A STATISTICAL ANALYSIS OF THE ABC MUSIC NOTATION CORPUS: EXPLORING DUPLICATION

Chris Walshaw

Department of Computing & Information Systems,
University of Greenwich, London SE10 9LS, UK
c.walshaw@gre.ac.uk

ABSTRACT

This paper presents a statistical analysis of the abc music notation corpus. The corpus contains around 435,000 transcriptions of which just over 400,000 are folk and traditional music. There is significant duplication within the corpus and so a large part of the paper discusses methods to assess the level of duplication and the analysis then indicates a headline figure of over 165,000 distinct folk and traditional melodies. The paper also describes TuneGraph, an online, interactive user interface for exploring tune variants, based on visualising the proximity graph of the underlying melodies.

1. INTRODUCTION

1.1 Background

Abc notation is a text-based music notation system popular for transcribing, publishing and sharing folk music, particularly online. Similar systems have been around for a long time but abc notation was formalised (and named) by the author in 1993 (Walshaw, 1993). Since its inception he has maintained a website, now at abcnotation.com, with links to resources such as tutorials, software and tune collections.

1.1.1 Tune search engine

In 2009 the functionality of the site was significantly enhanced with an online tune search engine, the basis of which is a robot which regularly crawls known sites for abc files and then downloads them. The downloaded abc code is cleaned and indexed and then stored in a database which backs the search engine front end. Users of the tune search are able to view and/or download the staff notation, midi representation and abc code for each tune, and the site currently attracts around ½ million visitors a year.

1.1.2 Breadth

The aim of the tune search is to index all abc notated transcriptions from across the web. However there are a number of reasons why it is unable to do this completely:

- Unknown / new abc sites: the robot indexer is seeded from around 350 known URLs (some of which are no longer active), but it does not search the entire web.
- HTML based transcriptions: in the main, the indexer searches for downloadable abc file types (.abc, or sometimes .txt). However, there are a number of sites where the abc code is embedded directly into a webpage. Mostly these tend to be small collections (especially if the abc code has to be manually inserted into the HTML code) and so these are omitted from the search. However, there are 3 larger collections which are included (by parsing the HTML and looking for identifiable start and end tags).
- JavaScript links: for a small number of sites the file download is enacted via JavaScript, making the link to the .abc file difficult to harvest.

1.1.3 Growth

Starting with an initial database of 36,000 tunes in 2009 the index has expanded to over 435,000 abc transcriptions at the time of writing (May 2014). Most of these are folk tunes and songs from Western Europe and North America, although two massive multiplayer online role-playing games, Lord of the Rings Online and Starbound, have adopted abc for their in-game music system resulting in a number of dedicated websites with mixed collections of rock, pop, jazz and, sometimes, folk melodies. The ~35,000 transcriptions from these sites are ignored for the purposes of this paper, leaving just over 400,000 to be analysed (though this number changes every time the robot runs).

Importantly, although each of the transcriptions comes from a distinct URL, over half are duplicates and these are a major focus of this study.

1.2 Aims

The original intention for this paper was to present a statistical survey of the abc music notation corpus in its current state (i.e. mid-2014) including analyses of the corpus segmented by key, meter and tune type. The purpose was threefold:

- To provide a historical marker of the notation system in its 20th year (abc2mtex v1.0, a transcription package which contained the first description of the abc syntax, was released in December 1993).
- To discuss the composition of this large online resource and give some insights into the issues of curating and managing it.
- To invite other academics to explore the corpus in detail: the author is willing to grant exceptional access to the database for academic study

and interested in collaborating with projects that wish to make use of it.

For the most part this paper still has these aims. However, in investigating the data, a fundamental question arose: how many distinct tunes are there in the corpus? That, in addition to the supplementary question: what is meant by “distinct” in the context of aural traditions (with all the variation that implies) which are transcribed electronically (sometimes in sketch form), published online (sometimes temporarily), and subsequently copied and republished freely by other web users (often with no modifications, but sometimes with additional notes and corrections)?

The remainder of this paper is organised as follows:

- Section 2 discusses duplication and attempts to answer the question of how many distinct tunes there are in the corpus. It also presents on-going development of a user interface to allow the exploration of tune variants.
- Having decided on a methodology for discounting duplicates, section 3 presents a (straightforward) statistical analysis of the corpus together with a number of observations and comments on the data.
- Finally, section 4 presents some conclusions and ideas for further work.

