The Journal of Open Source Software

# onsetsync: An R Package for Onset Synchrony Analysis

**Tuomas Eerola**[1] **and Martin Clayton**[1]

**1** Department of Music, Durham University

**DOI:**

**Software**

- Review
- Repository
- Archive

**Submitted:**
**Published:**



## Summary

Music performance relies on tight yet flexible timing coordination between performers. An important aspect of this interpersonal entrainment is the synchronization of musical events between performers, and this can be analysed from the note onsets obtained from recorded performances. Analysis of the asynchronies between onsets gives an insight into synchronization between performers, and the extent to which it is accurate (how close together the events are in time) and precise (how stable the relationship is). The synchrony between performers is influenced by various factors such as the genre of music, performer skill level, intention, and phrase and beat structures of the music. The analysis of synchrony in music benefits from shared tools as there are a number of common operations that need to be carried out in every dataset (e.g., calculating pairwise synchrony, or assessing the synchrony across other variables such as tempo, metrical hierarchy, or phrasing) and the number of datasets containing onset timing information has recently increased.

## Statement of need

`onsetsync` is a R package for assessment of musical synchrony through note onsets. Data is ingested in the form of CSV files with individual instrument onset times aligned with labelled beat positions; separate metre annotations comprising lists of time estimates for the downbeats are also used for some functions. The package includes functions for common operations such as adding isochronous beats based on metrical structure, adding annotations, calculating classic measures of synchrony between two performers, assessing the periodicity of the onsets, and visualising synchrony across metrical cycles, time, or another property. These functions make the analysis of onset corpora transparent and will allow more scholars to carry out investigations of entrainment in music.

`onsetsync` package comes with example performances from two traditions, Cuban Son and Salsa performances (Poole et al., 2022) and North Hindustani performances (Clayton, Leante, et al., 2021). The examples are taken from the Interpersonal Entrainment in Music Performance (IEMP) project collection that has compiled small ensemble performance data from six traditions (North Indian ragas, Malian jembe, Tunisian stambeli, Uruguayan candombe, European string quartet, and Cuban son and salsa). These collections contain tens of hours annotated and verified onsets and are available from Open Science Framework (OSF). There are also noteworthy datasets containing audio and annotated contents of music relevant to assessing synchrony such as *Carnatic Music Rhythm Dataset* (Srinivasamurthy et al., 2015), *Hindustani Music Rhythm Dataset* (Srinivasamurthy et al., 2016), *Arab-Andalusian (Maghreb) dataset* (Sordo et al., 2014), *Scandinavian Fiddle Tunes with Accompanying Foot-Tapping* (Lordelo et al., 2020) and *MAESTRO (MIDI and Audio Edited for Synchronous Tracks and Organisation)* dataset which is an extensive collection (200+ hours) of professional piano performances (Hawthorne et al., 2018). Some datasets such as Chopin performances compiled by Goebl et al. (2010) facilitate study of the synchrony between the two hands of the pianist.

### Terminology and measures of synchrony

Much of the research on synchrony in music has focused on *sensorimotor synchronization* (SMS) or just synchronization, a process occurring within 100-2000 ms timescales, which has commonly been studied with tapping experiments (Repp & Keller, 2008; Repp & Su, 2013) but also through analyses of music corpora, from string quartets (Wing et al., 2014) to drum ensembles (Polak et al., 2016), and to Indian instrumental music performances (Clayton et al., 2019).

A number of different measures of synchrony have been proposed. We follow the definitions from Clayton et al. (2020) who adopt and develop the measures defined by Rasch (1979, 1988):

Here we first define the onset time differences (or asynchronies), $d$, where $I_{1,i}$ and $I_{2,i}$ refer to an onset at time point $i$ for instrument 1 and 2:

$$d_i = I_{1,i} - I_{2,i} \tag{1}$$

**Precision measures** estimate how consistent the two parts are:

*Pairwise asynchronization*: standard deviation of the asynchronies of any pair of instruments, which is

$$SD = \sqrt{\frac{\sum_{i=1}(d_i - \bar{d})^2}{n-1}} \tag{2}$$

*Groupwise asynchronization*: Root mean square (RMS) of the pairwise asynchronizations calculated as

$$RMS = \sqrt{\Sigma_{i=1}^{n}\left(\frac{d_i}{n}\right)^2} \tag{3}$$

where $n$ is the number of onset time differences.

