MEASURING SIMILARITY OF AUTOMATICALLY EXTRACTED MELODIC PITCH CONTOURS FOR AUDIO-BASED QUERY BY HUMMING OF POLYPHONIC MUSIC COLLECTIONS

José Javier Valero Mas

MASTER THESIS UPF 2013

Master in Sound and Music Computing

Master Thesis Supervisors:

Justin Jonathan Salamon

Department of Information and Communication Technologies

Universitat Pompeu Fabra, Barcelona

Emilia Gómez Gutiérrez

Department of Information and Communication Technologies

Universitat Pompeu Fabra, Barcelona

Abstract

A study of melodic similarity of pitch contours automatically obtained from audio files in the context of Query by Humming is presented. Pitch contours are extracted directly from monophonic (query files) and polyphonic (commercial songs) audio files using a state-of-the-art algorithm *MELODIA* [SG12] for automatic estimation of predominant melodic contours. The contours are then coded using the *Symbolic Aggregate Approximation* [LKWL07] algorithm for the reduction of the huge amount of information each sequence contains, avoiding any step of automatic music transcription, and then compared using the subsequence matching and time warping method *Smith-Waterman* [SW81], being then an audio-based comparison.

Results using the commented approach do not reach state-of-the-art results obtained by other authors in audio-based Query by Humming and, analyzing the results, the main conclusion is that *Symbolic Aggregate Approximation* might not be appropriate for this task.

Resum

En aquesta Tesi es presenta un estudi sobre similitud entre contorns de freqüència fundamental obtinguts de fitxers d'àudio aplicat a Query by Humming. Els contorns de freqüència fundamental són extrets de fitxers d'àudio monofónic (per a les peticions dels usuaris) i polifónics (per a les cançons comercials) fent ús de l'algorisme *MELODIA* [SG12] per a l'estimació automàtica de contorns melòdics predominants. Aquestos contorns són codificats utilitzant l'algorisme *Symbolic Aggregate Approximation* [LKWL07] per a poder reduir la gran quantitat de dades que s'han de manejar, evitant la transcripció automàtica de música, i comparant-los amb l'algorisme *Smith-Waterman* [SW81], tenint per tant una comparació basada en àudio i no informació simbòlica com si d'una partitura es tractara.

Els resultats no alcancen els que altres algorismes de Query by Humming basats en informació d'àudio obtenen i, analitzant aquestos resultats, la principal conclusió és que l'algorisme *Symbolic Aggregate Approximation* no pareix apropiat per a aquesta tasca.

Acknowledgements

Turns out to be quite difficult to write a rather 'free' part after so much academic and scientific writing!

In my case, I would like to thank to the Music Technology Group from the Universitat Pompeu Fabra in general, and to Xavier Serra in particular, for giving me the chance of being part of the Sound and Music Computing Master, which for me seemed to be quite far not so long ago. It has been a tough year but all this experience will hopefully help in the future.

Many thanks to Justin Salamon and Emilia Gómez for their guidance, support, patience, ideas and help as some points of an endless list of acknowledgments. I know you were quite busy this year, especially you Justin finishing your PhD Thesis, so I really thank every minute devoted to both the Thesis and me. Also related to the Thesis, I would really like to thank Filippo Morelli for his ideas, corrections, food and jokes among other stuff that made all this work easier to deal with.

However, and as it always happens, what really makes something great is the people you meet along the way: SMC fellow students, this would have not been the same without you! I think I would not be able, even in my mother tongue, to express how much I have enjoyed this year with all of you and how glad I am because of having met you all. It does not matter whether we were working in the *Party room* or in the *SMC room*, chilling out in the *SMC flat*, going out around Barcelona or being part of a Thesis experiment, but thanks for all those moments. Wish nothing but the best for all of you!

Also, I would really like to acknowledge 'Fundació la Caixa' for their economic support during this year since, at least from my point of view, it is not only the economic relief you get for one year but also the fact of having been granted by such an institution as 'la Caixa' is.

And last, but not least, I thank my family for all their unconditional support, effort and struggle towards me. I would not be here without you and everything you have given me.

For sure, but not intentionally, I am forgetting people, but I know you know who you are and you will forgive me since, in the end, this is simply either a sheet of paper or a bunch of bytes altogether!

Moltes gràcies a tots!

Contents

List of Figures

List of Tables

'Music in general is looking for something new overall.'
Leslie Edward "Les" Claypool

I Introduction

Not so long ago, Internet did not exist, at least not as it is known and used these days. Music was not as widespread as nowadays and people used other means of finding new music to listen to as, for instance, asking friends, attending concerts, watching TV, reading magazines or listening to the radio, among many others. All of these are still used as sources of new music, but today we mostly rely on the *digital highway* to look for that data.

The music-store clerk [PB03] used to be a key figure in the search and retrieval of new or unknown music: people would go and ask this employee, a sort of a *living music library*, about a tune they heard somewhere. In some cases, the customers could provide some information about, for instance, the band or the genre, but often some kind of humming was actually used to query this music *guru*.

Music Information Retrieval (MIR), knowledge brach in which this Thesis is located, can be defined as "[...] *a field that covers all the research topics involved in the understanding and modeling of music and that use information processing technologies*" [XS13]. In other words, MIR aims at researching in music from different scientific points of view, such as engineering, psychology or physics apart from many other disciplines, but using computational approaches (information retrieval techniques) to deal with the different issues proposed.

Among the different topics MIR deals with, one of them is the aforementioned music recognition expert: *tags* describe music content and are used in music-search systems [PB03, IKMI10], but the issue is that most of the times these concepts are ill defined, since even human beings hardly agree on their

meaning. If these queries could be directly hummed to an *automated* music-store clerk, we might improve our retrieval accuracy. Systems which actually implement this task are known as Query-by-Humming (QBH) systems.

I.1 Motivation and Research Problem

In real life it could happen that, and it is the most usual situation, a music-store clerk might not be able to recognize the tune we are humming. Obviously it could be due to the simple reason that the tune we are producing is not similar enough to the music we are looking for, so it might take the clerk some time or even more information to retrieve the song. However, the most undesirable situation, which is also the most common one, is that this *guru* does not know the music we are looking for, in other words, the song we want is not part of his/her *database*.

Among many drawbacks QBH systems have, the most limiting one is actually this *lack of knowledge*: when we hum a certain tune, the QBH system compares it with its database so that it can later output a similarity result but, if the tune is not present in its corpus, there will never be a correct result.

Most databases in QBH systems are constituted by MIDI transcriptions of the main melody, which are usually obtained through manual annotation [IKMI10, DNS07, RK08]. The need for a manual intervention constitutes a major drawback since it is not possible to transcribe manually all existing songs [SSG12]. This fact clearly limits QBH systems as they are not able to grow by themselves.

In case we implement automatic melody transcription to improve the system, it is important to point out two main issues these methods have: the *fundamental frequency estimation* and the *transcription to notes*. The former process introduces much noise in the system, mainly in polyphonic tunes [CVG+08, IKMI10], since no known method is able to track perfectly the fundamental frequency of a signal; on the other hand, the second process may introduce errors as well since temporal segmentation might not be precise.

On the other hand, melodic similarity is another topic we have to be aware of. As we mentioned before, the music-store clerk would not be able to recognize immediately the tune we are asking for since queries might be sung in different keys, deviated from the original tempo, containing more or less notes than the target tune, among many other errors [TTM12]. Therefore, it is necessary to find a proper similarity measure able to deal with these issues, which actually is quite related to the way the data (queries and

database tunes) is represented in the system.

This Master Thesis aims at researching on melodic similarity for QBH systems avoiding the aforementioned transcription step (that means, working directly on audio-based similarity) for designing an algorithm able to create automatically its own database by extracting the predominant melody from polyphonic tunes. Our approach makes use of the *MELODIA* algorithm[1] [SG12] by Justin Salamon from the Music Technology Group (MTG) in Universitat Pompeu Fabra (Barcelona) to extract the fundamental frequency contour of the main melody from both queries and songs in the database as well as different time-series representations for coding the different contours.

I.2 Goals

The specific goals related to the present Thesis are:

1. Research on the usefulness of melody extraction (more precisely, the *MELODIA* algorithm) in QBH.

2. Study the advantages and disadvantages of different pitch contour representations.

3. Investigate on different distance measures for the comparison of the queries and the database elements.

4. Use automatic music transcription to compare results obtained with audio-based similarity and higher level representations.

5. Implement an approach and discuss/evaluate the obtained results.

I.3 Outline of the Thesis

The rest of the present Thesis is organized as it follows:

1. Chapter II introduces QBH systems, its structure and typical approaches used in this task, aside from music and melodic similarity, which are also important concepts on which these systems rely.

2. Chapter III describes the selected approach for the task.

3. Chapter IV depicts the dataset used for the validation of the approach as well as the measures used for its evaluation.

[1] http://www.justinsalamon.com/melody-extraction.html

4. Chapter V describes and analyzes the obtained results considering the proposed approach and methodology.

5. Chapter VI points out the different conclusions obtained after using the selected approach and proposes some work to be carried out in the future.

'Human beings, who are almost unique in having the ability to learn from the experience of others, are also remarkable for their apparent disinclination to do so.'

Douglas Noël Adams

II State of the Art

This Chapter aims at setting a theoretical background of the topic the present Thesis later develops. For that, an initial section is devoted to explain what music similarity is, and more precisely melodic similarity, since one of the core functions in QBH is to measure how similar two melodies are to determine whether they represent approximately the same or not; after that, the second section is dedicated to define what a QBH is, its relevance in the MIR field and the major challenges this kind of systems involve; finally, a review of previous work done in QBH is presented to the reader.

II.1 Music similarity

What does similarity mean? According to the Oxford English Dictionary [Mur89], two or more things are similar when they "have a resemblance in appearance, character, or quantity, without being identical". Following this idea, inferring that a chair and a sofa are similar since they somehow share their shape (four legs and seat) and their aim (letting people rest) or that a person and a snake are not similar since their appearance is quite different seems to be a straightforward idea.

However, this is not the case in music [BGC+11] since music similarity is a highly-subjective task that does not only depend on the user but also on the context where it is defined [Kel12]: similar music could make reference to different tunes belonging to the same genre, musical period, artist/composer, mood and so on [Wie07]. An example of this in the MIR field is music recommendation [Kel12], where the goal is to retrieve music that is somehow *similar* to what a given user likes.

So, since music similarity is so context-dependent and has no specific definition, which are the most common approaches to this field? Typically, they have been divided in two categories [Wie07, BGC+11]: *metadata-based*, which is the approach that bases its performance in the information that is gathered without considering the physical content of the tune but from high-level labels that describe it as, for instance, lyrics[1], genre or band among many others, and *content-based* approach which refers to the techniques in which the information is directly obtained from the physical signal itself (for example, spectral descriptors, energy or key). Two well-known examples of music-recommendation systems are **Pandora Radio**[2], which is a *content-based* system, and **Last.fm**[3], thought as a *metadata-based* system.

II.1.1 Melodic similarity

As it can be seen, music similarity is a wide topic by itself and its study is out of the scope of the present Thesis. Melodic similarity, which is actually one subtopic of this vast branch of knowledge, is a special type of music similarity in which the aim is finding out whether two or more melodies are similar.

Justifying why melodic similarity is the relevant measure that must be taken into account falls into the definition of QBH systems, which will be addressed later in this Chapter. However, as music similarity is the topic developed here, it makes sense to introduce this concept now.

Melody, as most if not all musicological concepts is quite related to human perception, and basically due to this perceptual connotation, no definition would fulfill two different tunes [PEE+07]. In a practical sense, it can be defined as follows: "[...] *melody is the single (monophonic) pitch sequence that a listener might reproduce if asked to whistle or hum a piece of polyphonic music, and that a listener would recognize as being the 'essence' of that music when heard in comparison*" [PEE+07, SG12]). In other cases, it would be defined as "[...]*one of the most memorable and characteristic features of Western music*" [Typ07] or as "[...]*main perceptive feature of a song*"[QLL11].

Despite all the perceptual implications the previous definitions have, it is quite clear that melody seems to represent quite well a tune by itself, even

[1]Lyrics could also fall into *content-based* since they are actually sung in the audio file but, as voice transcription still has many issues, they still remain as *metadata-based*.

[2]This system is only available in United States, New Zealand and Australia as of February 2013, but its official webpage is http://www.pandora.com/

[3]http://www.last.fm/

if it is only a little excerpt of it, making it reasonable for QBH systems to work with this feature. Also, as it has been pointed out by some authors, the melody developed by the voice or a solo instrument would be the first part somebody would sing rather than bass lines for instance [SSG12, DNS07].

Before finishing this section, it is necessary to define one concept that will later be used: **melodic contour**. By this term we understand how the melody 'evolves' in intervals, that is, whether is goes up, maintains its value or goes down [Typ07].

II.2 What is a QBH system?

As discussed in Chapter I, QBH might be seen as a computer-based version of a music-store clerk [PB03], a person to whom somebody would hum/sing a tune in order to gather more information about it. Another point of view, but actually quite related to the previous one, is defining these systems as an "*automated version of the game 'Name That Tune'* " [DBT+04], game in which somebody would sing/hum a tune expecting the other players to guess it.

These definitions work fine for getting a general idea but, in a more technical scope, QBH would be an example of content-based music similarity system in which the input query is a sung/hummed/whistled tune [DNS07]. A basic diagram of this kind of systems can be found in Figure II.1.

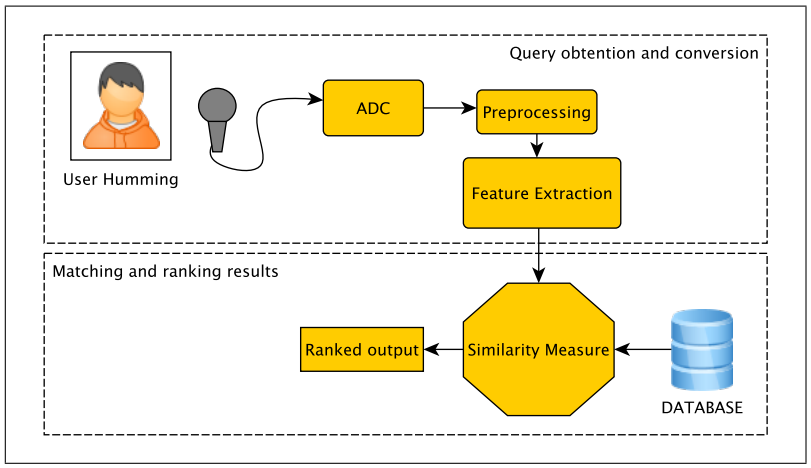

Figure II.1: Diagram of a QBH system.

As the basic diagram in Figure II.1 shows, QBH systems have classically been divided in two subtasks [RK08]: (i) *conversion of the query into a*

certain robust format and (ii) *the comparison of the converted query with the database*. These two subtasks are now explained:

(i) Query formatting

Following what it is shown in Figure II.1, it can be seen that the first step is creating the query by singing/humming/whistling a certain tune and recording it using a microphone. Since the QBH task is meant to be performed during everyday life lacking any specialized equipment, it is reasonable to think that these recordings may have really low quality, forcing QBH systems to work in very noisy environments. Some preprocessing right after having digitalized the query might be performed in order to reduce all the noise in the recorded signal.

Once the signal has been obtained and some basic preprocessing has been applied to it, the next step is the feature extraction. The main question at this stage is: which feature/s would best represent a certain tune? Obviously, this depends on the particular problem we are dealing with but, in the case of QBH, as it was explained in Section II.1.1, melody constitutes the best *descriptor* of a tune itself. Therefore, the Feature Extraction step is, in most of the cases, devoted to the extraction of melodic contours.

Depending on the representation these melodic contours receive, QBH systems are typically divided in two categories [WLL+06, JMC09, Pap10]: (a) *note-based approaches* and (b) *frame-based approaches*. The main difference between these two melodic contour coding approximations is that, while *note-based approaches* require a conversion step of the contour into discrete notes, *frame-based approaches* do not [JMC09]. This difference is shown in Figure II.2

Choosing one of these two approaches carries an important consequence: *note-based approaches* require an extra transcription stage to obtain the notes, step that is carried out by means of automatic transcription. Since automatic transcription is not a completely solved problem, this stage might introduce some noise in the system, which is an undesired effect [JMC09, MD01]. However, even with that disadvantage, most of the research has focused in the *note-based* paradigm [WLL+06, SBY02].