2. DUPLICATION

Duplication occurs widely within the abc corpus for a number of observable reasons:

- **Compilations**: particularly in the past, certain enthusiasts have published compilations of all the abc tunes they could find, gathered from across the web.
- **Selections**: some sites, usually those containing repertoires (perhaps that of a band or an open session), publish a selection of tunes gathered from other sites.
- **Ease-of-access**: a number of sites publish collections or sub-collections both as one-tune-per-file together with a single file containing all of the tunes.

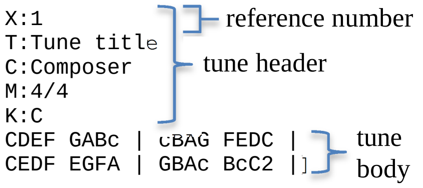
With respect to the tune search engine, there is little point in presenting users with dozens of identical results and so an important part of the pre-indexing clean-up involves identifying and, where appropriate, removing duplicates from the index. However, it is not necessarily clear which level of duplication to remove.

Furthermore, in the context of this paper, the elimination of duplicates is a fundamental process in determining how many distinct tunes there are in the corpus and the subsequent statistical analysis.

2.1 Eliminating duplicates

2.1.1 Classification

To discuss this topic further it is helpful to consider the structure of an abc tune transcription (see example in Figure 1).



```
X:1
T:Tune title
C:Composer
M:4/4
K:C
CDEF GABc | cBAG FEDC |
CEDF EGFA | GBAc BcC2 |]
```

Figure 1. An example abc transcription.

Each tune consists of a **tune header** (including a **reference number**) and the **tune body**.

The header contains descriptive meta-data mostly, though not exclusively, with no musical information. Typically this includes the title and composer (where known), but amongst other data may also include information about where the tune was sourced (book, recording, etc.), who transcribed it, historical notes and anecdotes and instrumentation details (particularly for multi-voice music).

The tune body contains the music, and may also contain song lyrics.

With this structure in mind, duplication can be classified into 4 increasingly broad categories:

- **Electronic**: the duplicates are electronically identical (the exact same string of characters) – i.e. the tune headers and bodies are identical (although in practice this is relaxed somewhat by ignoring the reference number and any whitespace).
- **Musical**: the duplicates are musically identical (including song lyrics) although they may contain different meta-data in the tune header – i.e. the tune bodies are identical.
- **Melodic**: neglecting any song lyrics, grace notes, decorations and chord symbols, the first voice of each duplicate is identical – i.e. the primary melodies are identical.
- **Incipit**: when transposed to the same key, the duplicates are melodically identical over the first few bars of the tune.

2.1.2 Implementation

Code which analyses and counts the size of each category has been developed. In the first three categories this is done without actually parsing the abc music notation in the tune body: for the most part it involves stripping the transcriptions of data, for example by extracting parts and removing decorations, lyrics, grace notes, etc.

For each duplication class, the code derives a comparison string from each abc transcription which is then compared with all other comparison strings in that class: identical strings indicate duplicates.

As a small percentage of transcriptions contain errors and / or extraneous text, part of the parsing task involves exception handling. These can arise for a number of reasons including: misplaced characters which do not fit with agreed abc syntax and transcription errors such as unmatched start / end tags (for example, in abc syntax grace notes are delimited with curly braces, { … }, – an exception is thrown if one of the braces is missing).

Transcriptions which cannot be parsed, or are empty, fall back on the previous classification. In other words if an exception is thrown when a transcription is being parsed for *incipit* comparison, the comparison string reverts to a *melodic* comparison string. Likewise, if a transcription contains an empty tune body (as can often happen when abc headers are used as placeholders or for indexing purposes) then the *melodic* and *musical* comparison strings would revert to *electronic*.

2.1.3 Results

Table 1 shows the duplication results for the 4 duplicate classes. Here a **duplicate cluster** refers to a group of identical transcriptions. A cluster of size n has 1 primary transcription and $n - 1$ duplicates, so the number of duplicates (column 4) refers to the total number of duplicated transcriptions with a contribution of $n - 1$ duplicates from each cluster.

Class	#duplicate clusters	max. duplicate cluster size	#duplicates
Electronic	71,156	39	171,203
Musical	75,752	132	222,241
Melodic	73,199	132	232,528
Incipit	58,090	207	281,552

Table 1. The different levels of duplication.

As one might expect, the number of duplicates (and the maximum duplicate cluster size) increases with each successive class, since the duplication refers to a diminishing portion of each transcription. The increase and subsequent decrease of duplicate clusters is less intuitive, but is easily explained: for example, if there are two duplicate clusters of sizes n_1 and n_2 which differ from each other only after the 4th bar, then under melodic duplication this would result in two clusters whereas under incipit duplication it would result in a single cluster of size $n_1 + n_2$.