*Mean absolute asynchrony*: Mean of all unsigned asynchrony values which is

$$|\bar{s}| = \frac{1}{n}\sum_{i=1}^{n}|d_i| \tag{4}$$

**Accuracy measures** estimate how far one part is ahead of or behind the other(s):

*Mean pairwise asynchrony*: mean difference in the onset values of two instruments, $\bar{d}$, i.e., the signed asynchrony values.

*Mean relative asynchrony*: mean position of an instrument's onsets relative to average position ($\bar{I}$) of the group, where the average position is

$$\bar{I} = \frac{I_1 + I_2 + \ldots + I_n}{n} \tag{5}$$

and the relative onset time $R$ is

$$R_i = I_i - \bar{I} \tag{6}$$

and therefore mean relative asynchrony is

$$\bar{s} = \frac{1}{n}\sum_{i=1}^{n} R_i \tag{7}$$

These methods are based on asynchronies between onsets occurring at the same beat positions; other methods exist for assessing synchrony such as utilising circular statistics (Berens, 2009), recurrence quantification analysis (Wallot & Leonardi, 2018), or calculating the synchrony of spike trains (Kreuz et al., 2007).

## Availability and functionality

`onsetsync` is available at Github and can be loaded and installed using the code below. The current version is 0.4.8.

```
library(devtools)
devtools::install_github("tuomaseerola/onsetsync")
library(onsetsync)
```

To carry out such comparisons, `onsetsync` has several auxiliary functions. There are functions that help assessing relative timings, to associate onsets with beat subdivisions and cycles in music structure, and to select and compare the instruments. The library has seven categories of functions summarised in Table 1.

**Table 1:** Function categories and example functions or datasets.

| Category | Examples |
|---|---|
| Input/Output | `get_OSF_csv`, `synthesise_onsets` |
| Annotation | `add_annotation`, `add_isobeats` |
| Visualise | `plot_by_beat`, `plot_by_pair`, `plot_by_variable` |
| Synchrony | `sync_sample_paired`, `sync_execute_pairs` |
| Periodicity | `periodicity`, `period_to_BPM`, `period_nPVI` |
| Summarise | `summarise_onsets`, `summarise_sync` |
| Datasets | `Asere_OU_2`, `DebBh_Drut`, `CSS_IEMP` |

In the next sections we will demonstrate the analysis processes related to asynchrony between several performers. It is also worth pointing out that the library is not dedicated to extraction of onsets from audio as that can be done in other packages (e.g. Librosa, Essentia, MIR toolbox for Matlab, or Sonic Visualiser using well-known onset detection algorithms). Here we take it as granted that the onsets have been extracted already, someone has carried out a quality check, assigned relevant onsets to beat positions and saved them in csv files. For solid overviews on computational onset extraction from audio, see Schlüter & Böck (2014) or Böck et al. (2016), and for a full workflow of how to combine onset detection and annotation of the musical information, see Clayton, Tarsitani, et al. (2021).

### Onset representation

Here we will take one Cuban Son performance that has been included in the library. This song, *Palo Santo*, has been performed by seven performers using the following combination of instruments: bass, guitar, tres, and trumpet, and five percussion instruments, clave, bongo, bell, cajon, and conga (the instrumentation varies slightly between songs and sections within songs). The full data including other Cuban salsa and son performances is available from an open access repository at *Open Science Framework* (OSF) and has been annotated and processed as part of the Interpersonal Entrainment in Music Performance project, available at https://osf.io/sfxa2/.

