaluations are computed with an error tolerance of only 4%, shows a vast score decline in both ACC_1 and ACC_2 for VoAlign and librosa as compared to 8% tolerance (see **Figure 1**). As can be seen also in *Figure 2*, madmom has the best precision of all tested algorithms and such both referred scores for this algorithm stay relatively unaffected by the tolerance decrease.

Regarding the evaluation time, the algorithm featured in librosa was the fastest in computing tempo estimations for all 148 vocal tracks in about 16 minutes, both madmom and VoAlign took significantly longer with about $2\frac{1}{1}$ hours each. This is of course dependent on the technical specifications of the machine used for computation.

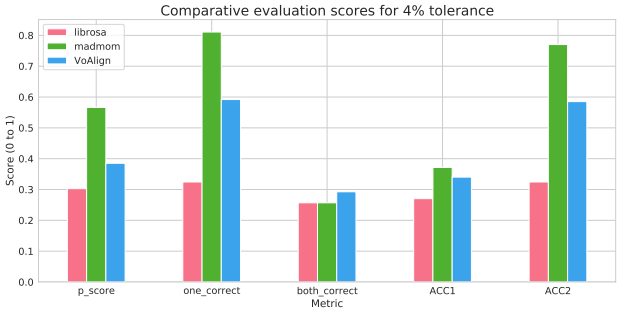

Figure 3. Score metrics with 4% tolerance. Algorithms: **Red:** librosa tempo, **Green:** madmom TempoDetect, **Blue:** VoAlign. Metrics from left to right: P-Score, One-Correct, Two-Correct, ACC1, ACC2

4. DISCUSSION

We can see in **Figure 2** and **Figure 3** that the algorithm of madmom shows to produce the most accurate tempo estimations. Referring to the application of automatic accompaniment generation, this is especially important when it is not an option to align the given acapella vocal track to the estimated tempo. In that case, **Figure 3** gives the most indicative results as to which algorithm would be the best option. However, if time-alignment of the vocal track is possible, the restrictions on tempo precision can be relaxed, so we can see that both madmom and VoAlign would produce good results for this scenario as depicted in **Figure 1**. Additionally, VoAlign seems to have capabilities for better precision on a global estimate, as in the current setup it does just use an average of the temporal tempo evolution as global estimate, such potentially being more precise on actual temporal tempo estimation.

For an automatic accompaniment use-case, especially the metric ACC_2 shows to be expressive as in this case the priority would be a musically meaningful and matching tempo estimation rather than a perfect estimation (ACC_1).

Regarding the computation times mentioned in **Section 3**, the librosa algorithm would on the first look – and despite its lack in accuracy – be the most suitable for real-time scenarios, as it would take 13 seconds on average to compute tempo estimations for each file of the dataset (musdb last 4 minutes on average as mentioned before). But also madmom and VoAlign take about 1 minute on average for their computation on a song, such fulfilling real-time requirements in being computationally faster than the duration of the given sound file. Especially for VoAlign, this kind of calculation cannot be applied with realistic information on its tempo estimation speed as VoAlign does additionally compute tempo- and pitch alignment information in the same process as tempo estimation.

5. CONCLUSION

Three tempo estimation systems have been compared on vocal sound files. The algorithm from madmom has shown to be the most precise, while VoAlign seems to produce the most accurate tempo estimations when precision is not a

must. When octave errors are acceptable for the use case, madmom shows the best performance. The algorithm in librosa is outperformed by both madmom and VoAlign in both scenarios.

6. REFERENCES

[1] S. Böck, F. Krebs, and G. Widmer. Accurate tempo estimation based on recurrent neural networks and resonating comb filters. In *ISMIR*, 2015.

[2] Sebastian Böck, Filip Korzeniowski, Jan Schlüter, Florian Krebs, and Gerhard Widmer. madmom: a new Python Audio and Music Signal Processing Library. In *Proceedings of the 24th ACM International Conference on Multimedia*, pages 1174–1178, Amsterdam, The Netherlands, 10 2016.

[3] Dmitry Bogdanov, Nicolas Wack, Emilia Gómez, Sankalp Gulati, Perfecto Herrera, O. Mayor, Gerard Roma, Justin Salamon, J. R. Zapata, and Xavier Serra. Essentia: an audio analysis library for music information retrieval. In *International Society for Music Information Retrieval Conference (ISMIR'13)*, pages 493–498, Curitiba, Brazil, 04/11/2013 2013.

[4] Sebastian Böck and Markus Schedl. Enhanced beat tracking with context-aware neural networks. In *in Proceedings of the 14th International Conference on Digital Audio Effects (DAFx-11*, 2011.

[5] Peter Grosche and Meinard Müller. A mid-level representation for capturing dominant tempo and pulse information in music recordings. *Proceedings of th 10th International Society for Music Information Retrieval Conference, 189-194 (2009)*, 01 2009.

[6] Peter Grosche, Meinard Müller, and Frank Kurth. Cyclic tempogram—a mid-level tempo representation for musicsignals. pages 5522 – 5525, 04 2010.

[7] Brian McFee, Colin Raffel, Dawen Liang, Daniel PW Ellis, Matt McVicar, Eric Battenberg, and Oriol Nieto. librosa: Audio and music signal analysis in python. In *Proceedings of the 14th python in science conference*, volume 8, 2015.

[8] Martin McKinney and Dirk Moelants. Deviations from the resonance theory of tempo induction. 01 2004.

[9] Martin McKinney and Dirk Moelants. Ambiguity in tempo perception: What draws listeners to different metrical levels? *Music Perception*, 24:155–166, 12 2006.

[10] D. Morris, S. Basu, and I. Simon. Automatic accompaniment for vocal melodies, U.S. Patent 7 705 231 B2, 2010.

[11] G. Percival and G. Tzanetakis. Streamlined tempo estimation based on autocorrelation and cross-correlation with pulses. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 22(12):1765–1776, 2014.

[12] Graham Percival and George Tzanetakis. Streamlined tempo estimation based on autocorrelation and cross-correlation with pulses. *Audio, Speech, and Language Processing, IEEE/ACM Transactions on*, 22:1765–1776, 12 2014.

[13] Zafar Rafii, Antoine Liutkus, Fabian-Robert Stöter, Stylianos Ioannis Mimilakis, and Rachel Bittner. The MUSDB18 corpus for music separation, December 2017.

[14] Eric D. Scheirer. Tempo and beat analysis of acoustic musical signals. *The Journal of the Acoustical Society of America*, 103(1):588–601, 1998.

[15] J. Schlüter and S. Böck. Improved musical onset detection with convolutional neural networks. In *2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 6979–6983, 2014.

[16] Hendrik Schreiber, Julián Urbano, and Meinard Müller. Music tempo estimation: Are we done yet? *Transactions of the International Society for Music Information Retrieval*, 3:111, 08 2020.

[17] Fabian-Robert Stöter, Antoine Liutkus, and Nobutaka Ito. The 2018 signal separation evaluation campaign, 2018.