It is also important to comment that, as it is shown in Figure II.2, when talking about *note-based* and *frame-based*, the extracted melodic contour remains neither as a classical score nor as a continuous sequence of pitches, but it has to be coded in a certain format. Some possible formats are:

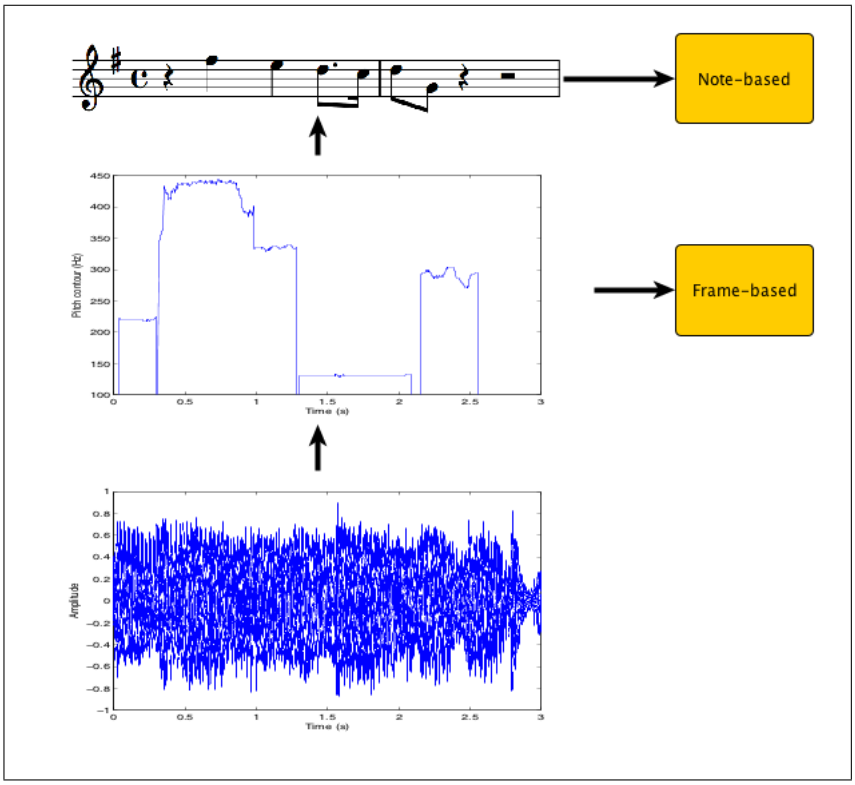

Figure II.2: Difference between note-based and frame-based approaches: melodic contour (middle) is extracted from the audio (bottom) and then, typically, we can either use a *frame-based* representation or transcribe and use a *note-based* approach.

(a) *Frame-based*
Time-series coding has proved to be an important approach in QBH systems without transcription stages: **wavelets** is a possible format [JMC09], **Symbolic Aggregate Approximation** [DNS07], **melody slopes** [ZKT02], **direct f0 contours** or **pitch histograms** [SSG12] are some examples of it.

(b) *Note-based*
Pitches can be coded very easily using the MIDI standard: a possibility is to encode them using the equivalent of the note in the MIDI scale, which is a way of **absolute pitch encoding**, or, instead of that, storing the difference between two consecutive values encoded in this MIDI scale (**relative pitch encoding**), which actually makes the system invariant to key transpositions[4] since the query might be sung in a different key compared to the original tune [DBT+04], a quite common issue in QBH systems

[4]Singing the tune in a different key compared to the original one.

[LRP07]. Another possibility is to encode pitch information, taking as the initial point the absolute pitch code, using **Parsons code**[5] or a variant of it to obtain a melodic contour [GLCS95] with no specific key information, as in relative pitch encoding.

Regarding tempo information, *inter-onset-interval* (**IOI**), which stands for the time difference between two consecutive note onsets, along with *IOI Ratio* (**IOIR**), ratio between two consecutive IOI values, and *Log IOI Ratio* (**LogIOIR**), which is the 2-base logarithm of IOIR, are the most commonly used codifications. Note duration, on the other hand, does not perform so well in QBH [DBT+04].

(ii) Query comparison

Taking a look back to Figure II.1, once the proper descriptors (most of the times, the melodic contour) have been obtained from the query, the system proceeds to the comparison with the database.

The first point here is that the elements in the database must have the same representation as the query, at least when the comparison is going to take place (it is not necessary to store the database in the format we need for the comparisons, but it saves time since there is no need to process the database each time the system is run).

The core in this subtask is the *Similarity Measure* stage for computing the melodic similarity. A initial reference that must be taken into consideration is that the type of measure used is quite conditioned by the representation the system uses for the query: *frame-based* approaches work with a time resolution equivalent to the size of the FFT computation and, therefore, they are slow but accurate; on the other hand, *note-based* approaches work at a note-level, offering faster but less precise results [Pap10].

It is also important to point out that these measures must be quite robust against noise since, as it is explained in Section II.2.3 about the errors in the query production, there is never an exact match between query and database element [DBT+04].

Focusing on particular melodic similarity algorithms, the most common ones used in QBH can be classified in three categories [TTM12,

[5]This coding system, which is actually thought for melodic contours, describes the relation between two consecutive pitches: whether there is an increase ('U'), decrease ('D') or the pitch is maintained ('S'), being the first reference (the first pitch) coded as '*' [NTN10].

KPH+12]: (a) *Sequence Matching*, (b) *Model-based Matching* and (c) *N-grams*.

(a) *Sequence Matching*
Sequence Matching/Alignment is a well-known topic in bioinformatics because of all the research in DNA over the past years, research from which other knowledge areas can take advantage of [LKWL07] as, for instance, MIR.

In melodic similarity, this technique is usually divided in two groups [KPH+12, LPHA10]: *whole sequence matching* and *subsequence matching*. In both cases, the most common approach is to use Dynamic Programming (DP) algorithms, which is an approach for optimizing a problem by dividing it into simpler subproblems [SPB77].

Dynamic Time Warping (DTW) is a very well-known and commonly used *whole sequence matching* algorithm. Basically, the idea behind this method is looking for the best alignment between two sequences which suffer from misalignment and time warps [KPH+12], which means, for instance, that the two sequences might be exactly the same being one 'played' faster than the other one.

Despite *whole sequence matching* has been used in QBH, it seems more suitable to use *subsequence matching* in these systems. Algorithms such as Smith-Waterman or, more recently, SMBGT are examples of this trend which have been applied to QBH [KPH+12]. There is also a possibility of using *whole sequence matching* algorithms by applying a sliding window and retrofitting the system or cutting the database sequences in small pieces and then perform the algorithms. Obviously, these *hybrid* approaches are not free from drawbacks and, for instance, the former approach is quite computationally expensive and the latter one can cause problems when the small pieces have a different size than the query [LPHA10]. As an example, SPRING [LPHA10] constitutes a modification of DTW for suitable *subsequence matching*.

(b) *Model-based Matching*
In this particular approach, the idea is to create models, usually *Hidden Markov Models* (HMM), of the melodies in the database that can be later compared to the queries produced by the users [KPH+12]. Despite being *Model-based Matching* quite similar in performance to *Sequence Matching* (the first one performs slightly worse than the second) [TTM12], the main drawback this approach has is that it requires a previous training stage for each

model, which can be a quite tough task depending on the size of the database [KPH+12].

(c) *N-grams*
This technique has been widely used in string matching and, despite having a lower computational complexity compared to the two previous techniques [TTM12], it has been proved that for QBH, because of the noisy input data, this approach does not perform as well as the previous two do [TTM12, KPH+12].

Finally, as a result of the comparison stage, QBH systems give an output score of the similarity between the query and the elements in the database. It is important to point out that this output is actually a rank (*Ranked output* in Figure II.1) and not a single result as it would be, in principle, expectable since QBH systems retrieve the K most similar tunes[6] to the produced query. As a consequence of this particular output format, the main evaluation criteria are ranking evaluation measures that, in the case of QBH, are typically two:

(a) *Mean Reciprocal Rank (MRR)*
When a user produces a query **Q** related to a certain tune **A**, the QBH system returns a rank of a certain length **N** in which the tune **A** is located at position **r**[7].

The particular *reciprocal rank* for that **A** query is defined as $1/\mathbf{r}$ [DBT+04]. Generalizing this concept, *Mean Reciprocal Rank* stands for the mean value of the *reciprocal ranks* obtained when the system is evaluated with **n** queries. It can be described mathematically as in Equation II.1.

$$\mathrm{MRR} = \frac{1}{\mathbf{n}} \cdot \sum_{i=1}^{\mathbf{n}} \frac{1}{\mathbf{r}\,(\mathbf{Q}_i)} \tag{II.1}$$

(b) *Top-X Hit Rate*
Taking as an initial point an **N**-length rank, this measure considers just two cases: whether the position **r** of the correct result of the search is in the first **X** positions or whether it is not (mathematically, $\mathbf{r}(\mathbf{Q}_i) \leq \mathbf{X}$). By doing this, we can obtain the average of how many times the QBH system retrieves the correct result among the first **X** positions [SSG12].

[6] This number is something fixed in the system that could be varied or not by the user.
[7] This values ranges from **1** to **N**, being **1** the best result (correct result is ranked first).

II.2.1 Database generation

Once these two basic subtasks in QBH systems have been introduced, it is important to point out a basic issue that has not been given much attention in previous work: in most of the research in QBH, the availability of databases is given for granted, that means, there are **annotated transcriptions**[8] (most of the times, MIDI files) [SSG12] for all the existing tunes, transcriptions that can be easily processed to obtain the necessary information. However, this assumption is quite unrealistic since there are not annotated transcriptions for every single tune in the world [IKMI10], constituting a great drawback in QBH research.

As a possible solution to the previous paradigm, there are systems that, instead of relying on these annotations, what they do is that they **store the queries** each user makes to build up a database [SSG12]. The main advantage of this approach is that, although we have audio elements in the database, they are all monophonic[9], so the feature extraction can be 'easily' performed [DBT+04], at least when compared to a polyphonic tune. However, this approach also suffers from a drawback: the first time a user hums a certain tune not present in the database no correct results can be retrieved, since there are no proper matches in the database. This issue is called *cold-start problem* and will always happen in systems with this approach since there is no way to have previous information about every single tune.

A third possibility, which actually would solve the previously exposed limitations, is using a **fully automated approach** for extracting the features from real audio files (polyphonic files) [SSG12, RK08]: the database could contain any song since there is no human factor involved and there would be no *cold-start problem*. In this case, the limitations for feature extraction in polyphonic audio files must be taken into consideration: although melody extraction did not achieve great results in the past [IKMI10, SBY02] and it still remains an unsolved issue, results have improved noticeably [SG12], making this third possibility a proper option to explore.

II.2.2 Relevance of QBH

As introduced in Chapter I, the idea of QBH seems quite an attractive alternative to classic text-based retrieval [PB03, IKMI10] since no musical knowledge is needed [DBT+04].

[8] By **annotated transcriptions** we refer to those produced by humans as opposed to the automatically obtained ones.

[9] Queries are monophonic and we are storing them as the elements of the database since they represent certain tune that might be later needed to retrieve.

Currently, QBH is one of the tasks that is evaluated every year in the *Music Information Retrieval eXchange* (MIREX[10]), in which it appeared for the first time in 2006. Moreover, QBH is not only an academic/research topic, but there are also a number of 'real' applications (both commercial and free) that make use of this music retrieval approach as, for instance, **Musipedia**[11] or **SoundHound**[12], among others.

For finishing, it is also important to point out that QBH is not the only alternative to classic text-based retrieval. In Query-by-Tapping systems, which are also evaluated in MIREX and constitutes one of the search possibilities in **Musipedia**, the queries are created by tapping or clapping the rhythmic section of a tune [HR09]. Also, systems for music recognition such as **Shazam**[13] or **Gracenote's MusicID**[14] constitute a special case of retrieval in which the query is actually the song (useful, for instance, for situations in which we hear a tune coming from a radio and we want to identify it).

II.2.3 Main issues in QBH

As some of the issues researchers in QBH systems have to deal with have already been introduced, it might seem unnecessary to comment them again. However, these issues are the research subtopics in which QBH can be divided, so it is important to remark them to get a clear idea of what QBH comprises.

The main issues in QBH systems are:

(a) Transpositions
As previously commented, when producing the query, singers may go out of tune (either instantaneously or during the whole query) or sing in a different key [LRP07], mainly because of the user's vocal skills and/or a not proper recall of the tune [LML+03].

(b) Tempo deviations
Tempo tends to be different in the query and in the real piece [TTM12, LRP07, IKMI10] and, since tempo has been proved to be as important as pitch information in QBH [KPH+12], it is important to manage those errors for a better search and retrieval.

[10] http://www.music-ir.org/mirex/wiki/MIREX_HOME
[11] http://www.musipedia.org/
[12] http://www.soundhound.com/
[13] http://www.shazam.com/
[14] http://www.gracenote.com/music/recognition/

Some proposed solutions for this issue are forcing the user to sing in a certain tempo by producing clicking sounds when doing the query [IKMI10], time-scaling the target data using fixed values [DBT+04] or using Dynamic Programming (DP) techniques to look for the optimal alignment between sequences [KPH+12].

(c) Databases

The creation of the database is another of the major issues in QBH [RK08]: as it has already been commented, many systems use either **annotated transcriptions** of the tunes or **sung queries** as the database, however these two approaches are limited by the availability of transcriptions and the *cold-start problem* respectively [SSG12].

The most suitable solution for this issue seems to be using a **fully automated approach** for feature extraction, typically the main melodic contour, able to work in polyphonic audio files [SSG12, RK08].

(d) Indexing

Due to the huge size of the databases, a basic search method (for instance, linear search) is not acceptable since it would take too much time to be computed [RK08], getting even worse when the distance function has a higher complexity. In a case like this one, it is necessary to create an indexing structure to speed up this retrieval task.

However, it is also important to point out that some similarity measures used in melodic similarity are not metric, making necessary to use special approaches, able to deal with this drawback, to create the indexing structure [TWT10].

II.3 Previous work

This last section aims at introducing the reader to different real implementations of QBH and relate them to all what has been exposed in the previous sections of this Chapter II.

As it has been commented, *note-based* QBH systems have classically been the ones in which research has focused the most. Actually, the first QBH system, proposed by Ghias et al. in 1995 [GLCS95], is one of those: queries are transcribed (using autocorrelation for pitch tracking) and then converted into melodic contour using a similar technique to Parsons Code; database is comprised of MIDI songs; search is performed using an approximate string matching algorithm (referred to it as 'fuzzy' matching algorithm).

Dannenberg et al. in 2007 [DBT+04] compared different search algorithms for *note-based* QBH systems under the MUSART testbed project, a framework created for automatically comparing various search algorithms and summarize the results.

Ryynänen and Klapuri in 2008 [RK08] proposed another system in which queries are transcribed using frame-wise pitch salience functions, for estimating pitch values, and a musicological model, for estimating note lengths and transitions; database is also comprised of MIDI tunes; search is done using Locality Sensitive Hashing (LSH), which also allows indexing for fast retrieval.

Tsai et al. introduced in 2012 [TTM12] a *note-based* method in which the comparison is performed in the frequency domain: queries are converted to notes using an approach based on *average magnitude difference function* (AMDF), which first of all obtains pitch values for every 1/64 s and then converts it to a MIDI value; melody in target tunes is extracted from MIDI files; queries and target melodies are processed with the *Fast Fourier Transform* (FFT) so that they have the same length and comparison can be performed in a fast way.