To interpret the figures further, consider melodic duplicates: of the 400,160 transcriptions, 232,528 (58.1%) are duplicates and can be excluded from the statistical analysis. Of the remaining 167,632 transcriptions, 73,199 (18.3%) have a duplicate in the excluded set and therefore 94,433 (23.6%) are not duplicated anywhere in the corpus. The maximum duplicate cluster size is 132 (in other words there is 1 tune with 131 excluded duplicates) and the average cluster size is 4.18, i.e. (232,528 + 73,199) / 73,199.

Whilst this indicates a very substantial amount of duplication within the corpus, this gives a headline figure of 167,632 distinct melodies, even when all of the metadata, decoration and lyrics are stripped away. Doubtless that some of these are very minor variants or corrections, but nonetheless it indicates that the abc music notation corpus represents a substantial online resource.

2.2 Exploring variants

The algorithm that is used for identifying incipit duplicates is actually based on a difference metric which numerically quantifies the difference between each pair of incipits. Pairs of melodies with a difference of 0 are duplicates (at least for the length of the incipit), but those with small difference values are very likely to be tune variants.

Tune variants are an important part of folk music's aural tradition and so near duplicates which appear only in the incipit category are of interest to researchers and musicians alike. However they are not always easy to identify by eye from a large number of search results.

2.2.1 TuneGraph

To facilitate user exploration of such variants the author is developing TuneGraph (Walshaw, 2014), an online tool for the visual exploration of melodic similarity, outlined below.

Given a corpus of melodies, the idea behind TuneGraph is to calculate the difference between each pair of melodies numerically with a difference metric or similarity measure (e.g. Kelly, 2012; Stober, 2011; Typke, Wiering, & Veltkamp, 2005). Next a proximity graph is formed by representing every tune with a vertex and including (weighted) edges for every pair of vertices which are "similar". Finally, the resulting graph can be visualised using standard graph layout techniques such as force-directed placement, (e.g. Walshaw, 2003), either applied to the entire graph or just to a vertex and its neighbours (i.e. a tune and similar melodies).

The concept is not dissimilar to a number of other software systems which give a visual display of relationships between tunes, often based on a graph (e.g. Langer, 2010; Orio & Roda, 2009; Stober, 2011).

TuneGraph consists of two parts – TuneGraph Builder, which analyses the corpus and constructs the required graphs, and TuneGraph Viewer, which provides the online and interactive visualisation.

2.2.2 The difference metric

In the current implementation, each melody is represented by quantising the first 4 bars (the incipit) into 1/64th notes and then constructing a pitch vector (or pitch contour) where each vector element stores the interval, in semitones, between the corresponding note and the first note of the melody (neglecting any anacrusis). Since everything is calculated as an interval it is invariant under transposition.

The difference metric then calculates the difference between two pitch vectors either using the 1-norm (i.e. the sum of the absolute values of the differences between each pair of vector elements) or the 2-norm (i.e. the square root of the sum of squared differences between each pair of vector elements). The 1-norm has long been available as part of the abc2mtex indexing facilities (Walshaw, 1994), but experimentation suggests that the 2-norm gives marginally better results (Walshaw, 2014).

If the pitch vectors have different lengths then the sum is over the length of the shorter vector (although see below – section 2.2.4).

Similarity measures of this kind are well explored in the field of music information retrieval, (e.g. Kelly, 2012; Typke et al., 2005), and there may be other, more advanced similarity measures that would work even better. However, in principle any suitable metric can be used to build the proximity graph, provided that it expresses the difference between pairs of melodies with a single numerical value. Indeed, even combinations of similarity measures could be used by forming a weighted linear combination of their values.

2.2.3 Building the proximity graph

The proximity graph is formed by representing every tune with a vertex and including (weighted) edges for every pair of vertices which are "similar" (i.e. every pair where the numerical difference is below some threshold value). However the question arises: what is a suitable threshold and how should it be chosen?

Perhaps the simplest choice, and one which is well-known for geometric proximity graphs, is to find the smallest threshold value which results in connected graph, i.e. a graph in which a path exists between every pair of vertices. Although computationally expensive, this can be done relatively straightforwardly starting with an initial guess at a suitable threshold and then either doubling or halving it until a pair of bounding values are found, one of which is too small (and does not result in a connected graph) and one of which is large enough (and does give a connected graph). Finally the minimal connecting threshold (minimal so as to exclude unnecessary edges) can be found with a bisection algorithm, bisecting the interval between upper and lower bounds each iteration.