The code example loads the onset data and annotations of metre of the son performance. The library comes with a set of five Cuban salsa and son (CSS) performances and this is the second example from the IEMP `CSS_IEMP` mini-corpus. The second line of the code selects the nine columns of interest for the analysis; the first two columns are meta-data, followed by four instruments, and three timing related columns are included at the end.

```
CSS_Song2 <- dplyr::select(onsetsync::CSS_IEMP[[2]],
  Piece, Section, Clave, Bass, Guitar, Tres,
  SD, Cycle, Isochronous.SD.Time)
print(knitr::kable(head(CSS_Song2),digits = 2,
  caption = 'Onset data structure example.'))
```

**Table 2:** Onset data structure example.

| Piece | Section | Clave | Bass | Guitar | Tres | SD | Cycle | Isochronous.SD.Time |
|---|---|---|---|---|---|---|---|---|
| Song_2 | Son | NA | NA | NA | NA | 1 | 1 | 5.04 |
| Song_2 | Son | NA | NA | 5.28 | NA | 2 | 1 | 5.26 |
| Song_2 | Son | NA | NA | 5.48 | NA | 3 | 1 | 5.48 |
| Song_2 | Son | NA | 5.71 | 5.71 | 5.73 | 4 | 1 | 5.71 |
| Song_2 | Son | NA | 5.93 | 5.94 | 5.92 | 5 | 1 | 5.93 |
| Song_2 | Son | NA | NA | 6.15 | 6.14 | 6 | 1 | 6.15 |

Table 2 shows a portion of the onset data structure, just 6 rows out of 1568 rows in the data containing 3286 onsets. The first two columns are *meta-data*, referring to the piece and section. The next four columns represent the onset times in seconds (rounded to two decimals for convenience) of four selected instruments, `Clave`, `Bass`, `Guitar` and `Tres`. The onset times for the instruments are available in seconds (with precision of 6 digits, although we prefer to use milliseconds (ms) in the analyses of synchrony to avoid reporting unnecessarily many digits). The last three columns refer to information about the relative time: `SD` represents the beat subdivision within the cycle. In this music there are 16 subdivisions per cycle (which can be felt as 4 beats $\times$ 4 subdivisions). `Cycle` refers to a running number of the beat cycles (each will have 16 beats in this music). `Isochronous.SD.Time` is an evenly temporally spaced time point for each beat subdivision (calculated by dividing each cycle duration by 16) that can act as temporal reference grid. The beat subdivisions and cycles are manually annotated and can be integrated to the data frame with a specific function (`add_annotation`) if given separately. Also a reference timing (`Isochronous.SD.Time`) has been provided for the cycles but it can also be estimated from the cycles and beat subdivisions using a function (`add_isobeats`) in the library. Information about cycles and beat subdivisions is generally based on manual annotation, and originates separately from the automatically extracted onsets; the two are combined to produce tables with onsets assigned to beat positions. This explains why the first row of the table has no onsets but it contains annotation information (Cycle 1 and beat sub-division 1). It is also possible to read the csv files directly from OSF using `get_OSF_csv` function to allow transparent and reproducible analysis workflows with published datasets.

### Onset summaries

Let's explore the overall structure of the onsets in the example piece using the function `plot_by_beat` to visualise the asynchrony relative to an equal division subdivision of the beats for each instrument across the time. This visualisation gives a summary about when the instruments are playing relative to the beat division and the sections of the song, shown in Figure 1.

