# Can I Have the Keys?: Key Validation Using a MIDI Database

Joshua Albrecht,[*1] Daniel Shanahan[#2]

[*]*Music Department, The University of Mary Hardin-Baylor, Belton, TX, USA*
[#]*School of Music, Louisiana State University, Baton Rouge, LA, USA*
[1]jalbrecht@umhb.edu, [2]dshanahan@lsu.com

## ABSTRACT

The accuracy rates of a recently-developed key finding algorithm were estimated to be quite high, higher than most other currently available algorithms. This study provides a) an external validation of the accuracy rates of that algorithm on an independent dataset, and b) a validation of that dataset using the algorithm. Specifically, a large dataset of over 12,000 MIDI works found on ClassicalArchives.com were sampled for this purpose. MIDI data tends to contain a high percentage of errors. To determine whether the database provides a sufficient signal to noise ratio for pitch-reliant tasks like key-finding, an initial validation study was performed on the dataset in which accuracy rates for the key-finding algorithm on a small subset of 32 works were compared with a ground truth of accuracy rates. In the initial study, it was found that 28 of 32 (87.5%) of MIDI works returned the same result as the ground truth. Given this relatively high accuracy rate, it was determined that the database can be an appropriate vehicle for testing a key finding algorithm. On a dataset of 192 separate works, it was found that the algorithm correctly assigned keys to 164 works (85.4%). Together, the two studies are consistent with the ideas that, 1) though there will be some errors as a result of noisy data, a MIDI database can serve some uses for some studies, and 2) the key finding meta-algorithm provides an accurate way to assign keys to this database.

## I. INTRODUCTION

MIDI data can be notoriously messy. In an unpublished validation study, Shanahan and Albrecht (MS) found that ClassicalArchives.com, consisting of a large MIDI database of over 12,000 user-generated works, was exceptionally prone to errors. They measured fifteen different types of error, including pitch, rhythms, meter, and key signature error. Depending on the type of error, error rates ranged from a low of 5.18% (± ~3%) for misaligned note onsets to a high of 53.88% (± ~7%) for incorrect number of measures due to blank measures opening files.

Despite the untrustworthy nature of the data, the dataset is nevertheless an attractive potential source of future research. There are already studies beginning to be published that draw on this particular dataset (e.g. White 2013, 2014). One reason this large database is attractive is because most of the current digitally encoded datasets available have been used extensively in many studies for years. The overuse of particular datasets in empirical studies could lead to methodological errors such as double-use data or multiple tests, and this new repository offers a way to avoid using the same data too many times. A second reason for using this dataset is because it is more representative of composers and ensemble sizes than most currently available datasets, which tend to focus on Germanic composers and small ensembles.

Given the advantages offered by adding such a database to the available resources, it is reasonable to further investigate its viability for different types of research questions. For example, even if the error rates are high for note durations, one might ask if reliable estimates of scale degree distributions can still be made.

For the purposes of illustration, consider Figure 1. This figure shows an encoding from the database for the first seven measures of Chopin's Nocture in G minor, Op. 15, No. 3. The top of Figure 1 is from the 1915 Rafael Josephy Schirmer edition. Underneath this excerpt is a visualization of the MIDI data, obtained by converting the MIDI data to MusicXML and opening it in Musescore 2.0.2. Inspection of the two versions reveals sizable differences in metric placement, rhythmic durations, and number of note attacks, along with missing dynamics and articulations.

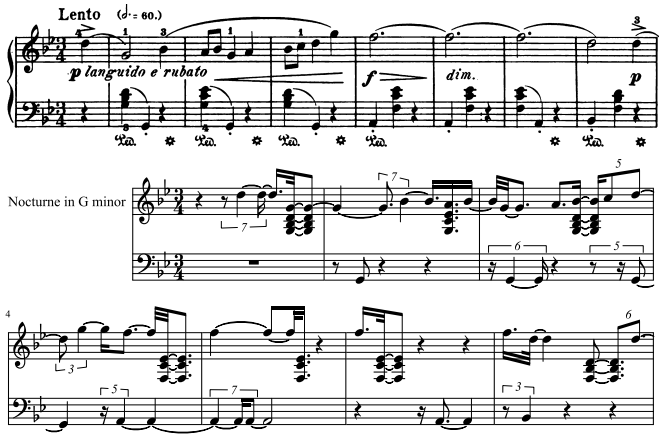


**Figure 1. An illustration of some of the problems of MIDI data. The top image is from a 1915 published score of Chopin Op. 15, No. 3 and the bottom is a visual representation of corresponding MIDI data. As can be seen, there are many discrepancies.**