Regarding *frame-based* approaches, Duda et al. in 2007 [DNS07] proposed a system in which different features (*Mel-Frequency Cepstrum Coefficients* (MFCCs), Audio Power, f0 contour, Formants and Chroma) are extracted from the melodies (both the ones extracted from the queries and the ones obtained from the tunes in the database) and coded using *Symbolic Aggregate Approximation* (SAX); melodies from polyphonic tunes in the database are extracted using something similar to the "karaoke effect", that is, using panning information to keep the singer's voice; search is performed using edit-distance and N-grams. This approach is particularly important for the development of the present Thesis since it uses the same codification method as the one used here.

Jeon et al. in 2009 [JMC09] published a system in which f0 contours are represented using wavelets; regarding database melodies, this paper uses both a MIDI database approach and a method for real-world music based on pitch extraction from polyphonic music using Constant-Q Transform; search and indexing is performed using K-D Trees, which performs faster than DTW.

Ito et al. in 2010 [IKMI10] came up with a system in which, instead of obtaining a single melodic contour in the elements in the database, multiple F0 candidates are obtained using a variation of the PreFEst algorithm; similarity and search is performed using a basic scoring function.

In Salamon et al. in 2012 [SSG12], melody is extracted, both from queries and database tunes, using a method based on salience function [SG12], available as a *vamp-plugin*[15] called *MELODIA*[16]; comparison is performed using the Q_{max} algorithm [SSA09].

Regarding automatic generation of databases, topic developed in Section II.2.1, as it has been seen from the previously commented approaches, most of the QBH systems that perform automatic extraction of melodic contours from audio files belong to *frame-based* approaches. From a chronological point of view, some of the systems that have addressed this issue are:

(a) Song et al. in 2002 [SBY02] implemented a system in which, at each frame, the spectrum of the excerpt is analyzed in order to find a harmonic structure that leads to a series of probable notes for each frame (sort of an automatic transcription system but without all the precision that those systems are expected to have).

(b) Ryynänen and Klapuri in 2008 [RK08] stated the importance of automatic generation of databases despite the work in this publication is focused on annotated MIDI databases.

(c) Jeon et al. in 2009 [JMC09] carried out a basic experiment on real-work music for assessing the performance of their QBH system in melodic contours not extracted from annotated MIDI files.

(d) Ito et al. in 2010 [IKMI10] published a fully-automated QBH systems in which elements in the databased are analyzed in order to estimate multiple F0 candidates and match them to the f0 contour extracted from the query.

(e) Salamon et al. in 2012 [SSG12] used *MELODIA* for automatically extracting the main melody of polyphonic tunes and compared them to the melody extracted from the monophonic queries.

II.3.1 Current results in QBH

As a final note on QBH systems, it is interesting not only to know about the different approaches people have developed but also to know about the state-of-the-art accuracy results to have, at least, a reference on what it is possible to obtain with the current technology.

The three best results are:

[15] http://www.vamp-plugins.org/
[16] http://www.justinsalamon.com/melody-extraction.html

(i) Note-based Query VS Note-based Database
The best result is credited in MIREX 2012 by Lei Wang, who describes a QBH system with an accuracy of MRR = 0.9689[17].

(ii) Frame-based Query VS Note-based Database
In this case, the best results are credited by Ryynänen and Klapuri in 2008 scoring an accuracy of MRR = 0.885 [RK08].

(iii) Frame-based Query VS Frame-based Database
In this last approach, the best result is credited by Salamon et al. in 2012 with a basic accuracy of MRR = 0.45 using an initial dataset, and a better result of MRR = 0.56 using the initial dataset extended using covers of the songs in the database [SSG12].

It is quite noticeable that there is still much room for improvement, specially in the approaches **(ii)** and **(iii)**, supporting what was exposed before about the fact that QBH is still a not solved issue in the MIR field.

[17] http://nema.lis.illinois.edu/nema_out/mirex2012/results/qbsh/qbsh_task1b_thinkit/summary.html

'A lot of music is mathematics. It's balance.'
Melvin Kaminsky "Mel Brooks"

III Approach

The present Chapter describes the actual approach that, once having studied the background described in Chapter II, has been chosen to tackle our research topic. For that, the first step is to depict the melody extraction algorithm to the reader; the following section is devoted to explain the different time-series representations used; finally, the third section is dedicated to the time-series alignment and its application to melodic similarity.

III.1 Melody extraction

As commented in Chapters I and II, the main melody of a tune is a very appropriate *descriptor* in the task of QBH since it represents, in a general way, the first part of a song a person would try to reproduce[1].

In the present Thesis, the algorithm for extracting this main melody from the different tunes is the *MELODIA* algorithm published in Salamon and Gómez 2012 [SG12], which can be downloaded from `http://mtg.upf.edu/technologies/melodia` as a vamp plug-in[2].

[1] Section II.1.1 in Chapter II discusses this statement more in depth.

[2] Justin Salamon's personal website also provides much information about this algorithm: `http://www.justinsalamon.com/melody-extraction.html`.

Figure III.1: Logo given to the *MELODIA* algorithm for melody extraction in Salamon and Gómez 2012 [SG12].

A basic introduction to this particular approach for predominant melody extraction is now given. *MELODIA* comprises four basic stages, as shown in Figure III.2: *Sinusoid extraction*, *Salience function*, *Pitch contour creation* and *Melody selection*.

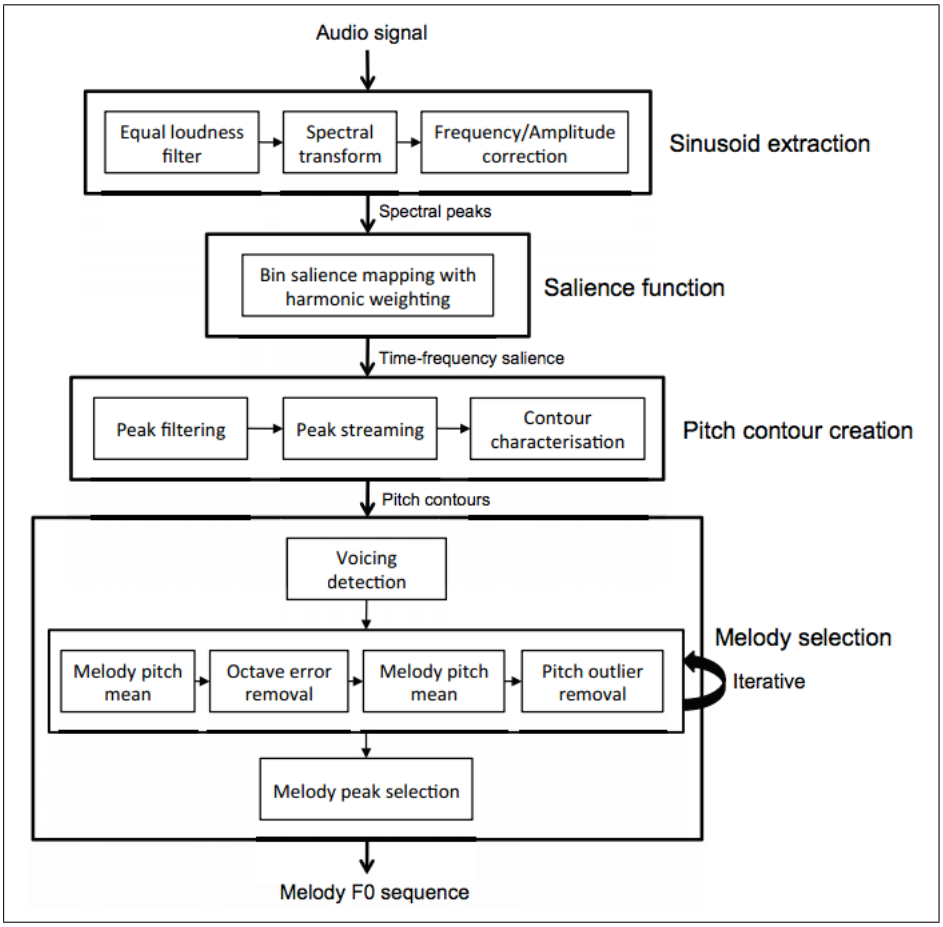

Figure III.2: General scheme of *MELODIA*. Obtained from Salamon and Gómez 2012 [SG12].

1. **Sinusoid extraction**
   In this first step of the algorithm, the aim is to find the frequencies present in the signal at every point in time. The core of this step is the Short-Time Fourier Transform (STFT), which is applied after using an equal loudness filter thought to enhance frequencies human beings are prone to listen to. Also, due to the limited resolution of the STFT, a frequency and amplitude correction step is performed.

   Figure III.3 shows an example of the result of this particular step.

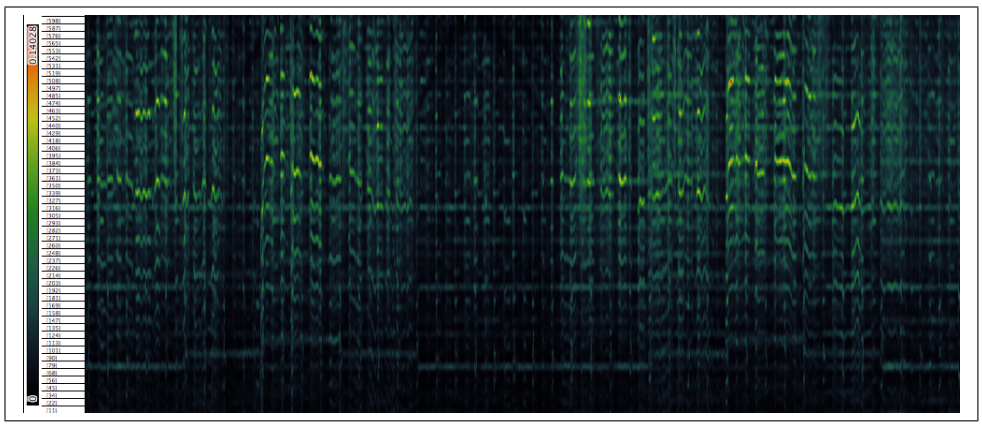

Figure III.3: Example of the *Sinusoid extraction* step from *MELODIA* applied to the chorus section of *Spirit of Radio* from *Rush* (album *Permanent Waves*, 1980).

2. **Salience function**
   Using the result obtained from the *Sinusoid extraction* step, the idea is to create a representation of pitch salience over time covering a range of approximately five octaves, from 55 Hz to 1.76 kHz.

   The *Salience function* is constructed by looking for harmonic series in the sinusoid representation.

   An example of this step can be found in Figure III.4.

3. **Pitch contour creation**
   From the *Salience function* previously obtained and using the main peaks from that function, the possible contours are extracted by using a set of rules based on Auditory Scene Analysis (ASA).

Figure III.5 shows the result of this step applied to the results shown in Figure III.4.

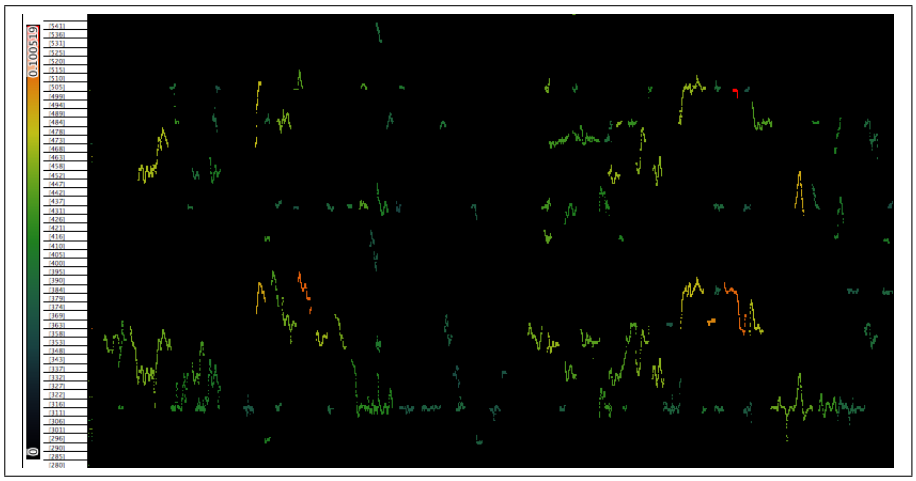

Figure III.4: Example of the *Salience function* step from *MELODIA* applied to the chorus section of *Spirit of Radio* from *Rush* (album *Permanent Waves*, 1980).

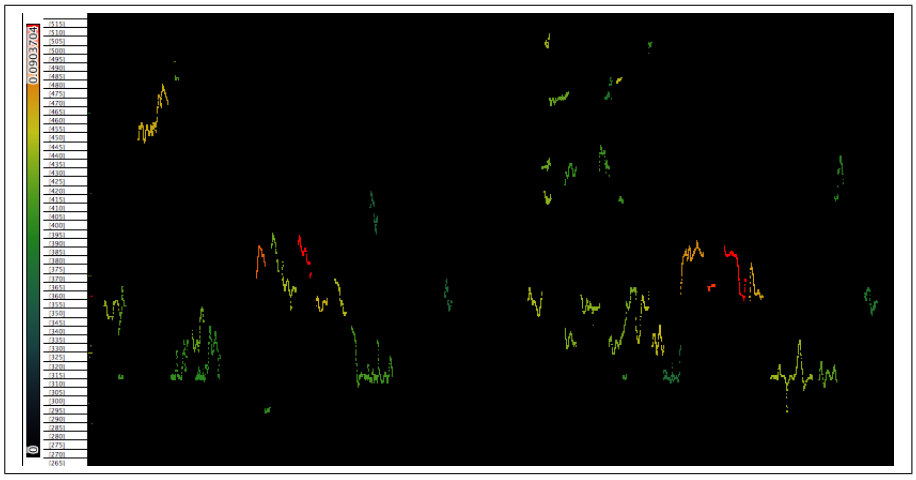

Figure III.5: Example of the *Pitch contour creation* step from *MELODIA* applied to the chorus section of *Spirit of Radio* from *Rush* (album *Permanent Waves*, 1980).

4. **Melody selection**
   Finally, from the possible contours obtained in the previous step, a melody is obtained filtering non-melodic parts of the contours using a set of rules extracted from previous studies of contours belonging to melodies.

   The melody extracted from the previous example can be seen in Figure III.6.

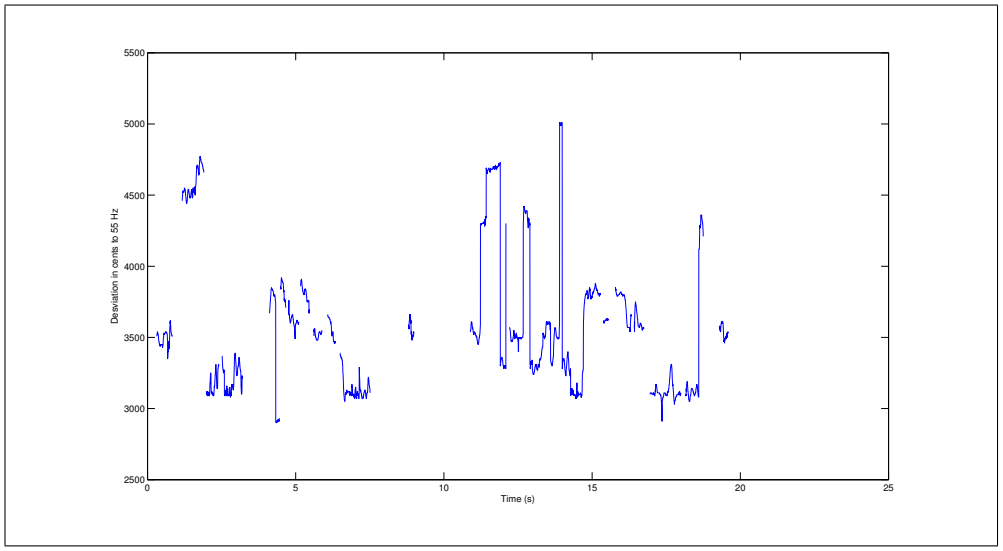

Figure III.6: Example of the *Melody selection* step from *MELODIA* applied to the chorus section of *Spirit of Radio* from *Rush* (album *Permanent Waves*, 1980).