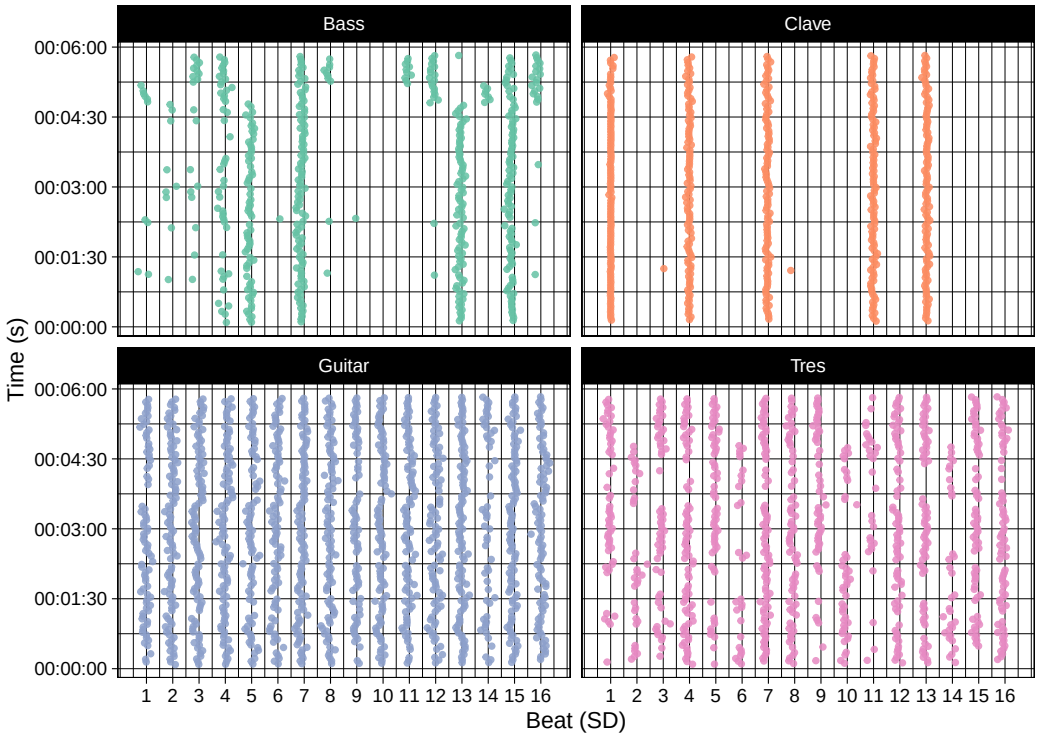


**Figure 1:** Onsets arranged for beat sub-divisions for four instruments across the whole piece.

```
library(dplyr)
fig1 <- plot_by_beat(df = CSS_Song2,
                     instr = c('Bass','Clave','Guitar','Tres'),
                     beat = 'SD',
                     virtual='Isochronous.SD.Time',
                     pcols=2)
print(fig1)
```

At first glance of Figure 1, it is easy to spot the distinctive clave pattern going through the song (the beginning of the song is at the bottom of the graphs, and every cycle is cascaded on top of the previous cycle). Notice how the bass plays mainly on subdivisions 5, 7, 13, 15, except in the last minute when the pattern changes into playing subdivisions 3, 4, 7, 8, 15, 16. The guitar plays almost on every beat subdivision, and towards the end of the piece the tres – a cuban guitar with 6-strings tuned to 3 pitches – plays a pattern of 3 subdivisions and a break.

## Analysis of synchrony

Moving on to the analysis of synchrony, we can first explore how much the onsets deviate from the isochronous beats or from the mean onset times. Figure 2 shows an example of the former for two guitars, guitar and tres.

```
print(plot_by_beat(df = CSS_Song2,
  instr = c("Guitar","Tres"),
  beat = "SD",
  virtual = "Isochronous.SD.Time",
  griddeviations = TRUE))
```

