A COMPARATIVE EVALUATION OF ALGORITHMS FOR DISCOVERING TRANSLATIONAL PATTERNS IN BAROQUE KEYBOARD WORKS

Tom Collins
The Open University, UK
t.e.collins@open.ac.uk

Jeremy Thurlow
University of Cambridge
jrt26@cam.ac.uk

Robin Laney, Alistair Willis, and Paul H. Garthwaite
The Open University, UK

ABSTRACT

We consider the problem of intra-opus pattern discovery, that is, the task of discovering patterns of a specified type within a piece of music. A music analyst undertook this task for works by Domenico Scarlattti and Johann Sebastian Bach, forming a benchmark of 'target' patterns. The performance of two existing algorithms and one of our own creation, called SIACT, is evaluated by comparison with this benchmark. SIACT out-performs the existing algorithms with regard to recall and, more often than not, precision. It is demonstrated that in all but the most carefully selected excerpts of music, the two existing algorithms can be affected by what is termed the 'problem of isolated membership'. Central to the relative success of SIACT is our intention that it should address this particular problem. The paper contrasts string-based and geometric approaches to pattern discovery, with an introduction to the latter. Suggestions for future work are given.

1. INTRODUCTION

This paper discusses and evaluates algorithms that are intended for the following task: given a piece of music in a semi-symbolic representation, discover so-called translational patterns [14] that occur within the piece. Translational patterns (in the geometric sense) are discussed further in Sec. 1.1. Although they are not the only type of pattern that could matter in music analysis, many music analysts would acknowledge that such a discovery task forms part of the preparation when writing an analytical essay [6]. Even if the final essay pays little or no heed to the discovery of translational patterns, neglecting this preparatory task entirely could result in failing to mention something that is musically very noticeable, or worse, very important. Hence we are motivated by the prospect of automating the discovery task, as it could have interesting implications for music analysts (and music listeners in general), enabling them to engage with pieces in a novel manner. We also consider this task to be an open problem within music information retrieval (MIR), so attempting to improve upon current solutions is another motivating factor. The algorithms are applied to Baroque keyboard pieces by Scarlatti (Sonatas L1 and 10) and—to ensure common ground with existing work [7, 13]—Bach (Preludes BWV849 and 854). Two existing algorithms and one of our own creation are evaluated by comparing their output with a music analyst's (the second author's) independent findings for these same keyboard pieces (Sec. 4).

1.1 Review of existing work

In MIR there do not seem to be clear distinctions between the terms pattern 'discovery' [5, 8, 14, 16], 'extraction' [10, 11, 17], 'identification' [7, 9] , and 'mining' [3], at least in the sense that most of the papers just cited address very similar discovery tasks to that stated at the beginning of Sec. 1. Conklin & Bergeron [5] give the label 'intra-opus' discovery to concentrating on patterns that occur *within* pieces. An alternative is 'inter-opus' discovery, where patterns are discovered *across* many pieces of music [5, 9]. This makes it possible to gauge the typicality of a particular pattern relative to the corpus style. Terms that *are* clearly distinguished in MIR are pattern 'discovery' and 'matching' [4]. Pattern matching is the central process in 'content-based retrieval' [18], where the user *provides* a query and then the algorithm searches a music database for more or less exact instances of the query. The output is ranked by some measure of proximity to the original query. This matching task is quite different from the intra-opus discovery task, where there is neither a query nor a database as such, just a single piece of music, and no obvious way of ranking an algorithm's output. While we have stressed their differences, some authors attempt to address both discovery and matching tasks in the same paper [12, 13], suggesting that representations/algorithms that work well for one task might be adapted and applied fruitfully to the other.