III.2 Time-series representation

The extracted melody has to be somehow coded for its later processing. A clear option for representation would be to code the melody into a note representation but, as introduced in Chapter II, automatic transcription might introduce errors in the system [CVG+08, IKMI10], reason why this Thesis focuses in non-transcription methods.

However, QBH systems based on transcribed music constitute a quite common approach in this field, so apart from non-score representations, an automatic transcription algorithm is used in this case for the comparison between methods.

III.2.1 Non-transcription methods

As it has already been commented, the main aim in the present Thesis is to research on melodic similarity applied to QBH using non-transcribed representations[3]. In Chapter II, more precisely in the *Frame-based* approach for **Query formatting**, a technique called **Symbolic Aggregate Approximation** was introduced to the reader, which is the actual method used for melodic contour representation chosen for the present work.

- **Symbolic Aggregate Approximation**
  Symbolic Aggregate Approximation (from now on, SAX) was introduced by Lin et al. in 2007 [LKWL07][4] as a novel symbolic representation[5] for general time-series analysis, that means, not related to MIR or any music knowledge field.

  The main advantage of using symbolic representations for time-series is that bioinformatics have developed much this particular field (for instance, alignment of DNA sequences) so these representations offer both a solid knowledge base for research and a great catalog of techniques.

  However, and despite symbolic representation has already been considered for time-series analysis, most of the algorithms proposed suffer

[3] Automatic transcription is used for setting a comparison between the two *philosophies*.

[4] Official webpages about SAX maintained by the authors:
http://www.cs.ucr.edu/~eamonn/SAX.htm
http://www.cs.gmu.edu/~jessica/sax.htm

[5] It is important to point out that, as commented, SAX does not have its origin in any music-related topic, but it is thought for generic time-series, so here *symbolic representation* does not refer to a score representation as it might be understood from a MIR-related point of view.

from two major drawbacks that SAX is able to cope with:

i) **Dimensionality reduction:** Algorithms are not able to reduce the data (reduce its dimensionality) and that can cause issues since data mining algorithms do not scale properly with dimensionality.

ii) **Lower bounding issue:** No method is able to calculate a distance in the symbolic domain while providing a lower bounding guarantee.

The steps for coding a certain time-series into the symbolic SAX representation are the ones that follow:

1. **Normalization of the time-series**
   SAX codes the different time-series assuming that they follow a certain statistical distribution[6], originally a Gaussian distribution.

   However, a normalization stage is applied to the sequence so that this Gaussian distribution assumption is closer to the reality. For that, Equation III.1 is applied.

$$x_i' = \frac{x_i - \mu}{\sigma} \quad \text{with } i \in \mathbb{N} \tag{III.1}$$

   where x_i represents each element of the initial time-series, μ is the mean value and σ the standard deviation.

   It is important to say that, in the case of the present Thesis and as it has been commented several times, the sequences to be normalized are melodic contours, which in the case of *MELODIA*, they are groups of frequency values in hertz. In order to avoid the logarithmic behavior of the hertz scale when related to music, the cents scale is used. This conversion is done using Equation III.2.

$$F_{\text{CENTS}} = 1200 \cdot \log_2 \frac{F_{\text{HERTZ}}}{F_{\text{REF}}} \tag{III.2}$$

   being F_{REF} a reference frequency for the transformation in cents, which in the present Thesis is set to 55 Hz since it constitutes the minimum value *MELODIA* is able to track.

[6] The reason why this is like that is explained later.

2. Piecewise Aggregate Approximation

Piecewise Aggregate Approximation (from now on, PAA) is a non-symbolic time-series representation that constitutes part of the SAX coding process. PAA maps the initial signal into **M** equally sized frames, being the value for each frame the average of the values in the initial signal frame. This can be seen in Figure III.7.

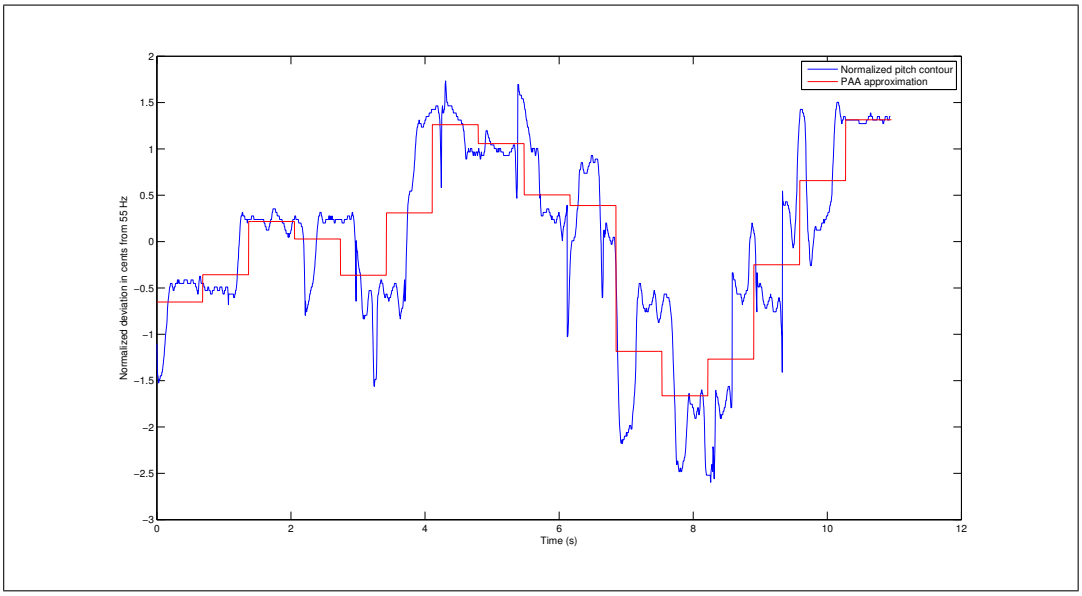

Figure III.7: PAA representation (red) of the melodic contour corresponding to Query 1 in the corpus described in Chapter IV (blue).

In a more mathematically way, PAA approximates a time-series $\mathbf{x}$ of length $\mathbf{n}$ into a vector $\bar{\mathbf{x}} = (\bar{\mathbf{x}}_\mathbf{1}, \bar{\mathbf{x}}_\mathbf{2}, ..., \bar{\mathbf{x}}_\mathbf{M})$ of length $\mathbf{M}$ using Equation III.3.

$$\bar{x}_i = \frac{M}{n} \cdot \sum_{j=\frac{n}{m}(i-1)+1}^{\frac{n}{M}i} x_j \qquad \text{(III.3)}$$

In this stage it can already be seen a dimensionality reduction from $\mathbf{n}$ to $\mathbf{M}$.

3. Symbolic Representation

The idea in this step is, taking the results previously obtained with PAA, quantize the vertical axis in different regions[7] and code each area using a certain symbol. The size **a** of the alphabet, which represents the amount of symbols we have available, is a parameter to be chosen. The fact of assigning the symbol to regions and not to each individual value obtained with PAA (what we called before $\bar{x}_i$) is what eventually leads to a dimensionality reduction[8].

Taking into account that in SAX symbols are expected to be equiprobable, the regions in which we have to quantize the vertical axis have to follow a certain law. For that, SAX assumes that the time-series follows a Gaussian distribution[9] and divides it into several regions that assure this equiprobability, which are mapped as the breakpoints for the different symbol areas in the vertical axis of the pitch contours. This can be seen in Figures III.8 and III.9.

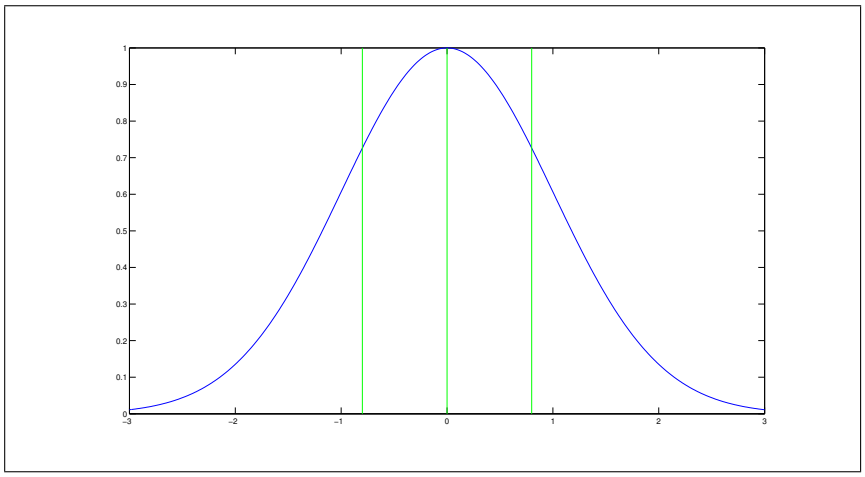

Figure III.8: Breakpoint for the Gaussian distribution (green) for a discretization of 4 regions.

In more formal terms, the group **B** of breakpoints is basically a series of **a** values $B = \beta_1, \beta_2, ..., \beta_{a-1}$ such that $\beta_{\mathbf{i-1}} < \beta_{\mathbf{i}}$ and $\beta_0 = -\infty$ and $\beta_a = +\infty$. Each interval $[\beta_{j-1}, \beta_j)$ represents a certain symbol α_j.

[7] In the previous step we quantized the horizontal or time axis.
[8] The second one since, as it has been commented, PAA already reduces dimensionality.
[9] As introduced in the first step, *Normalization of the time-series*.

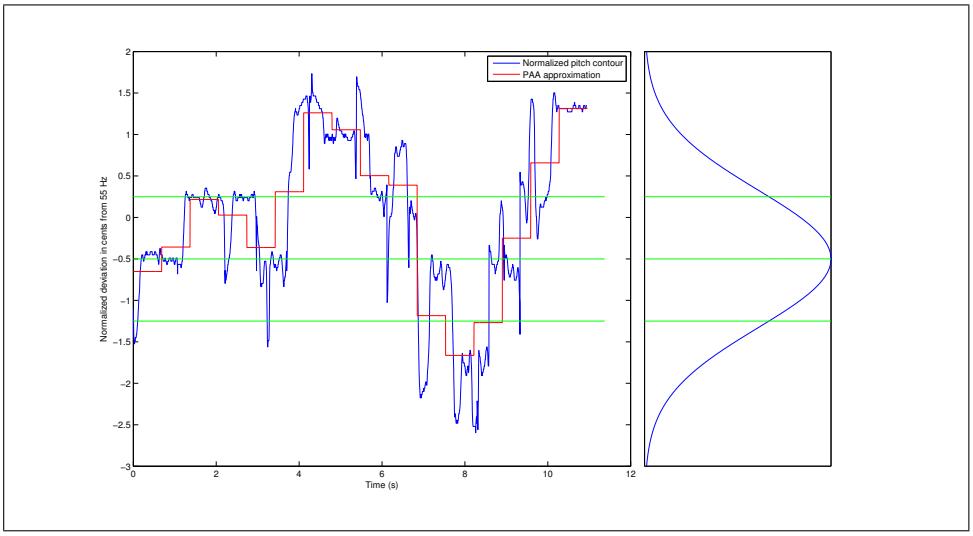

Figure III.9: Example of vertical discretization using 4 regions and Gaussian distribution for Query 1 from the corpus described in Chapter IV.

The conversion of the vector of PAA coefficients $\bar{C}$ into the string $\hat{C}$ is done as shown in Equation III.4.

$$\hat{c}_i = \alpha_j \quad \text{if} \quad \bar{c}_i \in [\beta_{j-1}, \beta_j) \tag{III.4}$$

Graphically, this can be seen in Figure III.10.

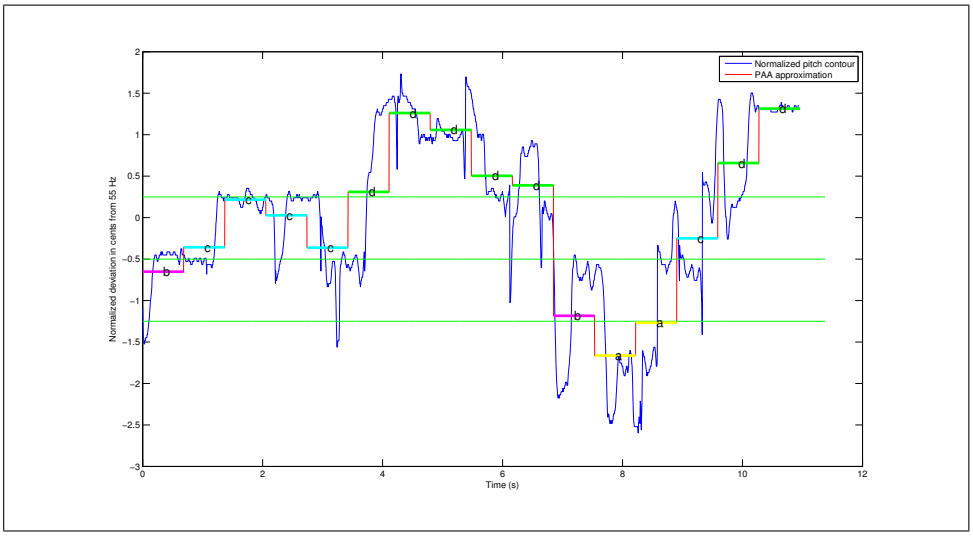

Figure III.10: Example of string encoding using 4 regions and gaussian distribution for Query 1 from the corpus described in Chapter IV.

- **SAX Extensions**
  As it has been shown, SAX has no direct relation with MIR in the sense that it does not use any musical knowledge in the core of its approximation algorithm. This generality, according to all the information found in its official webpages, does not limit the algorithm to non-MIR fields but makes it suitable for a wide range of applications.

  In order to check how a more musical representation might affect to the overall performance, two extensions have been applied to the original method:

  i) Semitone discretization with fixed time divisions
  The first extension performed to the initial SAX algorithm is discarding the codification using a certain statistical distribution: as the sequences to code represent musical information, more precisely pitch contours that represent melodies, it is possible to use certain musical tuning to describe the different values of the signal.

  Based on an *equal-tempered* scale, the vertical axis is divided in semitones using a *semi-fixed* pattern. The reasons for calling this pattern *semi-fixed* are the ones that follow:

  - **Fixed divisions:** The divisions have a fixed size of one semitone (100 cents), starting at 55 Hz, which is the minimum value the *MELODIA* algorithm might retrieve from the audio signal.
  - **Adaptive tuning reference:** Since the reference note in which the melody is produced is not known, a process of alignment between the theoretical grid and the tuning is performed. This process is done by obtaining a histogram of the frequencies of the contour and then, considering the three most prominent peaks, aligning the semitone-division grid with them by minimizing the distance between peaks in the histogram and the theoretical semitone divisions. Figure III.11 shows graphically the mentioned process.

  As a consequence of using this extension, the normalization step performed in SAX is now not needed: the idea of the normalization of the initial sequence is not only forcing it to be more similar to a gaussian distribution but also limiting the possible values of the sequence to a certain range[10]. In this case, as no statistical distribution is used for coding, there is no need to force the sequence to follow a gaussian distribution or to limit the range of

[10] This range is $[-3\sigma, +3\sigma]$, being σ the standard deviation of the sequence.