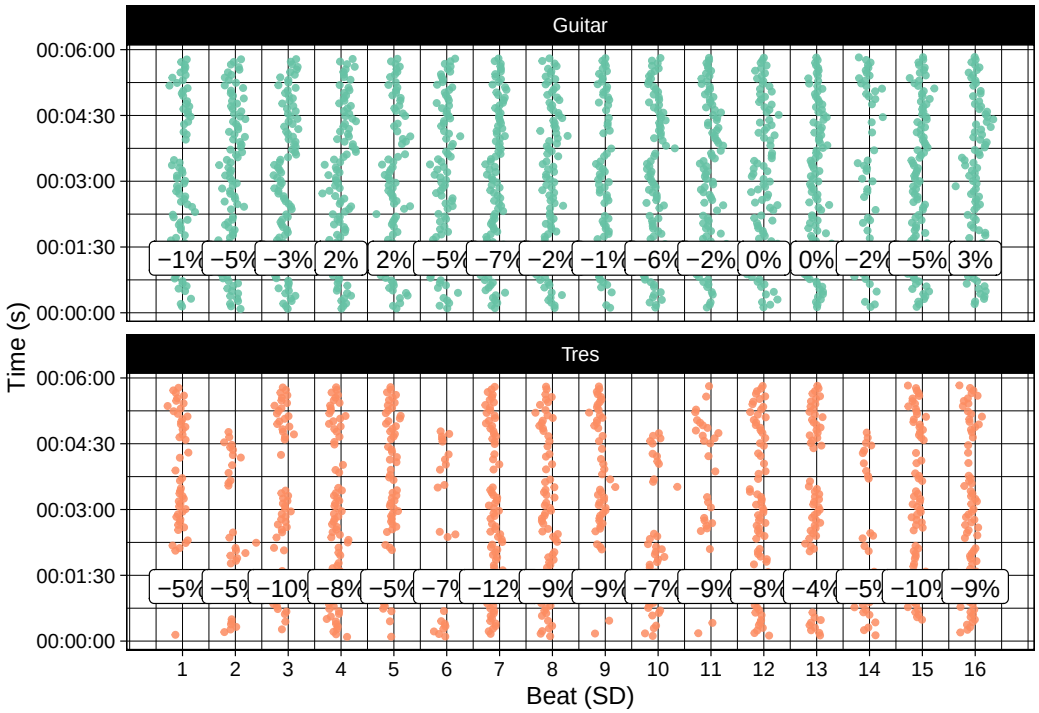


**Figure 2:** Relative timing deviations from beat sub-divisions across the performance for guitar and tres.

Overall this suggests that these two instruments tend to play earlier when compared to the isochronous beat division of the cycle, although the guitar is tightly aligned (<2%) with the reference beat on every fourth beat subdivisions (subdivisions of 1, 5, 9, and 13), which are also the salient metrical positions. Similar fluctuations across the beat subdivisions are not evident in the tres, but the graph reveals that the tres tends to play a little earlier than the guitar.

## Synchrony between the instruments

Let's determine the overall synchrony between specific instruments. Here we continue the analysis of the guitar and the tres and calculate the asynchrony between them.

```
d1 <- sync_sample_paired(CSS_Song2, instr1 = "Guitar", instr2 = "Tres",
  beat = "SD")
dplyr::summarise(data.frame(d1),
  N = n(), Mean.ms = mean(asynch*1000), Sd.ms = sd(asynch*1000))
```

```
##     N  Mean.ms    Sd.ms
## 1 853 12.53126 26.74134
```

This analysis indicates that on average, the tres plays 13 ms ahead of the guitar, with a standard deviation of 27 ms. When one is comparing instruments that have radically different number of joint onsets (such as the clave and the bass in this example), it is possible to make the comparisons between the pairs of instruments easier by specifying a sample of onsets that will be taken from each instrument for the analysis. It also possible to establish the confidence intervals by bootstrapping the asynchrony calculations.

To carry out the comparison for all possible pairings of the instruments is possible with `sync_execute_pairs` function and the results can be visualised with a related function (`plot_by_pair`).