Some attempts at pattern discovery have been made with audio representations of music [15]. However, we, like the majority of work cited in this section, begin with a semi-symbolic representation, such as a MIDI file. Work on semi-symbolic representations can be categorised into string-based [2,3,5,8–11,16,17] and geometric approaches [7, 12–14], and which approach is most appropriate depends on the musical situation. For instance the string-based method is more appropriate for the excerpt in Fig. 1. We propose that the most salient pattern in this short ex-

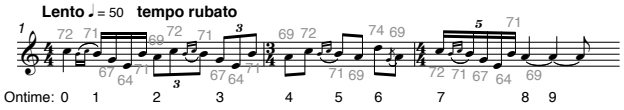

Figure 1. Bars 1-3 of the Introduction from *The Rite of Spring* by Igor Stravinsky, annotated with MIDI note numbers and ontimes in crotchets starting from zero. For clarity, phrasing is omitted and ornaments are not annotated.

cerpt consists of the notes C5, B4, G4, E4, B4, A4, ignoring ornaments for simplicity. The simplest way to discover the three occurrences of this pattern is to represent the excerpt as a string of MIDI note numbers and then to use an algorithm for pattern discovery in strings. The string 72, 71, 67, 64, 71, 69, ought to be discovered, and the user relates this back to the notes C5, B4, G4, E4, B4, A4. The geometric method is not appropriate here because each occurrence of the pattern has a different rhythmic profile.

On the other hand, the geometric method is better suited to finding the most salient pattern in Fig. 2a, consisting of all the notes in bar 13 except the tied-over G4. This pattern occurs again in bar 14, transposed up a fourth, and then once more at the original pitch in bar 15. Each note is annotated with its relative height on the stave (or *morphetic pitch number* [14]), taking C4 to be 60. Underneath the stave, ontimes are measured in quaver beats starting from zero. The first note in this excerpt, G3, can be represented by the datapoint $\mathbf{d}_1 = (0, 57)$, since it has ontime 0 and morphetic pitch number 57. A scatterplot of morphetic pitch number against ontime for this excerpt is shown in Fig. 2b. Meredith et al. [14] call the set of all datapoints representing an excerpt a *dataset*, denoted by D. Restricting attention to bars 13-15, we begin with the dataset

$$D = \{\mathbf{d}_1, \mathbf{d}_2, \ldots, \mathbf{d}_{26}\}. \quad (1)$$

A *pattern* is defined as a non-empty subset of a dataset. In our example, we will choose to look at the patterns

$$P = \{\mathbf{d}_1, \ldots, \mathbf{d}_8\}, \text{ and } Q = \{\mathbf{d}_9, \mathbf{d}_{11}, \ldots, \mathbf{d}_{17}\}. \quad (2)$$

The vector that translates $\mathbf{d}_1$ to $\mathbf{d}_9$ is

$$\mathbf{d}_9 - \mathbf{d}_1 = (3, 60) - (0, 57) = (3, 3) = \mathbf{v}. \quad (3)$$

We have given this vector a label $\mathbf{v} = (3, 3)$. It is this same vector $\mathbf{v}$ that translates $\mathbf{d}_2$ to $\mathbf{d}_{11}$, $\mathbf{d}_3$ to $\mathbf{d}_{12}, \ldots, \mathbf{d}_8$ to $\mathbf{d}_{17}$. Recalling the definitions of P and Q from (2), it is more succinct to say that 'the translation of P by $\mathbf{v}$ is equal to Q'. This translation is indicated in Fig. 2c.

Looking at Fig. 2c it is evident that as well as Q being a translation of P, pattern R is also a translation of P. Meredith et al. [14] call $\{P, Q, R\}$ the *translational equivalence class* of P in D, notated

$$TEC(P, D) = \{P, Q, R\}. \quad (4)$$

The TEC gives all the occurrences of a pattern in a dataset.