This was the first approach tried but it resulted in graphs with an enormous number of edges; the test code ran out of memory as the number of edges approached 200,000,000 and the threshold under test had not, at that point, yielded a connected graph.

Further investigation revealed the basic problem: the graph is potentially very dense in some regions, with many similar melodies clustered together, whereas elsewhere there are outlying melodies which are not similar to any others. This means that in order to connect the outliers, and hence the entire graph, the threshold has to be so large that in the denser regions huge cliques are generated.

2.2.4 Segmentation by meter

In order to reduce the density of the graph, one successful approach tested was to segment the graph by meter – i.e. so that tunes with different meters are never connected. In fact a simple way to implement this is to avoid connecting pitch vectors with different lengths. This has the added benefit that some meters can be connected (i.e. those with the same bar length such as 2/2 and 4/4) meaning that the strategy is blind to certain variations in transcription preferences (although not universally as it will fail to connect related melodies, such as Irish single jigs, which are variously transcribed in 6/8 and 12/8, and French 3-time bourrées, which can be either 3/4 or 3/8).

Each pitch vector length results in a subset of graph vertices: in all there were 314 subsets, ranging in size from 63,581 vertices (for length 256 – e.g. 2/2 and 4/4 tunes), down to 115 subsets containing just one vertex. However, 98.7% of vertices are in a subset of size 100 or more and 99.7% are in a subset of size 10 or more.

The small subsets generally result from unusual vector lengths, usually because of errors in the transcriptions (i.e. extra notes or incorrect note lengths) and there was often no close relation between the melodies, meaning that a very high threshold would have to be used to connect that subset. To avoid connecting very different transcriptions, for each segment the edge threshold was somewhat arbitrarily limited to the length of the pitch vector for that segment. In most cases, this upper limit was never needed, but for very small subsets it sometimes meant that no edges were generated at all.

2.2.5 Average degree

Even with segmentation by meter in place the method can still generate huge graphs. However, there is no particular reason that the graph needs to be connected so the idea of trying to build a connected graph (or connected subgraphs, one for each pitch vector length) was abandoned as unpractical. Nevertheless, it is attractive as essentially parameter-free and it does work for small collections of relatively closely related tunes (for example, English morris tunes, where there are many similar variants of the same melody).

For the purposes of representing the entire corpus as a (disconnected) proximity graph, this still leaves the choice of a suitable edge threshold open, but rather than picking a value out of the air, instead a *target average degree* is chosen for the resulting graph. With this average degree as a user-selected parameter the same bounding and bisection method as above can be used to find the smallest threshold that yields this average degree.

An important observation was that the small number of vertices which have very many similar neighbours generate a relatively large number of edges in the graph. For example a cluster of, say, 100 very similar melodies will form a (near) clique with up to 4,950 edges. This significantly skews the average if it is expressed as the mean degree. However, using the median degree ignores these outlying values and gave much more useful results empirically and so the current implementation uses this measure to calculate the average.

Considerable experimentation has been carried out with a number of average degree values (see Walshaw, 2014, for a full discussion) and the best – i.e. the one which yields local graphs (see below) that are small enough to be useful in search but which are sufficiently rich enough to express similarities visually – seems to be an average (median) degree of 3.

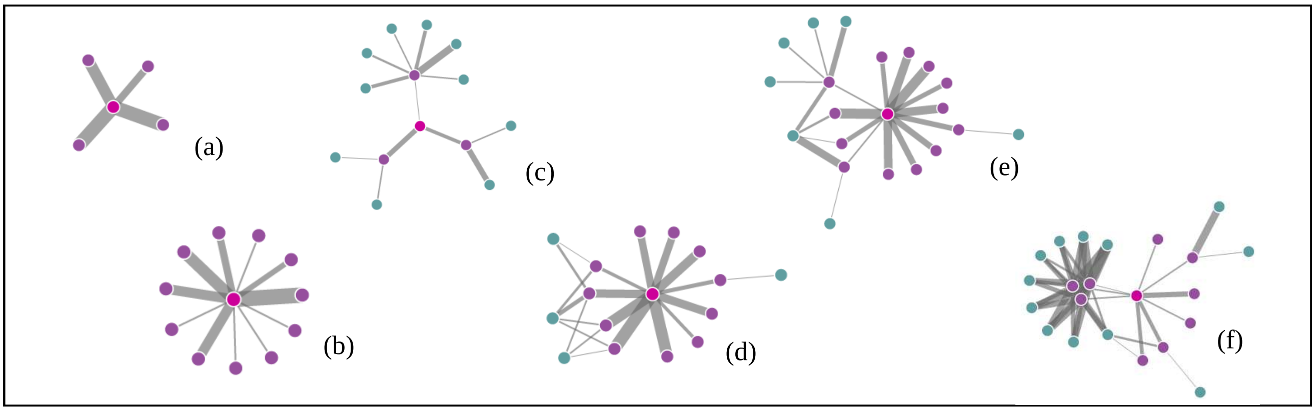


Figure 2. Some sample local graphs.