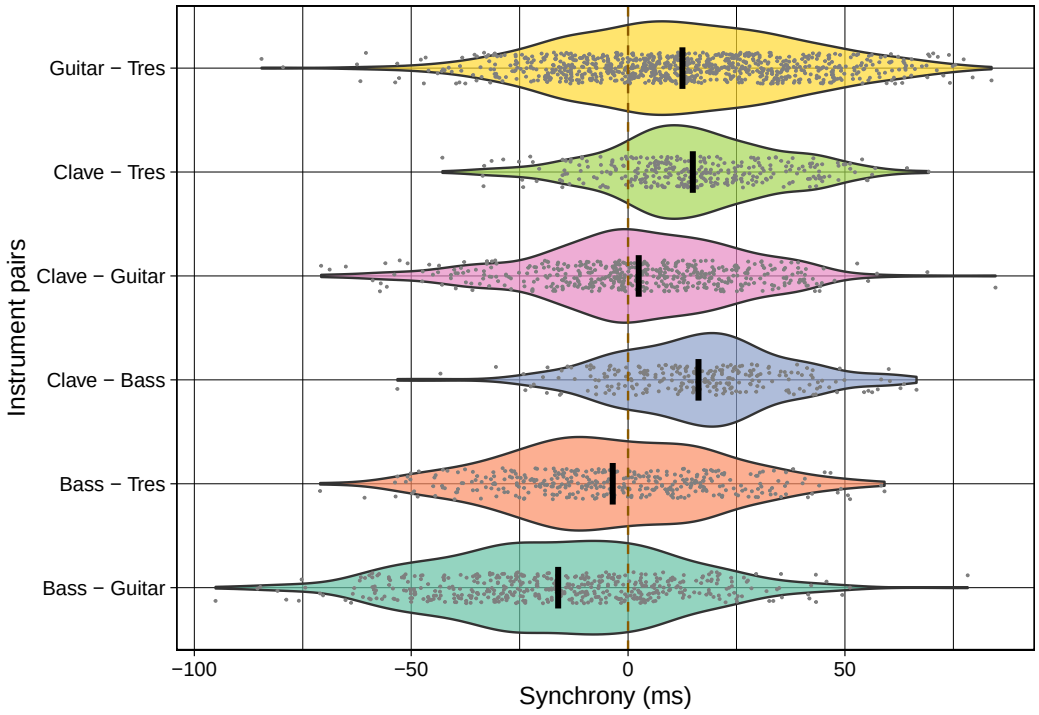


**Figure 3:** Asynchronies across multiple instrument pairs.

```
inst <- c("Clave","Bass","Guitar","Tres")
dn <- sync_execute_pairs(CSS_Song2, inst, beat = "SD")
print(plot_by_pair(dn))
```

In Figure 3 we see how different pairs of instruments have different asynchrony relationships; the bass is consistently ahead of the guitar, and the clave and the guitar play behind the tres, to pick some of the extreme examples from the visualisation. The vertical lines stand for median asynchrony and the violin plot shows the distribution of onset time differences.

We can calculate the classic measures of synchronization once the pairing has been done. Here are these measures calculated for the synchrony between the bass and the clave.

```
d <- sync_sample_paired(CSS_Song2,"Clave","Bass", beat = "SD")
print(t(summarise_sync(d)))
```

```
##                                [,1]
## Pairwise asynchronization 19.58636
## Mean absolute asynchrony  20.66440
## Mean pairwise asynchrony  16.23288
```

The output suggests that at least in this piece, these two instruments tend to have the synchronization precision around 20 ms, depending on the method of calculation. Pairwise asynchronization is the standard deviation of the signed asynchrony; this tends to be highly correlated with the mean absolute asynchrony. The Mean pairwise asynchrony measure shows that on average, the clave plays 16 ms after the bass.