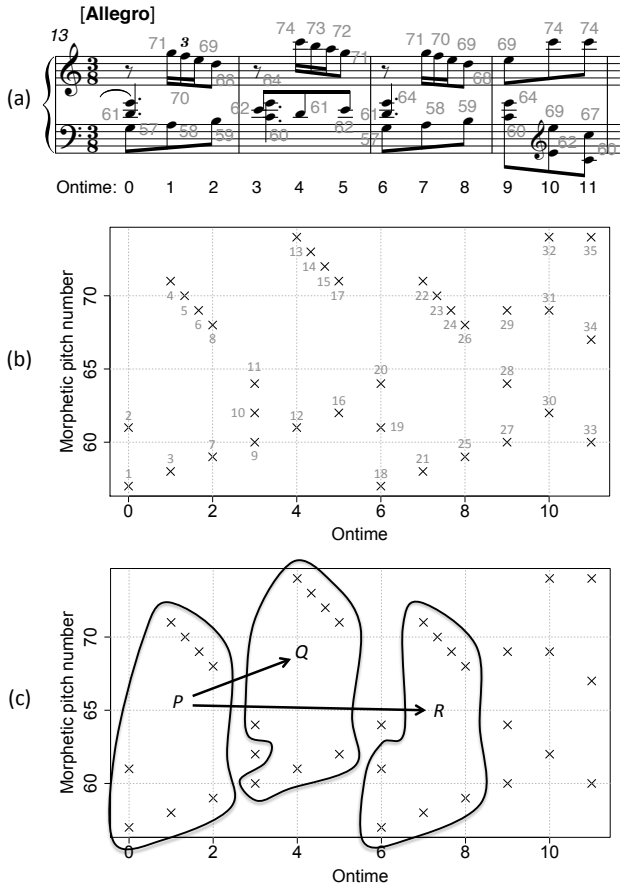

Figure 2. (a) Bars 13-16 of the Sonata in C major L3 by Scarlatti, annotated with morphetic pitch numbers and ontimes; (b) each note from the excerpt is converted to a point consisting of an ontime and a morphetic pitch number. Morphetic pitch number is plotted against ontime, and points are labelled in lexicographical order $\mathbf{d}_1$ to $\mathbf{d}_{35}$; (c) the same plot as above, with three ringed patterns, P, Q, R. Arrows indicate that both Q and R are translations of P.

So P is an example of a *translational pattern*, as translations of P, namely Q and R, exist in the dataset D. The formal definition of a translational pattern is as follows.

Definition. For a pattern P in a dataset D, the pattern P is a *translational pattern* if there exists at least one subset Q of D such that P and Q contain the same number of elements, and there exists *one* non-zero vector $\mathbf{v}$ that translates each datapoint in P to a datapoint in Q.

In the example in Fig. 2, two dimensions were considered (ontime and morphetic pitch number). The definitions and pattern discovery algorithms given by Meredith et al. [13] extend to k dimensions; specifically MIDI note number and duration are included as further dimensions.

The string-based method is not so well suited to Fig. 2a. The first step would be voice separation, generating perceptually valid melodies from the texture. Sometimes the scoring of the music makes separation simple [9], but even when voicing contains ambiguities, there are algorithms that can manage [1, 3]. Supposing fragments of the pattern in Fig. 2a were discovered among separated melodies,

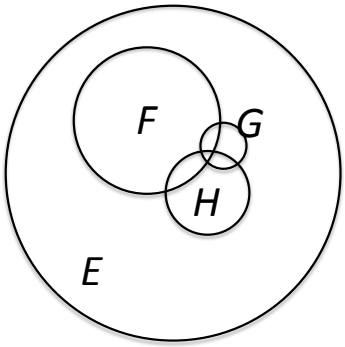

Figure 3. A Venn diagram (not to scale) for the number of patterns (up to translational equivalence) in a dataset. The total E is shown relative to the number typically returned by SIATEC (F), COSIATEC (G), and SIACT (H).

these fragments still would have to be correctly reunited. In this instance, even the most sophisticated string-based method [5] does not compare with the efficiency of the geometric method. The key difference between geometric and string-based approaches is the binding of ontimes to other musical information in the former, and the decoupling of this information in the latter. Both are valid methods for discovering patterns in music.

The reporting of existing *intra*-opus algorithms will often mention running time [3, 8, 12–14], occasionally *recall* is given [11, 17], and sometimes *precision* [10]. With the *inter*-opus discovery task an algorithm's output is seldom compared with a human benchmark [5,9]. The justification is that 'investigations of entire collections require considerable amounts of time and effort on the part of researchers' [9, p. 171]. Still, is it not worth knowing how an algorithm performs on a subset of the collection?

2. ALGORITHMS FOR PATTERN DISCOVERY

In equation 2, pattern P was introduced without explaining *how* it is discovered. It *could* be discovered by calculating all the TECs in the dataset D, and then certainly $TEC(P, D)$ will be among the output. However this approach is tremendously expensive and indiscriminate. It is expensive in terms of computational complexity, as there are 2^n patterns to partition into equivalence classes, where $n = |D|$ is the size of the dataset. Moreover, it is indiscriminate as no attempt is made to restrict the output in terms of 'musical importance': while P is arguably of importance, not all subsets of D are worth considering, yet they will also be among the output. The set E in Fig. 3 represents the output of this expensive and indiscriminate approach.