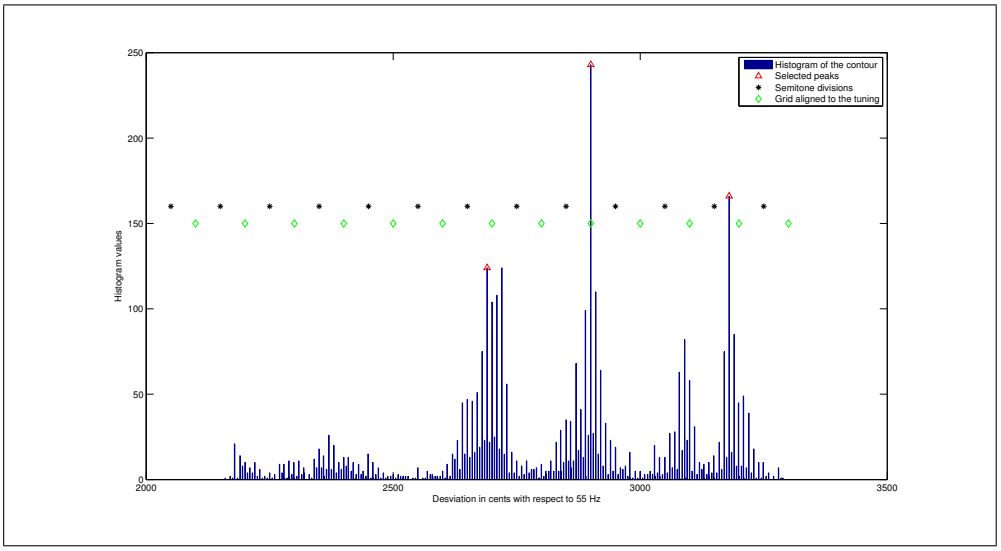

Figure III.11: Example of the adaptive tuning reference for the semitone discretization in the extension to the SAX coding algorithm. The figure represents the histogram of Query 1 from the corpus described in Chapter IV, red triangles (△) point out the three peaks for the alignment, black asterisks (∗) represent the initial grid and the green diamonds (◇) show the grid once it has been aligned to the histogram.

possible values, being the normalization step avoided.

In a general way, what this extension implements is somehow a basic 'automatic music transcription' system: each coded segment represents now a certain frequency value, which might be seen as a music note. Because of that, and as it was mentioned in Chapter II, to avoid key transpositions, a proper solution is using relative pitch coding, i.e. coding the differences between notes (for instance, semitones) rather than the absolute value of the pitch itself.

Figure III.12 shows an example of coding approach.

ii) Semitone discretization with pitch-change transitions

This second extension to the original SAX algorithm is also extension to the already commented modification: taking the previous approach as the initial point, instead of using PAA for doing a fixed-time temporal segmentation, the idea is to dynamically create new segments whenever there is a pitch change in the melodic contour.

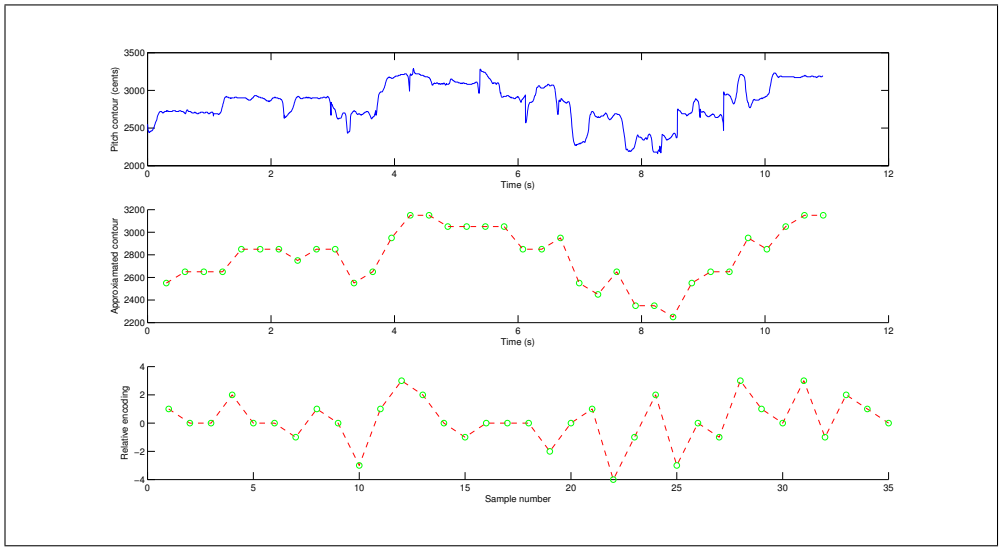

Figure III.12: Example of the semitone discretization with fixed time divisions. Upper image shows pitch contour of Query 1 from the corpus described in Chapter IV, center image represents the approximated contour before using the relative pitch coding and a PAA time approximation of 0.3 seconds and lower image shows the relative pitch coding.

However, it is important to take into consideration that the contours obtained using *MELODIA* might have artifacts as well as fast changes in the pitch values. Looking for pitch changes in the contour might produce several *false* segments that actually should be part of the same segment, so a process for softening the contour is required:

- **Signal smoothing:** An initial smoothing is applied to the signal by applied an average filter using a sliding window.
- **Glitches removal:** Pitch segments shorter than a certain threshold are merged to the previous segment.

Figure III.13 shows an example of the softening process applied to a melody extracted using *MELODIA*.

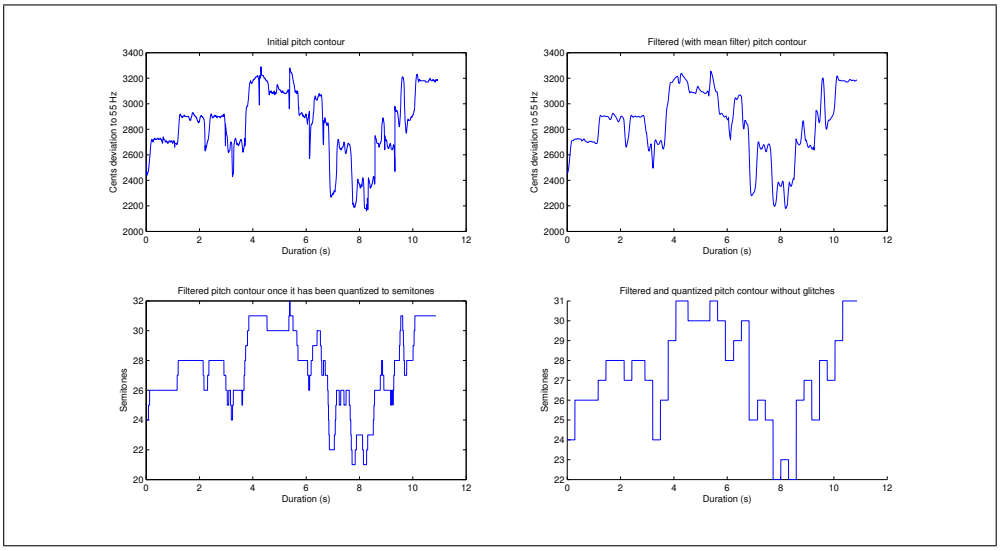

Figure III.13: Example of the softening process applied to the melodic contour of Query 1 from the corpus described in Chapter IV obtained using *MELODIA*. Initial pitch contour (top left), smoothed contour using average filter (top right), smoothed contour quantized to semitones (bottom left) and melodic contour smoothed without glitches (bottom right).

III.2.2 Automatic transcription algorithm

As it was commented in the introduction of the section, despite the present Thesis mainly deals with the idea of using methods that do not require automatic music transcription, it seems interesting to explore this option in order to check whether this approach might improve or not the results.

The approach used is the one in Gómez and Bonada in 2013 [GB13], which has been mainly applied to automatic transcription of flamenco. Figure III.14 shows the general scheme of the algorithm.

The algorithm, as shown in Figure III.14, comprises four stages, which are now introduced to the reader:

1. **Low-level feature extraction**
   The aim in this first stage is dividing the signal into frames[11] and computing the spectrum, energy and fundamental frequency for each of them.

   For the f0 estimation, three different approaches are compared: Time-domain autocorrelation represented by the well-known *yin* algorithm

[11] Overlapping frames of 50 msec, with approximately 5.8 msec between frames onsets, giving a figure of approximately 172 frames per second.

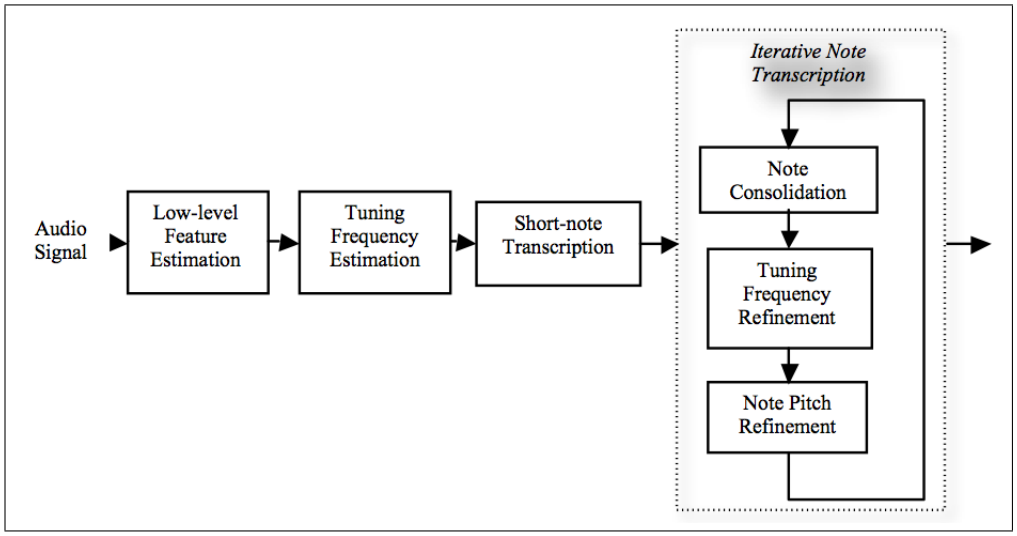

Figure III.14: General scheme for the transcription algorithm reproduced from Gómez and Bonada in 2013 [GB13].

[dCK02]; Frequency-domain harmonic matching, represented by an algorithm, which is based on Two-Way Mismatch (*twm*) [MB94], that tries to match the spectral peaks to a harmonic series [Can98]; frequency-domain autocorrelation using a method called *SAC* (Spectrum Auto-Correlation), presented in this same paper, based on the computation of amplitude correlation in the frequency domain.

2. **Tuning Frequency Estimation**
   In this step an initial estimation of the tuning frequency is performed by computing the maximum of the histogram of f0 deviations from an equal-tempered scale tuned to 440 Hz.

3. **Short-note Transcription**
   As a third step, the audio signal is segmented into short notes, which will be later processed in the last step of the algorithm, by finding the segmentation that maximizes a certain likelihood function.

4. **Iterative Note Transcription**
   In this last step an iterative process takes place with a double aim:

   (a) *Note consolidation*: Notes from the previous process may be merged since they might be the same note. This process only merges notes if the pitch is maintained and a certain stability measure of their connection falls below a certain threshold.

   (b) *Tuning frequency refinement*: Despite tuning frequency had previously been obtained directly using the f0 contour, it is computed

again using the note transcription obtained since it might benefit the estimation.

This iterative process takes place until no more notes can be consolidated.

As an example of the algorithm, Figure III.15 shows the Visualization Tool developed for the algorithm in which a certain transcription is shown.

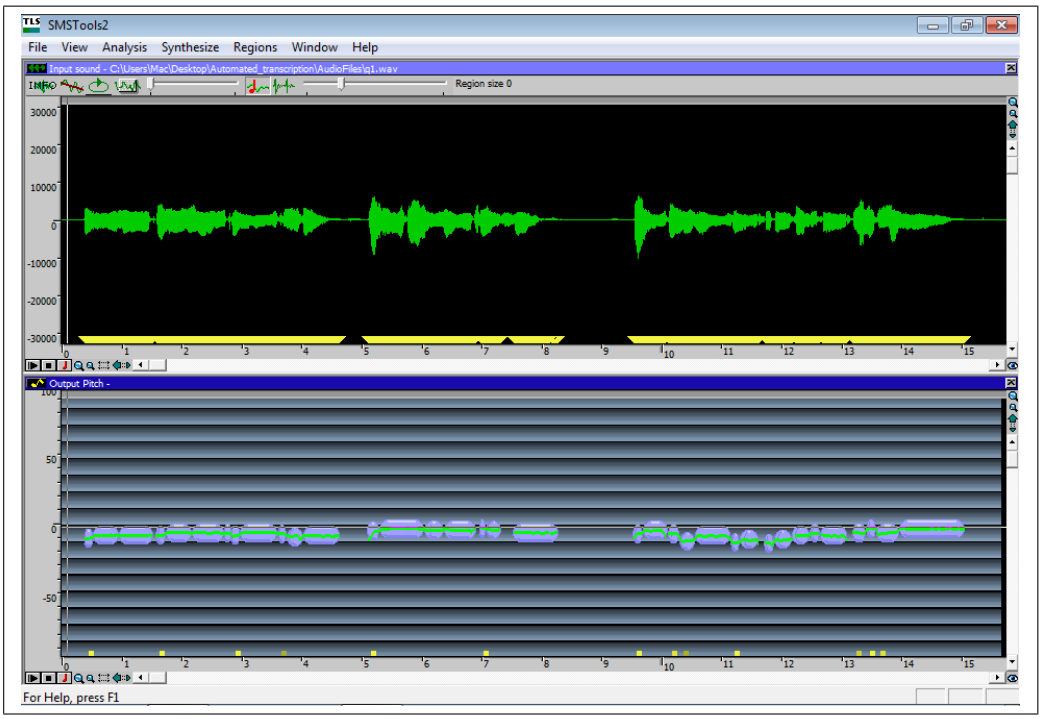

Figure III.15: Example of the graphic tool of the transcription algorithm by Gómez and Bonada in 2013 [GB13]. Upper window represents the initial signal (Query 1 in the corpus described in Chapter IV); lower window represents the extracted f0 contour and the consolidated notes (represented as ovals).

Once the sequences have been transcribed, and as it was pointed out in Chapter II, it is necessary to code them in a certain format able to deal with the common difficulties in QBH[12]. As the main aim in the present Thesis is studying audio-to-audio comparison, the results using music transcription are simply for comparing these two different approaches. Because of that, a single coding format for transcribed sequences is applied in this case.

[12] As a remainder, the reader may check Section II.2.3 in Chapter II.

The coding format used is the *Note Interval Matching* commented in Dannenberg et al. in 2004 [DBT+04]: melodies are represented as a series of tuples with the shape $< Pitch, Rhythm >$ that code the difference/intervals between notes:

1. **Pitch**: Pitch information is coded as **relative pitch encoding**, i.e. the difference in semitones between two consecutive notes.

2. **Rhythm**: Temporal information is coded using the **LogIOIR** representation, i.e. from an initial computation of the difference between onset intervals of the notes (**IOI**), the ratio between those values is obtained (**IOIR**) and 2-base logarithm is applied to that result. Since the possibility of having many different values is high, the **LogIOIR** results are quantized to the nearest integer ranging from -2 to +2.

Once all the note intervals have been coded using this representation, the whole melody is represented as the combination of all the tuples:

$$< Tuple_0, Tuple_1, Tuple_2, ..., Tuple_N >$$

Figure III.16 shows an example of the commented coding approach.

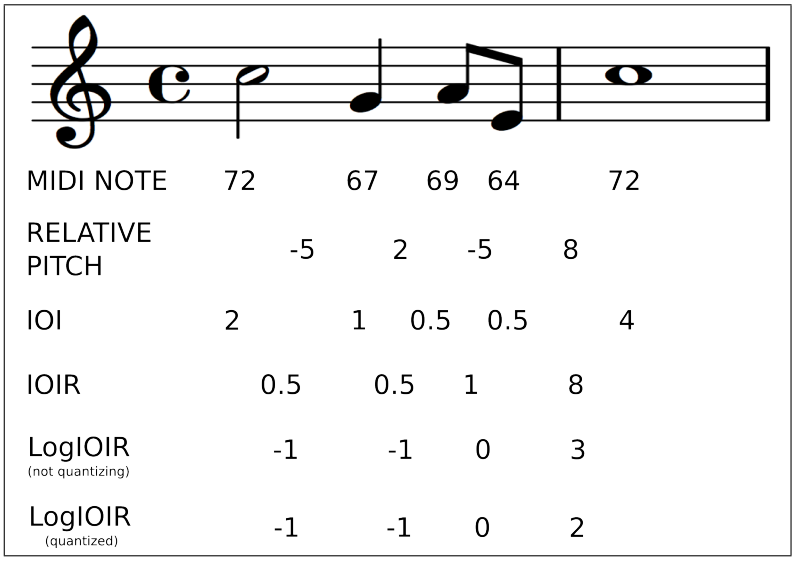

Figure III.16: Example of the *Note Interval Matching* described in Dannenberg et al. in 2004 [DBT+04].