2.2.6 Extracting local graphs

Once suitable parameters have been chosen the graph is built as a series of proximity (sub-)graphs (one for each one for each pitch vector length). Each proximity subgraph is unlikely to be connected and as a result the graph as a whole can be highly disconnected.

One option is to use multilevel force directed graph placement (Walshaw, 2003), to find a layout for the entire graph. This has been tried and yields an interesting, but not necessarily very useful, representation of the corpus.

Instead, to allow exploration of similarities in an interactive online setting, the TuneGraph Builder code extracts a **local graph** for each non-isolated vertex. One way to do this is simply to extract the vertex, plus all its neighbours plus any edges between them. However, this can lead to clique-like local graphs where edges are hard to discern.

Instead, the local graph is built in layers: the seed (layer 0) is the original vertex for which the local graph is being built, layer 1 is any vertices neighbouring layer 0 and layer 2 is any vertices (not already included) neighbouring layer 1, etc. In order to maximise the clarity of the local graph, it only includes edges between layers and excludes edges between vertices in the same layer.

If the local graphs are just built from layers 0 and 1, each will be star-like, as in Figure 2(a) and Figure 2(b), yielding limited immediate visual information to the user (other than the number of neighbours and the strength of the relationships). Instead the builder code uses layers 0, 1 and 2, e.g. Figure 2(c) to Figure 2(f), to show some of the richness of certain neighbourhoods. Here colours indicate the layers, with layer 0 shown in crimson, layer 2 in light blue, and layer 1 interpolated between the two of them.

Finally, the graph edges are all weighted in inverse proportion to the difference between the two transcriptions that they connect (Walshaw, 2014). Since graph edge weights are indicated in the online tool by their thickness this conveys helpful visual information to the user by showing the more closely related tunes with thicker lines between them (and also affects how the graph is laid out by force directed placement).

2.2.7 Results

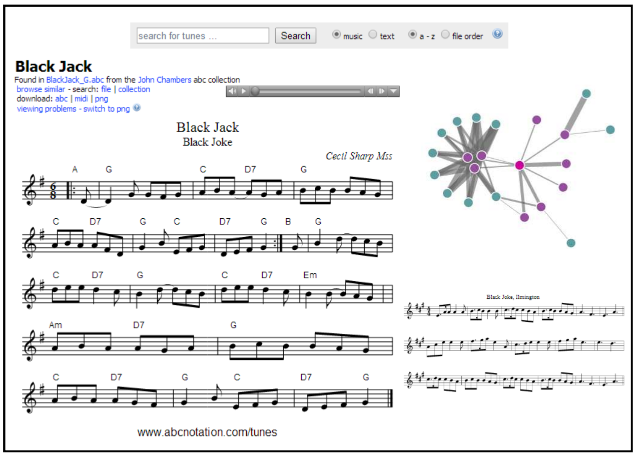
It is difficult to say exactly what features are desirable in the final graph, but experience with the local graphs suggests that they should be small enough not to overwhelm the user, but rich enough to convey some useful information. In particular the aim was to limit the maximum local graph size but maximise the average size. Experimentation was carried out with a number of different parameter settings (Walshaw, 2014) and often a small change can make a huge difference – for example, changing the target median degree from 3 to 4 increases the maximum local graph size from 121 to 724. However, the best parameters found were:

- Difference norm: $||.||_2$ – see section 2.2.2
- Segmentation by meter: true – see section 2.2.4
- Edge threshold limit: pitch vector length – see section 2.2.4
- Target average degree: median of 3 – see section 2.2.5

Using these settings results in a large number of isolated vertices, usually because there are no closely related melodies in the corpus or, less commonly, because there are no other transcriptions with the same pitch length. Eliminating these isolated vertices gave a final graph of 111,230 vertices in 31,784 connected subsets (many with as few as 2 vertices). The graph contains 250,182 edges, with a maximum degree of 68 and a minimum degree of 1, but is very sparse since the average degree is only 4.5.