Therefore Meredith et al. [14] restrict the focus to a smaller set F, by considering how a pattern like P is *maximal*. Recalling (1) and (2), the pattern P is maximal in the sense that it contains all datapoints that are translatable in the dataset D by the vector $\mathbf{v} = (3, 3)$. It is called a *maximal translatable pattern* [14], written

$$P = MTP(\mathbf{v}, D) = \{\mathbf{d} \in D : \mathbf{d} + \mathbf{v} \in D\}. \quad (5)$$

Meredith et al.'s [14] structural inference algorithm (SIA) calculates all MTPs in a dataset, which requires $O(kn^2)$ calculations. While the TEC of each MTP must still be determined to give the set F in Fig. 3, this approach is enormously less expensive than partitioning 2^n patterns and involves a decision about musical importance: 'In music, MTPs often correspond to the patterns involved in perceptually significant repetitions' [14, p. 331]. SIA works by traversing the upper triangle of the similarity matrix

$$\mathbf{A} = \begin{pmatrix} \mathbf{d}_1 - \mathbf{d}_1 & \mathbf{d}_2 - \mathbf{d}_1 & \cdots & \mathbf{d}_n - \mathbf{d}_1 \\ \mathbf{d}_1 - \mathbf{d}_2 & \mathbf{d}_2 - \mathbf{d}_2 & \cdots & \mathbf{d}_n - \mathbf{d}_2 \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{d}_1 - \mathbf{d}_n & \mathbf{d}_2 - \mathbf{d}_n & \cdots & \mathbf{d}_n - \mathbf{d}_n \end{pmatrix}. \quad (6)$$

If the vector $\mathbf{w} = \mathbf{d}_j - \mathbf{d}_i$ is not equal to a previously calculated vector then a new MTP is created, $MTP(\mathbf{w}, D)$, with $\mathbf{d}_i$ as its first member. Otherwise $\mathbf{w} = \mathbf{u}$ for some previously calculated vector $\mathbf{u}$, in which case $\mathbf{d}_i$ is included in $MTP(\mathbf{u}, D)$. So it is possible to determine the set F for a dataset D by first running SIA on the dataset and then calculating the TEC of each MTP. The structural inference algorithm for translational equivalence classes (SIATEC) performs this task [14], and requires $O(kn^3)$ calculations.

To our knowledge there are two further algorithms that apply the geometric method to pattern discovery: the covering structural inference algorithm for translational equivalence classes (COSIATEC) [13] and a variant proposed by Forth & Wiggins [7]. COSIATEC *rates* patterns according to a heuristic for musical importance and produces a smaller output than SIATEC, the set labelled G in Fig. 3. The name COSIATEC derives from the idea of creating a *cover* for the input dataset:

1. Run SIATEC on $D_0 = D$, rate the discovered patterns using the heuristic for musical importance, and return the pattern P_0 that receives the highest rating.
2. Define a new dataset D_1 by removing from D_0 each datapoint that belongs to an occurrence of P_0.
3. Repeat step 1 for D_1 to give P_1, repeat step 2 to define D_2 from D_1, and so on until the dataset D_{N+1} is empty.
4. The output is

$$G = \{TEC(P_0, D_0), \ldots, TEC(P_N, D_N)\}. \quad (7)$$

Forth & Wiggins' variant on COSIATEC [7] uses a non-parametric version of the heuristic for musical importance and requires only one run of SIATEC. While only one run reduces the computational complexity of their version, it does mean that the output is always a subset of F, whereas running SIATEC on successively smaller datasets (steps 2 and 3 above) makes it possible to discover patterns beyond F (the portion $G \backslash F$ in Fig. 3).