The melody in Figure III.16 would be coded as the following list of tuples:

$$<< -5, -1 >, < 2, -1 >, < -5, 0 >, < 8, 2 >>$$

III.3 Similarity measurement and sequence alignment

Once the pitch contours have been coded using a certain method, the next step is comparing them. For that, in basic terms, two methods are used in this case: one of them is SAX itself since it does not only describe a way of representing time-series but also a way of comparing them; the second one is a local alignment algorithm called Smith-Waterman.

III.3.1 SAX similarity

SAX proposes a distance measure thought for equal length[13] time-series coded with the same amount of levels.

The method is based on comparing element by element the two representations, which is possible since we have the same amount of SAX coefficients in both series, using a previously obtained lookup table that summarizes the distance between symbols so that the general result is calculated fast. This lookup table is defined as shown in Table III.i[14].

-	**a**	**b**	**c**
a	*dist*(a,a)	*dist*(b,a)	*dist*(c,a)
b	*dist*(a,b)	*dist*(b,b)	*dist*(c,b)
c	*dist*(a,c)	*dist*(b,c)	*dist*(c,c)

Table III.i: Example of lookup table in SAX similarity for 3 symbols.

The *dist* function is the one that defines the distance between individual symbols, which in SAX is defined as in Equation III.5.

$$dist\,(r,c) = \begin{cases} 0 & \text{if } |r-c| \leq 1 \\ \beta_{MAX(r,c)-1} - \beta_{MIN(r,c)} & \text{otherwise} \end{cases} \tag{III.5}$$

where β_i are the breakpoints previously defined for the amplitude discretization in the SAX codification.

In case two sequences differ significantly in length before being coded, SAX proposes obtaining subsequences of the long sequences and comparing

[13] The series may not have the same length before the coding, but they have to result in the same number of SAX coefficients when coding.

[14] For an example of an alphabet of 3 symbols.

every single subsequence in one of the time-series to all the other subsequences in the other time-series. Figure III.17 shows how these subsequences are obtained.

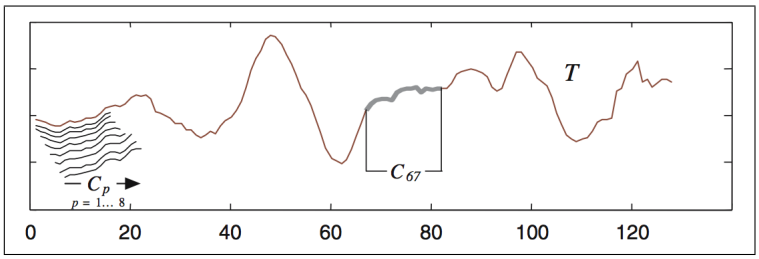

Figure III.17: Method for comparing sequences with different length once they have been coded with SAX. Figure extracted from Lin et al. in 2007 [LKWL07].

III.3.2 Smith-Waterman

The previous SAX similarity approach has a clear limitation in the sense that if two sequences are temporally misaligned, the method will fail. As commented in Chapter II[15], some time warping methods have been classically used for dealing with these misalignments for different tasks, being QBH one of them.

The Smith-Waterman algorithm, introduced in Chapter II, is a subsequence matching method originally published by T. F. Smith and M. S. Waterman in 1981 [SW81] for DNA sequences alignment. The idea behind this algorithm is to find the most similar subsequences in larger sequences, even if there are time warps.

For doing that, this algorithm generates an $(n+1) \times (m+1)$ similarity matrix (being n and m the lengths of the sequences A and B to be compared), notated as $\mathbf{H}$, which is filled following Equation III.6.

$$H(i,j) = \begin{cases} 0 & \text{if } i = 0 \\ 0 & \text{if } j = 0 \\ max \begin{cases} 0 \\ \textbf{Match/Mismatch} \\ \textbf{Insertion} \\ \textbf{Deletion} \end{cases} & \text{otherwise} \end{cases} \tag{III.6}$$

[15] As a summary, the reader may refer to Section II.2.3 in the commented Chapter.

where *max* is a function that returns the maximum of the four values and **Match/Mismatch**, **Insertion**, **Deletion** are defined as it follows:

- **Match/Mismatch:** This part of the algorithm tries to match the two sequences assuming that there is no temporal misalignment between them. Mathematically, this can be define as shown in Equation III.7

$$H\left(i-1, j-1\right) + w\left(A_i, B_j\right) \tag{III.7}$$

  were **w** is a function that gives a positive value (**+Match**) when A_i and B_j[16] are the same and a negative value (**-Mismatch**) otherwise.

- **Insertion:** This is a penalty score that is given in case there is a missing value (a temporal misalignment) in one of the sequences, more precisely, the one that leads the horizontal evolution of the matrix. Mathematically, it is represented using Equation III.8.

$$H\left(i, j-1\right) + \text{Insertion\_Cost} \tag{III.8}$$

- **Deletion:** The concept is the same as **Insertion** but now the temporal misalignment is produced by the sequence set in the vertical axis. Mathematically, it is represented using Equation III.9.

$$H\left(i-1, j\right) + \text{Deletion\_Cost} \tag{III.9}$$

Figure III.18 shows graphically how the **H** similarity matrix is constructed.

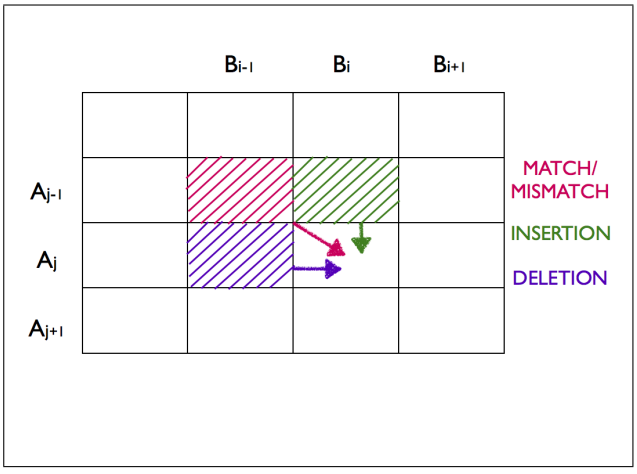

Figure III.18: Construction of the similarity matrix of the Smith-Waterman algorithm.

[16] A_i and B_j refer to each element of sequences A and B respectively.

As an example to check the result of the commented algorithm, Figure III.19 shows the **H** similarity matrix obtained after aligning two sequences: one of the sequences is "*Hello all you boys and girls*"[17] and the other one is "*all you*" (exact excerpt of the first sequence). Analyzing the **H** matrix, it can be seen that, in several regions, the algorithm give some positive similarity results between the strings but, despite those partial results, there is a clear match between sequence and subsequence.

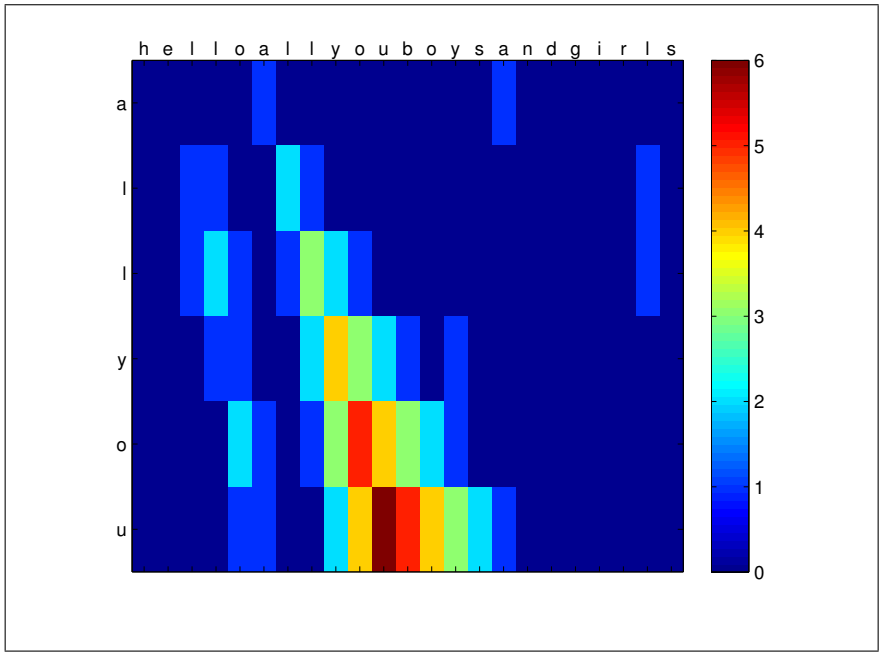

Figure III.19: Example of subsequence matching using Smith-Waterman.

[17] Excerpt of the song *Frizzle Fry* by the band *Primus* from the album *Frizzle Fry*.

'I have not failed. I've just found 10,000 ways that won't work.'

Thomas Alva Edison

IV Evaluation Methodology

The idea for evaluating the proposed approach is to use a similar methodology as the one used in Salamon et al. in 2012 [SSG12] so that results can be directly compared since, as it has been already mentioned, this work constitutes the previous step to the work presented in the present Thesis.

These results are obtained using an already existing QBH database, which already comprises both a set of sung/hummed queries and a real-world music collection acting as possible target songs, being no need of creating a new dataset, and standard measures typically employed in QBH evaluation. Both elements are described in this Chapter.

IV.1 Evaluation dataset

The dataset has two parts: a *music collection*, which constitutes the target songs a user might want to retrieve, and a *query corpus*, which is a set of melodies sung/hummed by some users. This dataset can be found in `http://mtg.upf.edu/download/datasets/MTG-QBH`.

IV.1.1 Music collection

The music collection is comprised of 2125 commercial songs originally used in Serrà et al. in 2011 [SKSA11] for the evaluation of a cover detection algorithm. This collection is divided in 523 song sets, being each set a group of

Figure IV.1: Logo given to the dataset created for Salamon et al. in 2012 [SSG12].

versions of the same song (on average, there are 4.06 songs per set, ranging from 2 to 18), having each tune an average length of 3.6 minutes (ranging from 0.5 to 8 minutes). In terms of genre, the elements in the collection correspond to a variety of genres: pop/rock (1226 songs), electronic (209), jazz/blues (196), world music (165), classical music (133) and miscellaneous (196).

The commented database is actually divided in two, as in Salamon et al. 2012 [SSG12], for the experimentation:

- **Music collection with only the original/canonical songs:** In this subgroup of the initial collection, the idea is that each query sung to the system has one and only one equivalent in the database, which should be the original/canonical[1] version of the target song. As 33 of the initial 523 sets before introduced do not contain the original/canonical version of the tunes they represent, they are removed, resulting in a subset of 481 canonical songs.

- **Whole music collection:** On the other hand, the whole music collection explained before (canonical + covers) can be used as our database of possible target songs. An increase of the size of the music collection may mean a worse performance overall but, since the songs that are being added are actually covers of the canonical songs, there are now more possible targets, which might also increase the overall results.

IV.1.2 Query corpus

The group of queries was recorded for the experimentation in Salamon et al. 2012 [SSG12] using a basic laptop microphone and no post-processing to simulate a realistic situation. A total of 118 queries were recorded by 17

[1]By either original or canonical we understand the song as it was published by the artist who composed/played the song.

users (9 female and 8 male), whose musical knowledge ranged from none to amateur musicians, choosing songs from the canonical subset (481 songs) of the music collection described before. Each user recorded an average of 6.8 queries, being the 1 minimum amount of queries recorded by a certain user and 11 the maximum. On average, the length of the queries is 26.8 seconds, ranging from 11 seconds to 98 seconds.

IV.1.3 Evaluation subsets

From the commented evaluation dataset, four subsets are done for the practical evaluation of the QBH system:

1. **9 Queries:** This subset is thought for the SAX similarity measure[2] in which sequences must have a similar length before being coded. These 9 queries are chosen since they represent the same canonical songs and, therefore, they seem suitable to be used for checking the commented approach when having sequences with similar length. This corpus subset is described in Table IV.i.

Song	Artist	Query Number	Duration (s)
More Than Words	Extreme	6	47
		13	23
		105	37
Over The Rainbow	Judy Garland	45	46
		51	27
		85	24
Sweet Home Alabama	Lynyrd Skynyrd	46	17
		55	25
		113	16

Table IV.i: Subset of the Query corpus for the SAX similarity method.

2. **10 queries/100 songs:** With this subset, an initial adjustment of the different parameters for optimizing the results can be done without the large time consumption of the other subsets. The 10 chosen queries are the initial one (1-10) and, regarding the songs, 10 of them correspond to the 10 songs that represent the first queries and, the rest, are the first 90 songs from the canonical dataset. In Chapter V it is labelled as **10x100**.

[2]Described in Section III.2.1 of Chapter III.

3. **All queries/canonical dataset:** This subset provides the performance of the QBH system when the music collection comprises only canonical songs, i.e. only one example of each song. In Chapter V it is labelled as **118x481**.

4. **All queries/whole music collection:** As shown in Salamon et al. 2012 [SSG12], including covers might improve the performance of the system. With this subset (actually, the whole evaluation dataset), this improvement is evaluated when comparing results to the ones obtained with the previous subset. In Chapter V it is labelled as **118x2133**.

An initial evaluation of the SAX similarity for similar length sequences is carried out using an excerpt of the Query corpus described in Chapter IV since, as commented, they have a similar length: as some queries represent the same canonical songs, an evaluation of the algorithm can be done to check whether SAX is able to find the 'repeated' songs.

IV.2 Evaluation measures

As it was introduced in Chapter II, the output of a QBH system is not a single result of similarity between the query and an element of the music collection but a vector of K[3] most similar elements of the music collection to the query ranked. Because of that, QBH systems are evaluated using raking measures, particularly two:

(a) *Mean Reciprocal Rank (MRR)*
When a user produces a query **Q** related to a certain tune **A**, the QBH system returns a rank of a certain length **N** in which the tune **A** is located at position **r**[4].

The particular *reciprocal rank* for that **A** query is defined as $1/\mathbf{r}$ [DBT+04]. Generalizing this concept, *Mean Reciprocal Rank* stands for the mean value of the *reciprocal ranks* obtained when the system is evaluated with **n** queries. It can be described mathematically as in Equation IV.1.

$$\mathrm{MRR} = \frac{1}{\mathbf{n}} \cdot \sum_{i=1}^{\mathbf{n}} \frac{1}{\mathbf{r}\left(\mathbf{Q}_i\right)} \tag{IV.1}$$

[3] This number is something fixed in the system that could be varied or not by the user.
[4] This values ranges from **1** to **N**, being **1** the best result (correct result is ranked first).

(b) *Top-X Hit Rate*
Taking as an initial point an **N**-length rank, this measure considers just two cases: whether the position **r** of the correct result of the search is in the first **X** positions or whether it is not (mathematically, $\mathbf{r}(\mathbf{Q}_i) \leq \mathbf{X}$). By doing this, we can obtain the average of how many times the QBH system retrieves the correct result among the first **X** positions [SSG12].

'Science never solves a problem without creating ten more.'

George Bernard Shaw

V Results and Discussion

Once the selected approach (Chapter III) and the evaluation methodology (Chapter IV) have been stablished and described, the results obtained are now introduced to the reader.

In the present Chapter, the first results commented are the ones related to the study of the statistical distribution of the query corpus for the SAX coding algorithm. After that, the results from the similarity measures are shown: first of all, results from the SAX similarity measure are commented, followed by the ones obtained using the Smith-Waterman algorithm and, finally, the figures obtained using the automatic music transcription approach. For a better comprehension of the results, a section after the description of the results for each approach is devoted to a summary of the results. Finally, in the last section of the Chapter, a discussion about the results is shown.

V.1 Statistical distribution study

The idea in this first section is to perform a statistical analysis of query corpus described in Chapter IV since the SAX algorithm, introduced to the reader in Chapter III, is based on the premise of coding sequences using the statistical distribution followed by them.