Another obstacle for the practical use of MIDI data is a tendency for no key interpretation to be provided. That is, pitch-classes are present with a key signature in more than half of the cases, but most files do not explicitly identify the key. Compounding this issue is a tendency for provided key signatures to be incorrect (~40% of the time). While certain questions can be answered without key identifications, any attempts to analyze functional progression, scale degree tendencies, or other musical elements related to key are greatly hindered without this information.

Even considering the gross inaccuracies that tend to plague this data, most of the pitch information is relatively intact. Shanahan and Albrecht (MS) estimate that only about 8.5% (± ~2%) of pitch information in this database is incorrectly encoded. It may be the case, then, that this dataset may be equipped to answer certain types of questions involving pitch.

One such question is the accuracy of key finding algorithms, which is a particularly germane issue for three

primary reasons. First, key finding algorithms only consider distributions of pitch-classes. Although there are some pitch-class encoding issues in the dataset, accuracy rates are relatively high. Moreover, music theorists have historically focused on pitch information more than on other types of musical structures for analysis. Therefore, resolving whether or not the database can reliably answer pitch-related questions promises to offer the most diverse analytical benefits.

Second, as mentioned above, one way to greatly increase the database's usefulness for examining functional or scale-degree information would be to assign keys to each file. By testing to what extent key finding algorithms perform accurately on this dataset, we can move towards using the database more constructively.

Finally, key finding algorithms have undergone some significant changes in recent years. Specifically, refinements and improvements have been made to key finding strategies, resulting in higher accuracy rates. One such algorithm uses Euclidean distance to estimate keys, rather than the more common use of correlation (Albrecht & Shanahan, 2013). In their study, Albrecht and Shanahan obtained an accuracy rate of 93.1% on a database of 982 works encoded in humdrum kern notation.

By combining their algorithm with the Aarden-Essen algorithm (Aarden 2003), they were able to create a meta-algorithm that performed at 95.1% accuracy. However, because the algorithm was determined in a *post hoc* manner, further research on an independent dataset is needed to fully assess the actual accuracy rate. The ClassicalArchives.com dataset provides such an opportunity to test this algorithm's veracity.

## II. AIMS

In light of the above considerations, this study has two complementary aims: 1) we will examine how appropriate the classical archives MIDI dataset is for examining pitch-related inquiries like key profiling and key finding, and 2) we will test the Albrecht & Shanahan (2013) meta-algorithm on this independent dataset to determine how accurate the algorithm is. If the results are consistent with both the idea that the MIDI dataset can be used for certain pitch-related tasks and that the meta-algorithm is as accurate as Albrecht & Shanahan (2013) suggest, then using this new dataset with algorithmically assigned keys can open up new possibilities for computational research.

## III. METHODOLOGY

### A.Sampling

The ClassicalArchives.com dataset is relatively large by the standards of classical music corpora. At over 12,000 files, the database promises to significantly augment current digitalized offerings. However, validation studies tend to be very labor intensive, requiring visual inspection of individual files. As a result, some means of sampling a small subset of these files is necessary.

Recall that there are two primary goals for this study. The first goal is to assess the viability of using this relatively messy data for key finding. The second goal is to test the key-finding algorithm. Different subsets are required for each individual goal.

*1.Database validation sampling*

The first step we used in the process of sampling for these two goals was simply to excerpt roughly 10% of the dataset. We first sampled 1,200 files from the dataset. After the conversion process, there were issues with several of the files (see III: B below). These problematic files were discarded, resulting in 1,125 files.