We can also calculate the relative asynchrony by comparing the onset times to the mean of the other instruments that we define. Here we compare the asynchrony of the bass and the clave to the mean of the rhythm section as defined by the mean onset times of the four instruments.

```
dl <- sync_sample_paired_relative(df = CSS_Song2,
  instr = "Bass", instr_ref = c("Guitar","Bass","Tres","Clave"),
  beat = "SD")
print(dl$`Mean pairwise asynchrony`)
```

```
## [1] -18.72675
```

```
dl <- sync_sample_paired_relative(df = CSS_Song2,
  instr = "Clave", instr_ref = c("Guitar","Bass","Tres","Clave"),
  beat = "SD")
print(dl$`Mean pairwise asynchrony`)
```

```
## [1] 0.8796104
```

The relative measure (mean relative asynchrony) shows that the bass is ahead (i.e. -19 ms) of the mean of other the instruments whereas the clave is almost exactly in synchrony with the other instruments (<1 ms late). In different types of performances, musical pieces, instruments, and genres, the magnitude of these differences is highly variable (Clayton et al., 2020).

It is also possible to calculate synchrony between instruments across beat sub-divisions, assess synchrony across multiple performances, and estimate and visualise synchrony with other variables. The library also comes with functionality to analyse the periodicity of the onsets and synthesising the onsets.

## Conclusions

`onsetsync` provides representations, visualisations and calculations of key measures relating to onset structures in music. The library does not take a stance on how the onset times have been estimated, but assumes that onset times are mapped onto beat positions, and that metre annotations are available. Future needs for development lie in creating a specific corpus format that would be open to many uses, to test the framework with other datasets, and to incorporate other analytical options such as circular statistics, Granger causality or cross-recurrence analyses.

We hope scholars and students of music, music information retrieval and psychology can pursue topics related to synchrony in music with a transparent workflow assisted by the library and capitalise on existing datasets that are compatible with the library. Eventually we hope that tools such as these will allow a larger number of people to create and utilise open datasets related to musical behaviours, helping the topic become data-driven, empirical and collaborative.

## Acknowledgements

We acknowledge contributions from IEMP Fellows, specifically Adrian Poole, who created Cuban Son and Salsa dataset with Simone Tarsitani and Martin Clayton, but also Rainer Polak, Richard Jankowsky, Nori Jacoby, Mark Doffman, Luis Jure, Andy McGuiness, Nikki Moran, and Martín Rocamora). We are thankful for technical support by Simone Tarsitani and useful feedback by Juliano Abramovay. For funding and time, we acknowledge EU Horizon 2020 FET project EnTimeMent - ENtrainment & synchronization at multiple TIME scales in the MENTal foundations of expressive gesture that was instrumental in turning the idea about the shared resource for temporal analyses such as musical onsets into an open access library.

The package is available from https://tuomaseerola.github.io/onsetsync/. The analyses shown in the article are contained in the vignette and article titled “Analysis Example”.

## References


Berens, P. (2009). CircStat: A MATLAB toolbox for circular statistics. *Journal of Statistical Software*, *31*, 1–21.

Böck, S., Korzeniowski, F., Schlüter, J., Krebs, F., & Widmer, G. (2016). Madmom: A new python audio and music signal processing library. *Proceedings of the 24th ACM International Conference on Multimedia*, 1174–1178.

Clayton, M., Jakubowski, K., & Eerola, T. (2019). Interpersonal entrainment in Indian instrumental music performance: Synchronization and movement coordination relate to tempo, dynamics, metrical and cadential structure. *Musicae Scientiae*, *23*, 304–331. https://doi.org/10.1177/1029864919844809

Clayton, M., Jakubowski, K., Eerola, T., Keller, P., Camurri, A., Volpe, G., & Alborno, P. (2020). Interpersonal entrainment in music performance: Theory, method and model. *Music Perception*, *38*(2), 136–194.