The original description of SAX, as a general way of sequence coding, uses a gaussian distribution as the statistical distribution to be used [LKWL07]. Moreover, Duda et al. in 2007 [DNS07] made an initial evaluation of SAX for coding different audio and music descriptors for QBH purposes, being one of them *audio fundamental frequency*, descriptor quite related to the melodic

contours in the present Thesis, which they stated followed an distribution for their dataset.

As the dataset used in this case is different from the one Duda et al. in 2007 [DNS07], this study has to be done. For that, the main tool used in this case is the *probability plot*. A *probability plot* is basically a graph that compares the distribution followed by the data to a theoretical distribution. An example of it can be seen in Figure V.1.

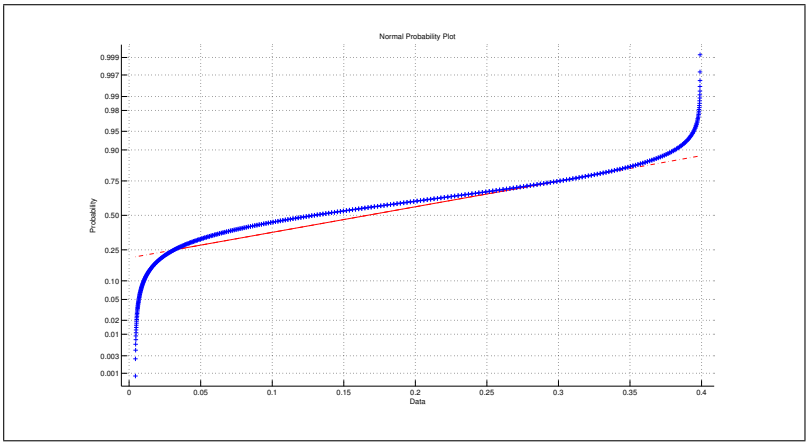

Figure V.1: Example of a *probability plot*: comparison between the statistical distribution of real sequence (blue) and the tendency of a theoretical gaussian distribution (red).

Using this tool it is quite easy to check visually whether a sequence follows a certain statistical distribution since the more similar the two curves are, the more probable it is that both two represent the same distribution.

However, with this simple approach, it is only possible to say that a sequence follows a certain distribution qualitatively, but not quantitatively. For that, it is possible to obtain the r^2 coefficient that represents the how well the two sequence get adjusted, which ranges from **0** (worst adjustment) to **1** (best adjustment).

An important point to take into consideration is the initial format of the sequence: as commented in Chapter III, *MELODIA* retrieves melodic contours using a hertz scale, which has a logarithmic scale, but in order to avoid that behavior, the cents scale is used[1]. As a direct consequence of this transformation in the representation format, the statistical distribution followed by the data might be affected.

[1]The transformation from hertz into cents is shown in Equation III.2.

Once the tool for analyzing the distribution has been defined, it is now important to set a methodology for the statistical evaluation of the sequences. Two possibilities may arise in this case: the first one would be analyzing all sequences together, assuming statistical independence among them, and the second one would be analyzing each sequence separately from the rest.

The first approach seems to be quite appropriate but it has a great drawback: the **Central Limit Theorem** (CLT) [Ric01] states that the statistical distribution of the sum S_n of a large amount of statistically independent variables $V_1, V_2, ..., V_n$ can be approximated to a Gaussian distribution. Therefore, merging all sequences to a single one and analyzing it may always result in a Gaussian distribution independently of the individual distributions of the single sequences.

On the other hand, the second approach assures that every sequence is analyzed and CLT is avoided, being then the most suitable approach to use.

As it has already been commented, SAX originally approximates sequences using a Gaussian distribution [LKWL07] but the study by Duda et al. in 2007 [DNS07] states that their dataset works better using an Exponential. In order to find out a possible distribution that represents the query corpus introduced in Chapter IV, it seems appropriate to initially evaluate these two distributions since they have already been used with SAX[2]. However, other distributions are also tested in order to find the best possible candidate.

Figure V.2 shows graphically and example of the difference between using a cents (columns 3 and 4) or hertz representation (columns 1, 2 and 5). Also, the difference between the two main distributions commented can be seen: Gaussian distribution in columns 1 to 3 and Exponential distribution in columns 4 and 5.

From what can be seen in Figure V.2, it seems that the best fitting is using a cents representation against a Gaussian distribution. However, and as it was commented before, this visual analysis is not enough, and a more analytical approach is needed: each query from the corpus in Chapter IV is adjusted to a certain distribution, being the previously commented r^2 coefficient obtained. Table V.i shows the median and average values of the factor[3].

[2]It is important to point out that Gaussian distributions, as opposed to Exponential, have not been used for QBH purposes.

[3]Average and median refers to, once the r^2 coefficient has been obtained for each query, average and median are computed. This is done to get a general idea of the adjustment for all queries at the same time while avoiding CLT.

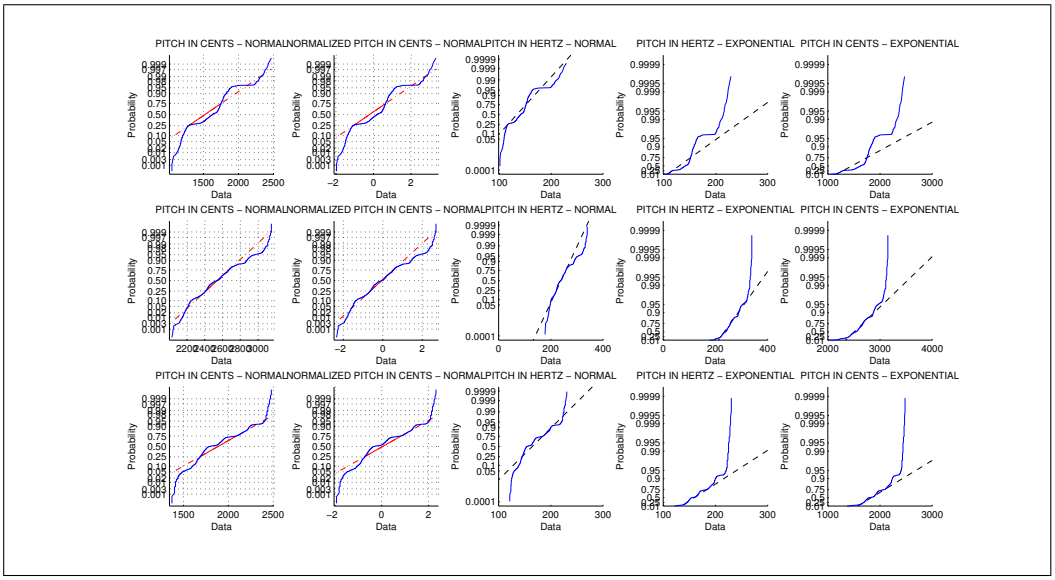

Figure V.2: *Probability plots* of Queries 1, 2 and 3 (one per row) from the corpus in Chapter IV. Each column differs from the others in the query representation or in the statistical distribution to compare with: **1)** query in cents against a Gaussian distribution; **2)** normalized query in cents against a Gaussian distribution; **3)** query in hertz against a Gaussian distribution; **4)** query in hertz against an Exponential distribution; **5)** query in cents against an Exponential distribution.

Statistical distribution	r^2 **fitting value**	
	Average	**Median**
Gaussian	0.9714	0.9758
Exponential	0.8664	0.8815
Arcsine	0.9347	0.9415
Cauchy	0.2493	0.2397
Anglit	0.9710	0.9780
Cosine	0.9726	0.9782

Table V.i: Results of the adjustment of the Query corpus introduced in Chapter IV to different statistical distributions when represented using cents.

Analyzing the results in Table V.i, it can be clearly seen that representing the sequence using cents against a Cosine distribution gives the best adjustment score, both on average and median terms. However, Gaussian distribution also gives great result in the adjustment and, since SAX is originally thought for using a Gaussian distribution, it seems a good idea to use it instead of the others.

V.2 Similarity results

Once it has been checked that a certain distribution, which in this case turns out to be Gaussian, is able to represent the dataset introduced in Chapter IV, the similarity measures can be obtained.

In the two following sections, the results from the **SAX similarity measure** and the **Smith-Waterman algorithm** are presented to the reader[4], being then followed by a last section where the results using **automatic music transcription** are presented as well[5].

V.2.1 SAX similarity

As commented in Chapter III, SAX not only proposes a fast similarity measure for sequences with a similar length before being coded (which results in sequences of the same length after the coding process), but also a procedure for using it with sequences that differ significantly in length using a sliding window.

For the evaluation of this approach, the best subset from the ones described in Chapter IV is **9 Queries** because of two reasons:

1. All elements in the subset have a similar length, which is appropriate for the evaluation of the SAX similarity technique when not using the sliding window approach.

2. The duration of the elements is around 30 seconds (on average), which is useful for the evaluation of the SAX similarity technique with sliding window, not only in terms of the similarity itself but also in terms of computational cost, since the amount of resulting windows may not be large.

The results from applying the SAX similarity technique for similar length sequences to the mentioned subset can be checked in Figure V.3, where it can be seen that the best MRR score is, roughly, 0.09.

On the other hand, processing the same subset but with the sliding window approach, there is a significant different in the results, which can be checked in Figure V.4. In this case, the size of the window is 23 seconds and the best MRR result is, roughly, 0.18. It is also important to point out that, despite it cannot be seen in the graphs, this approach is really time consuming.

[4]For the description of these approach, reader may refer to Section III.2.1 in Chapter III.

[5]This approach is introduced in Section III.2.2 in Chapter III.

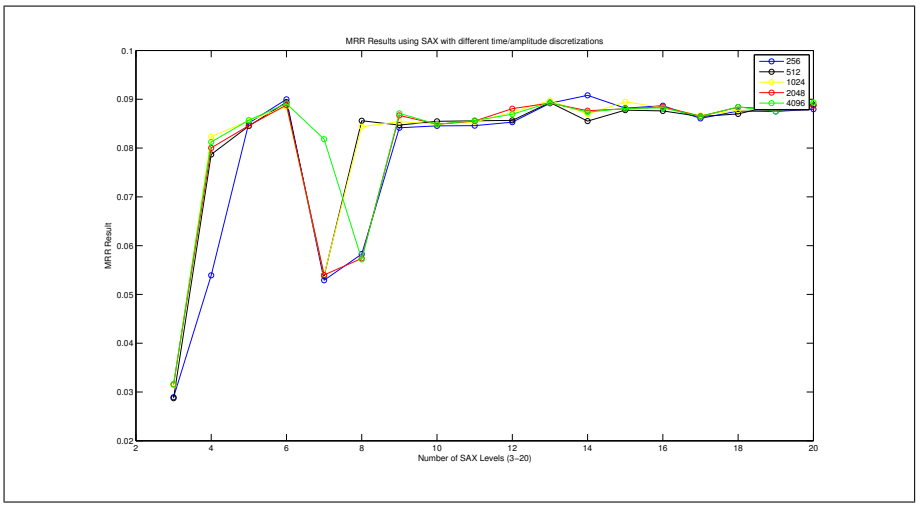

Figure V.3: Results obtained using the SAX similarity measure for similar length sequences for the subset described in Table IV.i: MRR is displayed in the Y-axis while the number of levels in the SAX coding is shown in the X-axis. Each line depicts a different temporal discretization (PAA approximation).

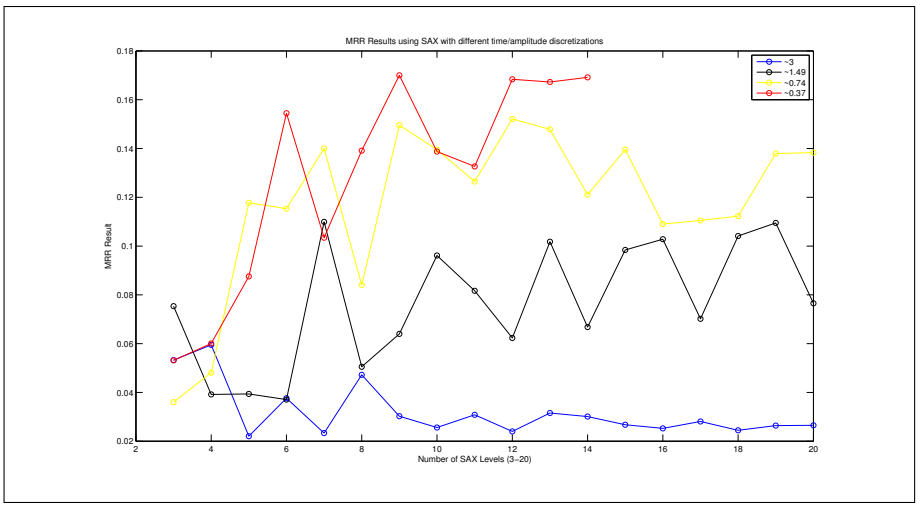

Figure V.4: Results obtained using the SAX similarity measure with sliding window for the subset described in Table IV.i: MRR is displayed in the Y-axis while the number of levels in the SAX coding is shown in the X-axis. Each line depicts a different temporal discretization: the equivalence between a symbol and the time, in seconds, it represents (PAA approximation). The size of the window is 23 seconds.

V.2.2 Smith-Waterman

The Smith-Waterman algorithm, as commented in Chapters II and III, is thought for dealing with temporal misalignments as well as being able to find subsequences inside larger sequences, characteristics that make it quite suitable for QBH. However, this characteristics have their cost and, in this case, it results in the introduction of four new parameters, which have already been commented in Chapter III.

These four parameters in the Smith-Waterman algorithm are typically set by 'Trial and error'. The four different configurations tried in the present Thesis are shown in Table V.ii.

Configuration	**Weights**			
	MATCH	**MISMATCH**	**INSERTION**	**DELETION**
CONF 1	1	-0.5	-0.5	-0.5
CONF 2	1	-1	-0.5	-0.5
CONF 3	1	-1	-1	-1
CONF 4	1	-0.5	-1	-1

Table V.ii: Smith-Waterman tested configurations.

Using the commented configurations for the Smith-Waterman algorithm it is possible to apply the four approaches described in Chapter III for time-series representation, both the **non-transcription methods** and the **automatic transcription**one.

a) SAX

In terms of coding, and as commented in Chapter III, SAX requires two parameters two be set, which have a 'Trial and error' behavior, just as the Smith-Waterman parameters. These two parameters have been given the following values:

- **Time discretization**: 0.3, 0.5, 0.8, 1 and 2 seconds.
- **Amplitude discretization**: 3, 4, 6, 8, 12, 16 and 20 levels.

Table V.iii shows the MRR scores obtained for the subsets 2 to 4[6] described in Chapter IV together with the parameters used for obtaining them.

[6] As commented in Chapter IV, subset 1, which is the **9 Queries** one, was simply thought for the evaluation of the SAX similarity measure.

Evaluation subset	MRR	Time discretization (s)	Amplitude discretization (levels)	Smith-Waterman configuration
10x100	0.2613	0.5	4	**CONF 2**
118x481	0.0398	0.3	6	**CONF 1**
118x2133	0.0890	0.3	6	**CONF 2**

Table V.iii: MRR results obtained when coding with the approach **SAX** and using **Smith-Waterman** for the similarity/alignment.

Table V.iv shows the results for the Top-X hit rate measure using the previously commented subsets.

Evaluation Subset	Top-X hit rate (%)			
	1	**3**	**5**	**10**
10x100	20	30	30	40
118x481	2.54	4.24	6.98	9.32
118x2133	5.08	9.32	11.02	15.25

Table V.iv: Top-X hit rate results obtained when coding with the approach **SAX** and using **Smith-Waterman** for the similarity/alignment.

b) **Semitone discretization with fixed time divisions**

In this case there is only one parameter to set, also having a 'Trial and error' behavior. This parameters received the following values:

- **Time discretization**: 0.3, 0.5, 0.8, 1 and 2 seconds.

Table V.v shows, as in the case before, the results obtained for the MRR measure using the same subsets 2 to 4 described in Chapter IV together with the settings used for obtaining them.