For the purpose of assessing how viable these data are for a key finding task, some ground truth was required. Therefore, the methodological decision was made to pair files from the 1,125 file subset with the corresponding musical works used in Albrecht and Shanahan (2013). In that study, 982 works had previously had keys assigned by independent analysts. These works had been encoded into the Humdrum kern representation, and the assigned keys were used as a ground truth for the key finding algorithm. By matching files from the current study's subset to the works used in that study, a ground truth would enable the testing of this dataset.

Importantly, note that the goal in the first phase of sampling is not to determine how accurate the key finding algorithm is. Instead, we wish to assess *how similarly the algorithm performs* on these messy MIDI files as it did on that study's cleanly-encoded files. Because we already know how well the algorithm performs on clean data, we can determine whether the algorithm performs in a similar or different way on MIDI data. In short, we will not be comparing the algorithm's results on the MIDI to the ground truth of analyzed keys, but rather to the ground truth of the algorithm's results on kern data. Therefore, it is important that we only use the exact same movements or works used in the first study.

After matching works used in Albrecht and Shanahan (2013) to the subset of 1,125 files selected for this study, only 32 files were found in common between the two datasets. These 32 files were used as a comparison to determine the effect of encoding on algorithm performance. Members of this subset included Bach's *Well-Tempered Clavier* and Brandenburg concerto movements, Beethoven piano sonata movements, Beethoven, Mozart, and Haydn string quartet movements, Scarlatti piano sonatas, Chopin Op. 28 preludes, Hummel Op. 67 *Préludes*, and movements from Vivaldi's *The Four Seasons.*

*2.Key finding algorithm validation sampling*

Depending on the results from the first element of this study, it may be appropriate to validate the key finding algorithm on the MIDI dataset. Of course, this depends on the initial results. If the way that the algorithm performs on the MIDI set of 32 files is significantly different from the way it performs on the Humdrum files, then that reveals that this dataset is likely inaccurate enough to be ill equipped for key finding tasks.

Each file used in the key finding validation needs to have a key manually encoded by an analyst. This is a time consuming process, requiring visual inspection not of the MIDI file, but of an original publication of the composition or movement (in case of encoded errors in the MIDI file). Of course, the bigger the sample, the more confidence can be placed in the results from the study. As a middle-ground between these two values, 204 of the 1,125 files originally subsetted that were not included in the first sample were chosen for validation of the algorithm.

### B.File conversion

The key-finding algorithm does not work directly on MIDI data, so in order for the analysis to be performed the data first needed to be converted into Humdrum's kern representation. However, tools that directly convert MIDI data to kern representations tend to perform poorly.

An alternative strategy is to use a middle-ground. Most computer-based music notation programs directly engage with MIDI data. For this study, we used MuseScore 2.0.2 to read the MIDI data.

When reading data from MIDI into MuseScore, shortest note length must be specified. To capture the most accurate representation of the data, we used the shortest note value of a 64th note. Once the data was read by MuseScore, it could easily be exported into MusicXML. From MusicXML, the conversion into kern format is relatively straightforward using the Humdrum extra tool xlm2hum, designed by Craig Sapp (2010).

Of the 1,200 works originally sampled for the subset, there were errors in at least one step of the file conversion process for 75 files. These 75 files were discarded from further consideration.

### C.Key analysis

For the second stage of this study, files were required to be analyzed with key identifications as a ground truth to compare performance of the key finding algorithm against. Because some of the MIDI files contained a large number of errors, the decision was made to assign keys based on published editions of each work rather than on the MIDI files.

One difficulty in identifying the compositions linked to the files in the ClassicalArchives.com database is that the files do not always contain metadata. However, the website contains a directory of its musical works in html documents. The first step in identifying keys, then, was to identify the works. The html files were searched for the file names in the subset of 1,125 works.

Once the composer and work were identified, scores were downloaded from the International Music Score Library Project (imslp.org). Editions were randomly sampled, with the exception that recent creative commons publications were excluded. The reason why this exclusion criterion was established is because many of these files were generated from online encodings of works. Because our goal was to validate these online encodings against publications, scores generated from the online encodings were problematic.