From this 111,230 local graphs were produced with an average size of 6.1 vertices. The maximum size was 121 vertices and 468 edges. Whilst the largest local graphs can be difficult to visualise well, a random sample of the rest are of a size and complexity which both helps explore similarities without overwhelming the user.

Figure 2 shows some interesting examples: Here (a) and (b) come from local clique-like graphs with no immediate neighbours (recall that edges between vertices in the same layer are not included in the local graph so not all edges of the clique are shown). The tree shown in (c) indicates a number of tunes which are related but probably not immediate relations of each other. The graphs in (d) and (e) are similar to (b) only with some outlying tunes related to those in the clique. Finally the graph in (f) shows a tune on the edge of a tightly coupled clique.



Found in BlackJack_G.abc from the John Chambers abc collection

search for tunes ...
Search
● music ○ text ● a - z ○ file order
Black Jack
browse similar - search: file | collection
download: abc | midi | png
viewing problems - switch to png
Black Jack
Black Joke
Cecil Sharp Mss
A G C D7 G
C D7 G C D7 G B G
C D7 G C D7 Em
Am D7 G
C D7 G C D7 G
Black Joke, Dimington
www.abcnotation.com/tunes

Figure 3. An example webpage.

2.2.8 TuneGraph Viewer

Although only in prototype version, TuneGraph Viewer contains a number of interactive features. The local graph is displayed on a webpage alongside the tune it corresponds to. It is visualised as a dynamic layout using D3.js (Bostock, 2012), a JavaScript library for manipulating documents based on data, and employing the inbuilt force-directed placement features.

It provides the following user interface:

- The graph vertices find their own natural position dynamically via force directed placement and vertices can be dragged to rearrange the layout (other vertices then relocate accordingly).
- Vertex colour indicates the relationship to the root vertex.
- Edge thickness indicates visually how closely related two vertices are (i.e. how similar their corresponding tunes are).
- Moving the mouse over a vertex reveals its name and displays the associated melody.
- Double clicking on a vertex (other than the root vertex) takes the user to the corresponding page (with its own tune graph).

Figure 3 shows an example webpage corresponding to the tune Black Jack (a well-known English tune). The tune is displayed on the left (the abc notation would appear underneath) and the local tune graph is shown on the right. If the user moves their mouse over one of the graph vertices, the tune associated with that vertex appears below.

3. STATISTICAL ANALYSIS

This section presents a brief and straightforward statistical analysis of the current abc music corpus (May 2014) based on those tunes found online by the abc search engine. It does not, of course, cover unpublished collections and so there are no real means to estimate what proportion of the abc corpus it represents.

Broadly speaking the analysis is qualitatively similar regardless of which method is used for eliminating duplicates. As a single example, neglecting the 171,203 *electronic* duplicates, 29.8% of the remaining melodies are transcribed in 4/4. With 222,241 *musical* duplicates removed this figure is 30.3% and respectively comes out at 30.7% and 32.4% when the 232,528 *melodic* or 281,552 *incipit* duplicates are removed.

To avoid filling the paper with statistics the rest of this section therefore concentrates on just one category of duplicates. In fact, *incipit* duplicates may not be duplicates at all – they may just have the same first four bars, so all of the following figures analyse the 167,632 distinct melodies remaining when the 232,528 *melodic* duplicates are removed from the corpus.

First note that, although abc is primarily used for monophonic tunes, of these 167,632 melodies, 6,480 (3.9%) are polyphonic and 12,574 (7.5%) are songs (i.e. with lyrics included in the abc transcription).

The tables below show an analysis of the corpus segmented by meter, rhythm (i.e. tune type), and in particular the key (a very expressive field in abc which allows the specification by mode).

It was also intended to include a table showing the corpus segmented by origin. However, this proved problematic for a number of reasons, specifically:

- The abc header field to specify origin (`O:`) allows free text and hence a wide variation in attribution and even spelling.
- The origin header field is not widely used and only 26.2% of tunes in the corpus make use of it.
- One particularly large collection (a compilation of other collections) has the default origin set to “England”, when many of the tunes are clearly identifiable as Irish or Scottish – this significantly distorts the results.

Nevertheless, the origin analysis does indicate significant diversity, with substantial contributions (i.e. more than 1,000 transcriptions) from, in alphabetical order, China, England, France, Germany, Ireland, Scotland, Sweden & Turkey.