Clayton, M., Leante, L., & Tarsitani, S. (2021). *IEMP North Indian Raga*. OSF. https://doi.org/10.17605/OSF.IO/KS325

Clayton, M., Tarsitani, S., Jankowsky, R., Jure, L., Leante, L., Polak, R., Poole, A., Rocamora, M., Alborno, P., Camurri, A., Eerola, T., Jacoby, N., & Jakubowski, K. (2021). The interpersonal entrainment in music performance data collection. *Empirical Musicology Review*, *16*(1), 65–84. https://doi.org/10.18061/emr.v16i1.7555

Goebl, W., Flossmann, S., & Widmer, G. (2010). Investigations of between-hand synchronization in Magaloff's Chopin. *Computer Music Journal*, *34*(3), 35–44.

Hawthorne, C., Stasyuk, A., Roberts, A., Simon, I., Huang, C.-Z. A., Dieleman, S., Elsen, E., Engel, J., & Eck, D. (2018). Enabling factorized piano music modeling and generation with the MAESTRO dataset. *arXiv Preprint arXiv:1810.12247*.

Kreuz, T., Haas, J. S., Morelli, A., Abarbanel, H. D., & Politi, A. (2007). Measuring spike train synchrony. *Journal of Neuroscience Methods*, *165*(1), 151–161.

Lordelo, C., Benetos, E., Dixon, S., Ahlbäck, S., & Ohlsson, P. (2020). Adversarial unsupervised domain adaptation for harmonic-percussive source separation. *IEEE Signal Processing Letters*, *28*, 81–85.

Polak, R., London, J., & Jacoby, N. (2016). Both isochronous and non-isochronous metrical subdivision afford precise and stable ensemble entrainment: A corpus study of malian jembe drumming. *Frontiers in Neuroscience*, *10*, 285.

Poole, A., Tarsitani, S., & Clayton, M. (2022). *IEMP Cuban Son and Salsa*. OSF. https://doi.org/10.17605/OSF.IO/SFXA2

Rasch, R. A. (1979). Synchronization in performed ensemble music. *Acta Acustica United with Acustica*, *43*(2), 121–131.

Rasch, R. A. (1988). Timing and synchronization in ensemble performance. In J. A. Sloboda (Ed.), *Generative processes in music: The psychology of performance, improvisation, and composition* (pp. 70–90). Clarendon Press/Oxford University Press.

Repp, B. H., & Keller, P. E. (2008). Sensorimotor synchronization with adaptively timed sequences. *Human Movement Science*, *27*(3), 423–456.

Repp, B. H., & Su, Y.-H. (2013). Sensorimotor synchronization: A review of recent research (2006–2012). *Psychonomic Bulletin & Review*, *20*(3), 403–452.

Schlüter, J., & Böck, S. (2014). Improved musical onset detection with convolutional neural networks. *2014 IEEE International Conference on Acoustics, Speech and Signal Processing (Icassp)*, 6979–6983.

Sordo, M., Chaachoo, A., & Serra, X. (2014). Creating corpora for computational research in Arab-Andalusian music. *Proceedings of the 1st International Workshop on Digital Libraries for Musicology*, 1–3.

Srinivasamurthy, A., Holzapfel, A., Cemgil, A. T., & Serra, X. (2015). Particle filters for efficient meter tracking with dynamic Bayesian networks. In M. M. & W. F. (Eds.), *Proceedings of the 16th international society for music information retrieval conference; 2015 oct 26-30*. International Society for Music Information Retrieval (ISMIR).

Srinivasamurthy, A., Holzapfel, A., Cemgil, A. T., & Serra, X. (2016). A generalized

Bayesian model for tracking long metrical cycles in acoustic music signals. *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 76–80.

Wallot, S., & Leonardi, G. (2018). Analyzing multivariate dynamics using cross-recurrence quantification analysis (CRQA), diagonal-cross-recurrence profiles (dcrp), and multidimensional recurrence quantification analysis (MDRQA)– A tutorial in R. *Frontiers in Psychology*, *9*, 2232.

Wing, A. M., Endo, S., Bradbury, A., & Vorberg, D. (2014). Optimal feedback correction in string quartet synchronization. *Journal of The Royal Society Interface*, *11*(93), 20131125.