The 204 works selected for validating the key finding algorithm were randomly assigned to one of the two authors of this study. The pdf documents were then analyzed for keys by one of the two authors. If the work was post-tonal, it was discarded as not appropriate for use with a key finding algorithm. Pre-tonal works were included if the tonal center and mode ('majorish' or 'minorish') was clear, otherwise they were discarded. Some of the MIDI files were also clearly transposed from their originally published versions; for the purpose of clarity, these were also discarded. Finally, works that began and ended in different keys were discarded from further consideration. After discarding these files, 192 works remained.

## IV. RESULTS

### A.Database validation

Recall that the initial goal of this study was to assess whether or not the MIDI data was viable for pitch-based tasks like key finding. In order to determine whether this was the case, we compared the key finding meta-algorithm's results on the 32 kern files used for ground truth to the 32 MIDI works sampled for this purpose, described in III: A, 1.

For example, assume that the algorithm estimates the keys of four works encoded in Humdrum format. For the sake of argument, we will call them compositions A, B, C, and D. If the algorithm correctly estimates the keys of A and B and incorrectly estimates the keys of C and D, the algorithm will have an accuracy rate of 50%. Once we locate MIDI versions of compositions A, B, C, and D, we can compare the algorithm's performance on the MIDI and Humdrum formats. If the same algorithm correctly estimates the keys on the MIDI versions of C and D and incorrectly estimates the keys of A and B, the accuracy will be 50% compared to the correct keys.

However, the performance of the algorithm on the MIDI data compared to the Humdrum data will be 0%. In short, because they are the same compositions, the algorithm should perform in the exact same way on either version of the data *if the representations of the data are equally useful for key estimation in either representation.* Because the algorithm performs differently on each work in this hypothetical example, we can safely conclude that the representation of the data between the two encodings has a large impact on key estimation.

Of the 32 kern files used from Albrecht and Shanahan (2013), 31 works were correctly assigned keys by their meta-algorithm (96.9%). Of the MIDI files of the matched works, only 29 works were correctly assigned keys (90.6%). Moreover, the one kern file incorrectly assigned a key, the first movement from Beethoven's string quartet No. 14, was *correctly* assigned to the corresponding MIDI file. Of the 32 works, then, only 28 were assigned matching keys, or 87.5%.

These results are consistent with the hypothesis that the errors involved in the MIDI data have an effect on certain pitch-based tasks, like key estimation. Nevertheless, 87.5% is not too different from the accuracy rates of many key finding algorithms. Although in this case the same algorithm is being applied to different formats of (allegedly) the same work, and so should be held to a higher standard, the results are nevertheless not out of the realm of normal performance. Therefore, we felt that it was appropriate to continue the validation of the algorithm. Nevertheless, it is important to keep in mind that at least some of the error in the following section may be attributable to error in the dataset rather than error in the key finding meta-algorithm.

### B.Meta-algorithm validation

After it was determined that the MIDI database provided a reasonably satisfactory source for pitch class information, at least for the purposes of estimating keys using pitch-class distributions, the accuracy of the Albrecht-Shanahan meta-algorithm (2013) was tested against a sample of 204 files. As mentioned above, we established ground truth for keys by a visual inspection of the corresponding scores downloaded from the International Music Score Library Project (imslp.org). Works in which the key was not clear or where the MIDI file was clearly transposed from the original source were discarded.

Of the 192 files remaining, the key finding algorithm correctly identified the keys of 164 files, or 85.4%. Four of

these files had the correct pitch-class assigned as a tonal center, but it was enharmonically respelled, such as a work in D♭ minor being identified as C# minor. The reason for these misattributions is likely due to the fact that the key finding algorithm picks an enharmonic key based on which accidentals are more prevalent in the piece. Because MIDI data only includes pitch-class information, MIDI files with no key signature or incorrect key signature (again, about 40% of the time (Shanahan & Albrecht, MS)) are interpreted by MuseScore as the pitch-class that appears more often in the repertoire. For example, B♭ is chosen over A# because it is a more common notated pitch. Because this lack of information is not an error with the algorithm, enharmonically respelled tonal centers were counted as matches.