3. THE PROBLEM OF ISOLATED MEMBERSHIP

In Sec. 2 we noted that pattern P from (2) could be discovered by running SIA on the dataset D from (1). This

is because P is the MTP (cf. equation 5) for the vector $\mathbf{v} = (3, 3)$ and SIA returns all such patterns in a dataset. However, D is a conveniently chosen example consisting only of bars 13-15 of Fig. 2a. How might an MTP be affected if the dataset is enlarged to include bar 16? Letting

$$D^+ = \{\mathbf{d}_1, \ldots, \mathbf{d}_{35}\}, \quad \mathbf{v} = (3, 3), \tag{8}$$

it can be verified that

$$P^+ = MTP(\mathbf{v}, D^+) = \{\mathbf{d}_1, \ldots, \mathbf{d}_8, \mathbf{d}_{18}, \mathbf{d}_{19}, \mathbf{d}_{22}\}. \tag{9}$$

Unfortunately P^+, the new version of P, contains three more datapoints, $\mathbf{d}_{18}$, $\mathbf{d}_{19}$, $\mathbf{d}_{22}$, that are isolated temporally from the rest of the pattern. This is an instance of what we call the 'problem of isolated membership'. It refers to a situation where a musically important pattern is contained *within* an MTP, along with other temporally isolated members that may or may not be musically important. Intuitively, the larger the dataset, the more likely it is that the problem will occur. Isolated membership affects all existing algorithms in the SIA family, and could prevent them from discovering some translational patterns that a music analyst considers noticeable or important (see Sec. 4 for further evidence in support of this claim).

Our proposed solution to the problem of isolated membership is to take the SIA output and 'trawl' *inside* each MTP from beginning to end, returning subsets that have a compactness greater than some threshold a and that contain at least b points. The *compactness* of a pattern is the ratio of the number of points in a pattern to the number of points in the region of the dataset in which the pattern occurs [13]. Different interpretations of 'region' lead to different versions of compactness. The version employed here is of least computational complexity $O(kn)$, and uses the lexicographical ordering shown in Fig. 2b. The *compactness* of a pattern $P = \{\mathbf{p}_1, \ldots, \mathbf{p}_l\}$ in a dataset D is defined by

$$c(P, D) = l/|\{\mathbf{d}_i \in D : \mathbf{p}_1 \preceq \mathbf{d}_i \preceq \mathbf{p}_l\}|. \tag{10}$$

For instance, the compactness of pattern Q in Fig. 2c is $8/9$, as there are 8 points in the pattern and 9 in the dataset region $\{\mathbf{d}_9, \mathbf{d}_{10}, \ldots, \mathbf{d}_{17}\}$ in which the pattern occurs.

One of Meredith et al.'s [14] suggestions for improving/extending the SIA family is to 'develop an algorithm that searches the MTP TECs generated by SIATEC and selects all and only those TECs that contain convex-hull compact patterns' [p. 341]. The way in which our proposed solution is crucially different to this suggestion is to trawl *inside* MTPs. It will not suffice to calculate the compactness of an entire MTP, since we know it is likely to contain isolated members. Other potential solutions to the problem of isolated membership are to:

- Segment the dataset before discovering patterns. The issue is how to segment appropriately—usually the discovery of patterns *guides* segmentation [2], not the other way round.
- Apply SIA with a 'sliding window' of size r. Approximately, this is equivalent to traversing only the elements on the first r superdiagonals of $\mathbf{A}$ in (6). The issue is that the sliding window could prevent the discovery of very noticeable or important patterns, if their generating vectors lie beyond the first r superdiagonals.
- Consider the set of all patterns that can be expressed as an *intersection* of MTPs, which may not be as susceptible to the problem of isolated membership. The issue with this larger class is that it is more computationally complex to calculate, and does not aim specifically at tackling isolated membership.

The algorithmic form of our solution is called a *compactness trawler*. It may be helpful to apply it to the example of P^+ in (9), using a compactness threshold of $a = 2/3$ and points threshold of $b = 3$. The compactness of successive subsets $\{\mathbf{d}_1\}, \{\mathbf{d}_1, \mathbf{d}_2\}, \ldots, \{\mathbf{d}_1, \ldots, \mathbf{d}_8\}$ of P^+ remains above the threshold of $2/3$ but then falls below, to $9/18$, for $\{\mathbf{d}_1, \ldots, \mathbf{d}_8, \mathbf{d}_{18}\}$. So we return to $\{\mathbf{d}_1, \ldots, \mathbf{d}_8\}$, and it is output as it contains $8 \geq 3 = b$ points. The process restarts with subsets $\{\mathbf{d}_{18}\}$, $\{\mathbf{d}_{18}, \mathbf{d}_{19}\}$, and then the compactness falls below $2/3$ to $3/5$ for $\{\mathbf{d}_{18}, \mathbf{d}_{19}, \mathbf{d}_{22}\}$. So we return to $\{\mathbf{d}_{18}, \mathbf{d}_{19}\}$, but it is discarded as it contains fewer than 3 points. The process restarts with subset $\{\mathbf{d}_{22}\}$ but this also gets discarded for having too few points. The whole of P^+ has now been trawled. The formal definition follows and has computational complexity $O(kn)$.