For each of the three tables that are included, key signature, meter and rhythm, the table shows all values with a count of 100 or more; any values with fewer than 100 instances are aggregated at the bottom.

3.1 Key signature

Table 2 shows the corpus segmented by key signature.

In abc, the key field is very expressive and allows the use of modes and even arbitrary accidentals, i.e. specified in the key signature and applied to all notes in the tune (unless overridden by another accidental applied to the individual note or notes in that bar).

There is even an option for the Great Highland Bagpipe (written `K:HP` in abc notation) where, by convention, tunes are usually played in Bb mixolydian but written in A mixolydian with no key signature (i.e. the C# and F#

are assumed but not written on the score). This is a throwback to the early days of abc and might now be better handled with an “omit-key-signature” output flag. Nonetheless, there are 2,326 transcriptions of this type.

Of more interest is the use of modes, the most common being A dorian with 3,638 transcriptions. In fact a survey of the entire range of key signatures (including aggregated values at the bottom of the table) shows that dorian is used for 9,008 transcriptions (5.4% of the corpus), mixolydian for 4,772 (2.9%), phrygian for 418 (0.3%), lydian for 85 (0.1%), aeolian for 84 (0.1%), ionian for 6 (0.0%) and locrian for 4 (0.0%). In addition, 19,596 of transcripitions (11.69%) are specified as being in a minor key.

Key signature	Count	Percentage	Cumulative
G	45,561	27.18%	27.18%
D	37,834	22.57%	49.75%
C	14,583	8.70%	58.45%
A	12,132	7.24%	65.69%
F	9,784	5.84%	71.52%
E minor	5,017	2.99%	74.52%
A minor	5,005	2.99%	77.50%
Bb	4,613	2.75%	80.25%
D minor	3,708	2.21%	82.46%
A dorian	3,638	2.17%	84.63%
G minor	2,995	1.79%	86.42%
E dorian	2,478	1.48%	87.90%
Great Highland Bagpipe	2,326	1.39%	89.29%
A mixolydian	1,957	1.17%	90.45%
B minor	1,945	1.16%	91.61%
none	1,798	1.07%	92.69%
D mixolydian	1,768	1.05%	93.74%
Eb	1,461	0.87%	94.61%
D dorian	1,172	0.70%	95.31%
E	1,115	0.67%	95.98%
G dorian	1,067	0.64%	96.61%
other	997	0.59%	97.21%
G mixolydian	593	0.35%	97.56%
C minor	561	0.33%	97.90%
Ab	286	0.17%	98.07%
C dorian	269	0.16%	98.23%
C mixolydian	240	0.14%	98.37%
B dorian	177	0.11%	98.48%
F minor	127	0.08%	98.55%
F# minor	127	0.08%	98.63%
E phrygian	117	0.07%	98.70%
other keys]	2,181	1.30%	100.00%

Table 2. A breakdown of the corpus by key.

3.2 Meter

Table 3 shows the corpus segmented by meter.

It is noticeable that much of the corpus is represented by meters common in Western European / North American folk music but there are significantly fewer of the more complex meters such as 7/8, 11/8, 15/8, etc., often found in Eastern Europe (9/8 is well represented but also includes slip jigs, commonly found in the British Isles).

Meter	Count	Percentage	Cumulative
4/4	51,493	30.72%	30.72%
6/8	34,840	20.78%	51.50%
2/4	22,378	13.35%	64.85%
2/2	19,764	11.79%	76.64%
3/4	19,614	11.70%	88.34%
free	6,006	3.58%	91.92%
9/8	3,679	2.19%	94.12%
3/8	2,166	1.29%	95.41%
6/4	1,868	1.11%	96.53%
12/8	1,425	0.85%	97.38%
3/2	1,411	0.84%	98.22%
4/2	409	0.24%	98.46%
7/8	371	0.22%	98.68%
8/8	317	0.19%	98.87%
9/4	302	0.18%	99.05%
10/8	293	0.17%	99.23%
5/4	122	0.07%	99.30%
5/8	113	0.07%	99.37%
[other meters]	1,061	0.63%	100.00%

Table 3. A breakdown of the corpus by meter.

3.3 Rhythm

Table 4 shows the corpus segmented by rhythm (tune type).

Unlike key signature and meter this is not a compulsory or assumed field (i.e. if no meter is specified, common time is assumed) and as result not all transcriptions have a rhythm indicated; nonetheless, 104,792 (62.5%) of them do.