Of the errors in key estimation, the most common error was assignment of the parallel key, occurring as 46.4% of errors. This result was also prevalent in the initial study (Albrecht & Shanahan, 2013). In this database, this error comprises a bigger percentage of the total. Many of these errors involved Baroque minor mode works. This is not surprising given the tendency of these works to employ a Picardy ending. This particular error, then, appears to be the result of the algorithm's focus on the first and last 8 measures of each work (for further details on how the algorithm works, see Albrecht & Shanahan, 2013). The second and third most common errors involved the relative key and fifth-related, each accounting for 25% of errors. The different types of errors and their frequencies are shown in Table 1.

**Table 1. Types of errors in key-assignment. The most common type of error is assignment of the relative key.**

| Error type | # of occurrences (%) |
|---|---|
| Parallel key | 13 (46.4%) |
| Relative key | 7 (25%) |
| Fifth-related | 7 (25%) |
| Non-relative mediant-related | 1 (3.5%) |
| Step-related | 1 (3.5%) |

## V. DISCUSSION

The results from this test of the algorithm reveal a lower accuracy rating than the initial study. This is not surprising, though, for several reasons. In the first case, the meta-algorithm was developed *post hoc* over several iterations to produce that formulation that would perform as well possible on the given data. It is likely, therefore, that there was some overfitting involved in the initial study. The current study is a direct consequence of the effort to test the extent of the overfitting involved in the first study.

In the second case, the lower accuracy ratings are unsurprising given the nature of the data. Because the MIDI data are fairly error-prone, a number of the mistakes in key assignment may be attributable to noisy data. Given the relatively high rate of errors in pitch-class (~8.5%) and rhythmic duration (~32%), both of which types of data are important in the estimation of key with this algorithm, it is not surprising that accuracy rates would be lower than in clean datasets.

Nevertheless, if a more generous calculation of accuracy is used, in which only the correct assignment of tonal center is checked, then this algorithm performs at a fairly impressive 92.2% accuracy rate, not significantly different from the accuracy rate found in Albrecht & Shanahan (2013).

Given the fluid nature of modality, especially in early common-practice era music and late common-practice era music, the identification of tonal center alone may be all that is possible in these styles. Because the ClassicalArchives.com dataset is broader than the sample used in the original study, it is possible that these early and late works are skewing the results of the study. In other words, it is possible that the accuracy rates are different because the samples represent slightly different populations.

Nevertheless, correct assessment of tonal center is still enough information to assign scale-degrees and harmonic function of chords. Although completely accurate key assignments of both tonal center and mode would be more useful, tonal center alone is enough to begin functional analysis.

These results are consistent with a) the hypothesis that the key-finding meta-algorithm is a reliable indicator of key, and b) that the MIDI database can somewhat reliably be used for the right pitch-related research questions. With the error rate so high in the comparison between the 32 files used in the original study and this study, it is difficult to determine the contribution of encoding error on key estimation error.

Future work will attempt to further disentangle these effects by further validating both the database and the meta-algorithm. Moreover, future research questions examined with this large and exciting new dataset will likely provide further insights into the nature of the data and its reliability.

## REFERENCES


Aarden, B. (2003). *Dynamic melodic expectancy* (Unpublished doctoral dissertation). Ohio State University, Columbus, OH.

Albrecht, J., & Shanahan, D. (2013). The use of large corpora to train a new type of key-finding algorithm: An improved treatment of the minor mode. *Music Perception: An Interdisciplinary Journal, 31*(1), 59-67.

Sapp, C. (2010.). http://extras.humdrum.org/man/xml2hum/. Accessed 5/6/2016.

White, C. (2014). Changing styles, changing corpora, changing tonal melodies. *Music Perception: An Interdisciplinary Journal, 31*(3), 244-253.

White, C. (2013). An alphabet reduction algorithm for chordal n-grams. In J. Yust & J. Wild, *Mathematics and Computation in Music.* Heidelberg: Springer.