1. Let $P = \{\mathbf{p}_1, \ldots, \mathbf{p}_l\}$ be a pattern in a dataset D and $i = 1$.
2. Let j be the smallest integer such that $i \leq j < l$ and $c(P_{j+1}, D) < a$, where $P_{j+1} = \{\mathbf{p}_i, \ldots, \mathbf{p}_{j+1}\}$. If no such integer exists then put $P' = P$, otherwise let $P' = \{\mathbf{p}_i, \ldots, \mathbf{p}_j\}$.
3. Return P' if it contains at least b points, otherwise discard it.
4. If j exists in step 2, re-define P in step 1 to equal $\{\mathbf{p}_{j+1}, \ldots, \mathbf{p}_l\}$, set $i = j + 1$, and repeat steps 2 and 3. Otherwise re-define P as empty.
5. After a certain number of iterations P will be empty and the output can be labelled $P'_1, \ldots, P'_N$, that is N subsets of the original P, where $0 \leq N \leq l$.

We give the name 'structural inference algorithm and compactness trawler' (SIACT) to the process of calculating all MTPs in a dataset (SIA), followed by the application of the compactness trawler to each. The compactness-trawling stage in SIACT requires $O(kmn)$ calculations, where m is the number of MTPs returned by SIA. If desired, it is then possible to take the output of SIACT and calculate the TECs. These TECs are represented by the set H in Fig. 3. To our knowledge, this newest member of the SIA family is the only algorithm intended to solve the problem of isolated membership.

4. COMPARATIVE EVALUATION

A music analyst (the second author) analysed the Sonata in C major L1 and the Sonata in C minor L10 by Scarlatti, the Prelude in C♯ minor BWV849 and the Prelude in E major BWV854 by Bach. The brief was similar to the intra-opus discovery task described in Sec. 1: given a piece of music in staff notation, discover translational patterns that occur within the piece. Thus, a benchmark of translational patterns was formed for each piece, the criteria for benchmark membership being left largely to the analyst's discretion. One criterion that was stipulated was to think in terms of an analytical essay: if a pattern would be mentioned in prose or as part of a diagram, then it should be included in the benchmark. The analyst is referred to as 'independent' because of the relative freedom of the brief and because they were not aware of the details of the SIA family, or our new algorithm. The analyst was also asked to report where aspects of musical interest had little or nothing to do with translational patterns, as these occasions will have implications for future work.

Three algorithms—SIA [14], COSIATEC [13] and our own, SIACT—were run on datasets that represented L1, L10, BWV849, and BWV854. For COSIATEC the non-parametric version of the rating heuristic was used [7] and for SIACT we used a compactness threshold of $a = 2/3$ and a points threshold of $b = 3$. The choice of $a = 2/3$ means that at the beginning of an input pattern, the compactness trawler will tolerate one non-pattern point between the first and second pattern points, which seems like a sensible threshold. The choice of $b = 3$ means that a pattern must contain at least three points to avoid being discarded. This is an arbitrary choice and may seem a little low to some. Each point in a dataset consisted of an ontime, MIDI note number (MNN), morphetic pitch number (MPN), and duration (voicing was omitted for simplicity on this occasion). Nine combinations of these four dimensions were used to produce 'projections' of datasets [14], on which the algorithms were run. These projections always included ontime, bound to: MNN *and* duration; MNN; MPN *and* duration; MPN; duration; MNN mod 12 *and* duration; MNN mod 12; MPN mod 7 *and* duration; MPN mod 7. For the first time to our knowledge, the use of pitch modulo 7 and 12 enabled the concept of octave equivalence to be incorporated into the geometric method.