Of interest in this table are the rhythms that indicate a specific origin. Reels, jigs and hornpipes are found widely in music from the British Isles and North America and the waltz, polka and schottische even more widely in Western European music. However, the strathspey indicates a Scottish origin – anecdotally there may be so many because of the large number of 19th Century tunebooks being transcribed into abc.

The polska and slängpolska indicate a Nordic origin, mostly likely Swedish, but found in other countries too and many come from a thriving wiki-based website, www.folkmusic.se.

Rhythm	Count	Percentage	Cumulative
no rhythm specified	62,840	37.49%	37.49%
reel	27,881	16.63%	54.12%
jig	20,353	12.14%	66.26%
hornpipe	6,943	4.14%	70.40%
waltz	4,636	2.77%	73.17%
strathspey	4,227	2.52%	75.69%
air	3,923	2.34%	78.03%
polka	3,863	2.30%	80.33%
march	2,343	1.40%	81.73%
slip jig	2,085	1.24%	82.98%
song	1,878	1.12%	84.10%
polska	1,663	0.99%	85.09%
barndance	1,181	0.70%	85.79%
country dance	1,126	0.67%	86.46%
slide	1,110	0.66%	87.13%
slängpolska	787	0.47%	87.60%
double jig	772	0.46%	88.06%
mazurka	581	0.35%	88.40%
dance	498	0.30%	88.70%
schottische	433	0.26%	88.96%
bourrée	386	0.23%	89.19%
triple hornpipe	379	0.23%	89.41%
quadrille	359	0.21%	89.63%
xiraldilla	247	0.15%	89.78%
minuet	221	0.13%	89.91%
miscellaneous	200	0.12%	90.03%
schottis	170	0.10%	90.13%
zwiefacher	135	0.08%	90.21%
single jig	123	0.07%	90.28%
other	108	0.06%	90.35%
set dance	106	0.06%	90.41%
[other rhythms]	16,075	9.59%	100.00%

Table 4. A breakdown of the corpus by rhythm.

4. CONCLUSION

This paper has presented a straightforward statistical analysis of the abc music notation corpus. The corpus contains around 435,000 transcriptions of which just over 400,000 are folk and traditional music.

There is significant duplication within the corpus and so a large part of the paper has discussed methods to assess the level of duplication. This has indicated a headline figure of over 165,000 distinct folk and traditional melodies. Much of the corpus seems to come from Western European and North American traditions, but there is a wide diversity included.

The paper has also described TuneGraph, an online interactive user interface for exploring tune variants, based on building a proximity graph of the underlying melodies. Although currently only in prototype form the intention is to deploy it on two sites with which the author is involved, abcnotation.com and the Full English Digital Archive at the Vaughan Williams Memorial Library (EDFSS, 2013).

4.1 Future work

The main focus for future work is to enhance the capabilities of TuneGraph. In particular it is intended to explore some of the wide range of similarity measures that are available as a means to build the proximity graph. As was indicated in section 2.2.2 there may be other, more advanced similarity measures, or combinations of similarity measures, that would work better than the 2-norm of the difference between pitch vectors.

5. REFERENCES

Bostock, M. 2012. Data-Driven Documents (d3.js), a visualization framework for internet browsers running JavaScript. http://d3js.org/

EDFSS. 2013. The Full English Digital Archive. *The Vaughan Williams Memorial Library*. http://www.efdss.org/efdss-the-full-english

Kelly, M. B. 2012. *Evaluation of Melody Similarity Measures*. Queen's University, Kingston, Ontario.

Langer, T. 2010. Music Information Retrieval & Visualization. In *Trends in Information Visualization*, pp. 15–22.

Orio, N., & Roda, A. 2009. A Measure of Melodic Similarity based on a Graph Representation of the Music Structure. *ISMIR*, pp 543–548.

Stober, S. 2011. Adaptive Distance Measures for Exploration and Structuring of Music Collections, Section 2, 1–10.

Typke, R., Wiering, F., & Veltkamp, R. C. 2005. A survey of music information retrieval systems. In *Proc. ISMIR*, pp. 153–160.

Walshaw, C. 1993. *ABC2MTEX: An easy way of transcribing folk and traditional music, Version 1.0*. University of Greenwich, London.

Walshaw, C. 1994. *The ABC Indexing Guide Version 1.2*. University of Greenwich, London.

Walshaw, C. 2003. A Multilevel Algorithm for Force-Directed Graph-Drawing. *Journal of Graph Algorithms and Applications*, *7*3, 253–285.

Walshaw, C. 2014. TuneGraph: an online visual tool for exploring melodic similarity. In *Proc. Digital Research in the Humanities and Arts (submitted)*. London.