If a pattern is in the benchmark, it is referred to as a target; otherwise it is a non-target. An algorithm is judged to have discovered a target if a member of the algorithm's output is *equal* to the target pattern or a translation of that pattern. In the case of COSIATEC the output consists of TECs, not patterns. So we will say it has discovered a target if that target is a member of one of the output TECs. Table 1 shows the recall and precision of the three algorithms for each of the four pieces. Often COSIATEC did not discover any target patterns, so for these pieces it has zero recall and precision. This is in contrast to the parametric version's quite encouraging results for Bach's two-part inventions [12,13]. When it did discover some target patterns in L10, COSIATEC achieved a better precision than the other algorithms, as it tends to return far fewer patterns per piece (168 on average compared with 8,284 for SIACT and 385,299 for SIA). Hence the two remaining contenders are SIA and SIACT. SIACT, defined in Sec. 3, out-performs SIA in terms of both recall and precision. Having examined cases in which SIA and COSIATEC fail to discover targets, we attribute the relative success of SIACT to its being intended to solve the problem of isolated membership. Across the four pieces, the running times of SIA and SIACT are comparable (the latter is always slightly greater since the first stage of SIACT *is* SIA).

Piece →	L1	L10	BWV849	BWV854
Algorithm ↓	Recall			
SIA	.29	.22	.28	.22
COSIATEC	.00	.17	.00	.00
SIACT	.50	.65	.56	.61
	Precision			
SIA	$1.5\,e^{-5}$	$1.1\,e^{-5}$	$1.3\,e^{-5}$	$1.8\,e^{-5}$
COSIATEC	.00	.02	.00	.00
SIACT	$2.6\,e^{-3}$	$1.5\,e^{-3}$	$7.8\,e^{-4}$	$2.0\,e^{-3}$

Table 1. Results for three algorithms on the intra-opus pattern discovery task, applied to four pieces of music. Recall is the number of targets discovered, divided by the sum of targets discovered and targets not discovered. Precision is the number of targets discovered, divided by the sum of targets discovered and non-targets discovered.

5. DISCUSSION

This paper has discussed and evaluated algorithms for the intra-opus discovery of translational patterns. One of our motivations was the prospect of improving upon current solutions to this open MIR problem. A comparative evaluation was conducted, including two existing algorithms and one of our own, SIACT. For the pieces of music considered, it was found that SIACT out-performs the existing algorithms considerably with regard to recall and, more often than not, it is more precise. Therefore, our aim of improving upon the best current solution has been achieved. Central to this achievement was the formalisation of the 'problem of isolated membership'. It was shown that for a small and conveniently chosen excerpt of music, a maximal translatable pattern corresponded exactly to a perceptually salient pattern. However, when the excerpt was enlarged by just one bar, the MTP gained some temporally isolated members, and the salient pattern was lost inside the MTP. Our proposed solution, to trawl inside an MTP, returning compact subsets, led to the definition of SIACT.

The weight placed on the improved results reported here is limited somewhat by the extent of the evaluation, which includes only four pieces, all from the Baroque period, and all analysed by one expert. Extending and altering these conditions and assessing their effect on the performance of the three algorithms is a clear candidate for future work. There are also more sophisticated versions of

compactness and the compactness trawler algorithm that could be explored, and alternative values for the compactness and points thresholds, a and b. The discovery of 'exact repetition' has provided a sensible starting point for this research, but extending definitions such as (5) to allow for 'inexact repetition' is an important and challenging next step. Cases of failure, where `SIACT` does not discover targets, will be investigated. Perhaps some of these cases share characteristics that can be addressed in a future version of the algorithm. Although we have seen `SIA` presented before as the sorting of matrix elements [14], the connection that **A** in (6) makes with similarity matrices [15, 16] may lead to new insights or efficiency gains.

We will be trying to elicit more knowledge about the attributes of a pattern that matter to human analysts, so as to rank output patterns and to compare these attributes with the assumptions underlying `SIACT`. It could be that current pattern discovery methods overlook particular aspects of musical interest. If so a string-based or geometric method might be easily adapted, or very different methods may have to be developed. Could one focused algorithm encompass the many and diverse categories of musical pattern? It seems improbable, and the discussion of Figs. 1 and 2 in Sec. 1.1 could be interpreted as a counterexample. Hence, given the improved voice separation algorithms, and string-based and geometric methods that now exist, another worthy topic for future work would be the unification of a select number of algorithms within a single user interface. This would bring us closer to achieving our opening, more ambitious aim, of enabling music analysts, listeners, and students to engage with pieces of their choice in a novel and rewarding manner. To this end, the work reported here clearly merits further development.

6. ACKNOWLEDGEMENTS

This paper benefited from helpful discussions with David Meredith and Jamie Forth. We would also like to thank the four anonymous reviewers for their comments.

7. REFERENCES

[1] E. Cambouropoulos: "Voice and stream: perceptual and computational modeling of voice separation," *Music Perception*, Vol. 26, No. 1, pp. 75–94, 2008.

[2] E. Cambouropoulos: "Musical parallelism and melodic segmentation: a computational approach," *Music Perception*, Vol. 23, No. 3, pp. 249–267, 2006.

[3] S-C. Chiu, M-K. Shan, J-L. Huang, and H-F. Li: "Mining polyphonic repeating patterns from music data using bit-string based approaches," *Proceedings of the IEEE International Conference on Multimedia and Expo*, pp. 1170–1173, 2009.

[4] R. Clifford, M. Christodoulakis, T. Crawford, D. Meredith, and G. Wiggins: "A fast, randomised, maximal subset matching algorithm for document-level music retrieval," *Proceedings of the International Symposium on Music Information Retrieval*, 2006.

[5] D. Conklin and M. Bergeron: "Feature set patterns in music," *Computer Music Journal*, Vol. 32, No. 1, pp. 60–70, 2008.

[6] N. Cook: *A guide to musical analysis*, J.M. Dent and Sons, London, 1987.

[7] J. Forth and G. Wiggins: "An approach for identifying salient repetition in multidimensional representations of polyphonic music," *London algorithmics 2008: theory and practice*, College Publications, London, 2009.

[8] J-L. Hsu, C-C. Liu, and A. Chen: "Discovering nontrivial repeating patterns in music data," *IEEE Transactions on Multimedia*, Vol. 3, No. 3, pp. 311–325, 2001.

[9] I. Knopke and F. Jürgensen: "A system for identifying common melodic phrases in the masses of Palestrina," *Journal of New Music Research*, Vol. 38, No. 2, pp. 171–181, 2009.

[10] O. Lartillot: "Efficient extraction of closed motivic patterns in multi-dimensional symbolic representations of music," *Proceedings of the International Symposium on Music Information Retrieval*, pp. 191–198, 2005.

[11] C. Meek and W. Birmingham: "Automatic thematic extractor," *Journal of Intelligent Information Systems*, Vol. 21, No. 1, pp. 9–33, 2003.

[12] D. Meredith: "Point-set algorithms for pattern discovery and pattern matching in music," *Proceedings of the Dagstuhl Seminar on Content-Based Retrieval*, 2006.

[13] D. Meredith, K. Lemström, and G. Wiggins: "Algorithms for discovering repeated patterns in multidimensional representations of polyphonic music," *Proceedings of the Cambridge Music Processing Colloquium*, 2003.

[14] D. Meredith, K. Lemström, and G. Wiggins: "Algorithms for discovering repeated patterns in multidimensional representations of polyphonic music," *Journal of New Music Research*, Vol. 31, No. 4, pp. 321–345, 2002.

[15] G. Peeters: "Sequence representation of music structure using higher-order similarity matrix and maximum-likelihood approach," *Proceedings of the International Symposium on Music Information Retrieval*, 2007.

[16] X. Ren, L. Smith, and R. Medina: "Discovery of retrograde and inverted themes for indexing musical scores," *Proceedings of the ACM/IEEE Joint Conference on Digital Libraries*, pp. 252–253, 2004.

[17] P-Y. Rolland: "FlExPat: flexible extraction of sequential patterns," *Proceedings of the IEEE International Conference on Data Mining*, pp. 481–488, 2001.

[18] E. Ukkonen, K. Lemström, and V. Mäkinen: "Geometric algorithms for transposition invariant content-based music retrieval," *Proceedings of the International Symposium on Music Information Retrieval*, 2003.