Evaluation subset	MRR	Time discretization (s)	Smith-Waterman configuration
10x100	0.1774	0.3	**CONF 3**
118x481	0.0649	0.3	**CONF 1**
118x2133	0.0705	0.3	**CONF 1**

Table V.v: MRR results obtained when coding with the approach **Semitone discretization with fixed time divisions** and using **Smith-Waterman** for the similarity/alignment.

Table V.vi shows the results for the Top-X hit rate measure.

Evaluation Subset	Top-X hit rate (%)			
	1	**3**	**5**	**10**
10x100	10	20	30	40
118x481	2.54	5.93	10.17	16.10
118x2133	3.39	5.93	9.32	13.56

Table V.vi: Top-X hit rate results obtained when coding with the approach **Semitone discretization with fixed time divisions** and using **Smith-Waterman** for the similarity/alignment.

c) Semitone discretization with pitch-change transitions

This approach, as commented in Chapter III, also has two parameters to be set, which are given the following values:

- **Signal smoothing filter**: 70, 140, 218 and 290 seconds.
- **Glitches removal filter**: 70, 140, 218 and 290 seconds.

Table V.vii shows the results obtained for the MRR measure using the same subsets 2 to 4 described in Chapter IV together with the values of the parameters used for obtaining them.

Evaluation subset	MRR	Signal smoothing filter (ms)	Glitches removal filter (ms)	Smith-Waterman configuration
10x100	0.4019	290	140	**CONF 4**
118x481	0.1077	140	70	**CONF 1**
118x2133	0.1490	218	70	**CONF 1**

Table V.vii: MRR results obtained when coding with the approach **Semitone discretization with pitch-change transitions** and using **Smith-Waterman** for the similarity/alignment.

Table V.viii shows the results obtained for the Top-X hit rate measure.

Evaluation Subset	Top-X hit rate (%)			
	1	**3**	**5**	**10**
10x100	30	50	50	60
118x481	6.78	11.86	15.25	18.64
118x2133	11.86	14.41	16.10	25.42

Table V.viii: Top-X hit rate results obtained when coding with the approach **Semitone discretization with pitch-change transitions** and using **Smith-Waterman** for the similarity/alignment.

V.2.3 Automatic transcription

As it has been commented in Chapters II and III, automatic transcription tends to be the most used approach in the QBH task, reason why a basic implementation is carried out in the present Thesis in order to check results with the other commented approaches.

This approach also uses Smith-Waterman for the similarity/alignment measure and the different possible configurations are the same as the ones used previously (Table V.ii). The results obtained can be checked in Table V.ix.

Evaluation subset	MRR	Smith-Waterman configuration
10x100	0.1412	**CONF 3**
118x481	0.0380	**CONF 1**
118x2133	0.0499	**CONF 4**

Table V.ix: MRR results obtained when coding with **Automatic transcription** and using **Smith-Waterman** for the similarity/alignment.

Table V.x introduces the results obtained for the Top-X hit rate measure to the reader.

Evaluation Subset	Top-X hit rate (%)			
	1	**3**	**5**	**10**
10x100	10	10	10	20
118x481	1.69	3.39	5.93	7.63
118x2133	3.39	5.08	6.78	11.02

Table V.x: Top-X hit rate results obtained when coding with the approach **Automatic transcription** and using **Smith-Waterman** for the similarity/alignment.

V.3 Results summary

For a better comprehension and later comparison, Table V.xi summarizes the MRR results obtained using the different approaches applied. It is important to point out that these results only make reference to the scores obtained using the Smith-Waterman similarity/alignment algorithm and not to the ones obtained as a result of using the SAX similarity measure due to the low scores obtained and computational cost involved.

Evaluation subset	SAX	Semitones + Fixed Time	Semitones + Transitions	Automatic Transcription
10x100	0.2613	0.1774	0.4019	0.1412
118x481	0.0398	0.0649	0.1077	0.0380
118x2133	0.0890	0.0705	0.1409	0.0499

Table V.xi: Summary of the MRR results obtained.

Table V.xii summarizes the different scores obtained, just as Table V.xi, but using the Top-X hit rate measure instead of the MRR.

Approach	Evaluation Subset	Top-X hit rate (%)			
		1	**3**	**5**	**10**
SAX	**10x100**	20	30	30	40
	118x481	2.54	4.24	6.98	9.32
	118x2133	5.08	9.32	11.02	15.25
Semitones + Fixed Time	**10x100**	10	20	30	40
	118x481	2.54	5.93	10.17	16.10
	118x2133	3.39	5.93	9.32	13.56
Semitones + Transitions	**10x100**	30	50	50	60
	118x481	6.78	11.86	15.25	18.64
	118x2133	11.86	14.41	16.10	25.42
Automatic Transcription	**10x100**	10	10	10	20
	118x481	1.69	3.39	5.93	7.63
	118x2133	3.39	5.08	6.78	11.02

Table V.xii: Summary of the Top-X hit rate results obtained.

V.4 Results discussion

Once all the results for the different approaches have been obtained, it is time to discuss them by comparing to results obtained in related work and finding out the reason why the results in the present Thesis may be better or worse than others by analyzing the performance of the selected approaches with the proposed dataset.

V.4.1 Results comparison

As introduced in Chapter II, Salamon et al. in 2012 [SSG12], using the same dataset as the one used for the present Thesis, scored an MRR of **0.45** for the case of the whole query corpus against the canonical song collection (subset **118x481**) and an MRR of **0.56** for the whole dataset (subset **118x2133**).

These results are quite better than the ones obtained by any approach in the present Thesis for the same subsets: as it can be seen in Table V.xi, for the subset with the canonical collection, the best result is **0.1077** and, in the case of the whole collection, it turns out to be **0.1409** so, taking into consideration that the dataset is the same, it seems reasonable to think that

the selected approach/es might be the main issue for the low scores.

On the other hand, and as commented in Chapter II, Duda et al. in 2007 [DNS07] also used SAX for QBH, obtaining the results shown in Table V.xiii.

Query production	Size of Query corpus	Size of Music collection	MRR score
Humming	150	200	0.0362
Singing	130	200	0.0585

Table V.xiii: MRR results obtained by Duda et al. in 2007 [DNS07].

Despite the results shown in Table V.xiii may not be directly compared to the results obtained in the present Thesis (summarized in Table V.xi) since datasets are not the same, qualitatively it can be said that no significant difference can be pointed out.

V.4.2 Results analysis

Before analyzing in depth the actual results, it may be interesting guessing, in a broad sense, why some approaches obtained better results than others. From now on and until specified, this explanation only refers to the different time-series coding approaches not based on music transcription, which requires an explanation apart from the others.

Checking the results in Tables V.xi and V.xii, it can be clearly seen that scores, in general, improve with each extension performed to the initial SAX algorithm[7].

On the other hand, analyzing theoretically the two extensions performed to the SAX initial algorithm, it may be easy to infer that, somehow, a really basic transcription algorithm is being implemented: from the initial approach (SAX), in which the coding is based on statistical properties, a first approach based on a constant time segmentation and an *equal-tempered* scale for the pitch axis is implemented for then moving to a second and last approach based on dynamic temporal segmentation whenever there is a pitch change.

[7] There is only one exception in the **10x100** subset where SAX obtains a better MRR result than the semitone discretization with fixed time divisions approach extension.

Therefore, from what has been commented, it seems that the more the approach tends to music transcription the better the results get, which might justify the general tendency in QBH to work with automatic transcription approaches [WLL+06, SBY02], as it has been commented in Chapter II.

However, why is it like that? It turns quite difficult to give a general answer for all other authors' work but it is possible to clarify why this happens in the scope of the present Thesis.

It is easy to see that in SAX, the temporal discretization might smooth too much, or even remove, parts of the sequence since, in the end, what it is being done, is taking an excerpt of the signal and averaging it. But also, considering the vertical discretization, typically SAX codes sequences using a small amount of possible values, being 20 the highest amount used in the present Thesis. Therefore, a SAX symbol might not only represent a huge amount of data in a temporal scope, but also when referring to the vertical axis.

Pitch sequences may, and usually do, cover a huge range of octaves, therefore coding with a small amount of levels with SAX makes the coded sequence too generic: for instance, a pitch contour that ranges 5 octaves comprises, theoretically, 80 different notes but, using the maximum amount of levels in SAX, which as commented before is 20 for the present Thesis, makes that each SAX level represents up to 4 notes.

Following the same idea, if an excerpt of the pitch contour evolves in 4 consecutive notes for a long time, never stable but varying through those 4, SAX will only output one symbol, which is the same coded sequence it would be obtained with an excerpt that keeps for the same amount of time one of those 4 notes without changing pitch. Therefore, it is possible that sequences that are different among them before being coded with SAX produce the same SAX sequence when coded.

The described ambiguity does not seem appropriate for QBH: many sequences of the song collection might have a similar SAX code and, therefore, when compared to a certain query, all of them might be possible candidates.

The two extensions proposed to SAX in Chapter III try to somehow solve this issue: the first extension tries to avoid that SAX represents much pitch information in one single level and the second extension, keeping the improvement of the first one, deals with the temporal information loss. The reason why these two approaches improve results is because, and as said before, these sequences are more unique in the sense that it is more difficult to obtain the same codification starting from different pitch contours, something that could not be asserted before, and therefore there are more

chances that a query resembles to just one element of the music collection rather than to many of them.

The following example describes this fact: using the **first extension to SAX** as the coding approach, Query 1, which represents *Mother Nature's Son* (The Beatles), is confronted against the whole canonical song collection. The result is that the correct equivalent is ranked in the 18th position, with a Smith-Waterman similarity matrix shown in Figure V.5.

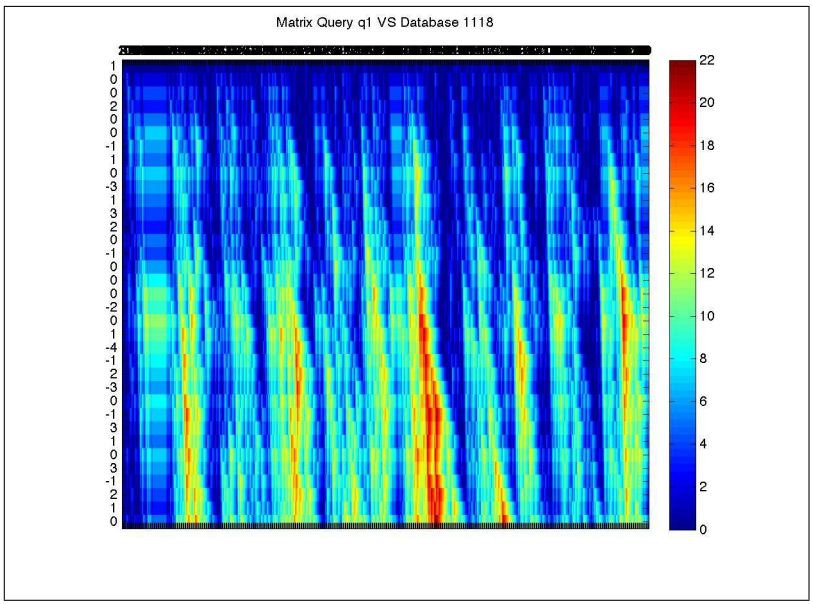

Figure V.5: Smith-Waterman similarity matrix of Query 1 (Mother Nature's Son - The Beatles) and Song 1118 (Mother Nature's Son - The Beatles) from the canonical dataset.

It can be clearly seen in the similarity matrix in Figure V.5 that the chosen approach is able to find a clear correspondency (actually, several of them) between the query and the song. However, if there is such a clear alignment, what is it happening with the song that finished the first in the rank? Figure V.6 shows its similarity matrix.

Figure V.6 corresponds to *Scarborough Fair* (Simon & Garfunkel) and the approach is finding an important 'similar' section where it should not. The same can be said for the element ranked second, whose similarity matrix is shown in Figure V.7, and for all the other songs until the 18th position.

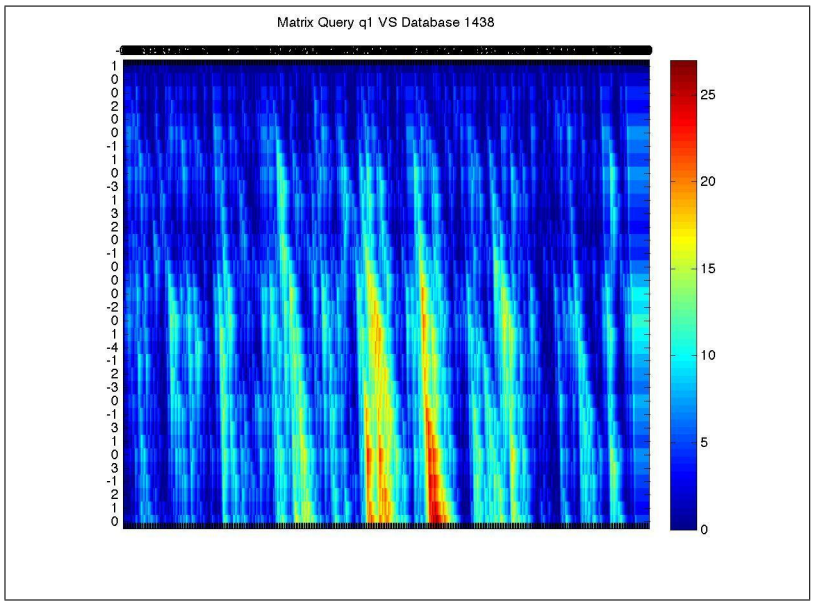

Figure V.6: Smith-Waterman similarity matrix of Query 1 (Mother Nature's Son - The Beatles) and Song 1438 (Scarborough Fair - Simon & Garfunkel) from the canonical dataset.

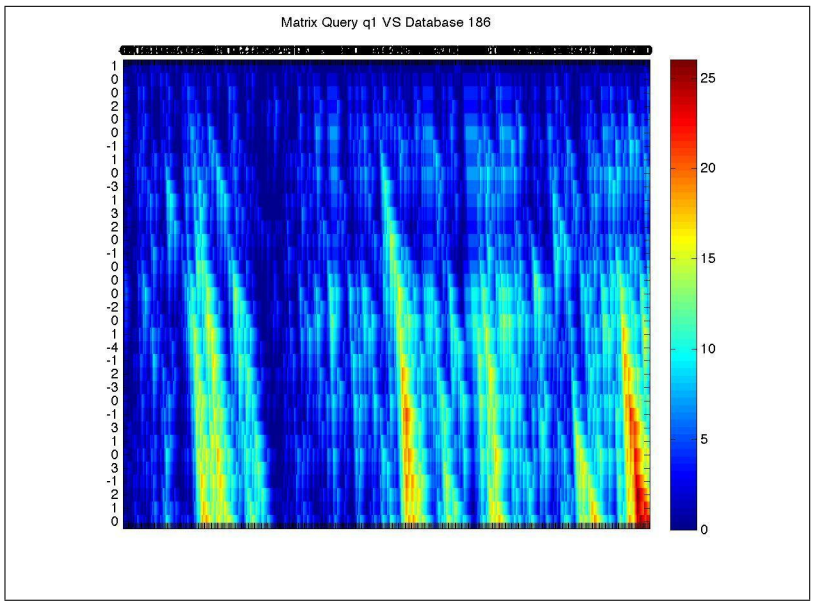

Figure V.7: Smith-Waterman similarity matrix of Query 1 (Mother Nature's Son - The Beatles) and Song 186 (Black Bird - The Beatles) from the canonical dataset.

Therefore, the idea to keep from what has been exposed is that the low scores obtained are due to the fact that SAX creates too generic sequences, at least for a QBH task, and a different representation able to deal with that problem would, presumably, improve results.

Before finishing with SAX it is also important to say why the SAX similarity measures are discarded in favor of Smith-Waterman: the fact that SAX implements a sliding window approach for comparing sequences that differ significantly in length might be useful for QBH but, as commented previously, this method has two important drawbacks:

a) Low scoring: Results obtained showed a very low MRR score, especially considering that the size of the dataset consisted of only 9 elements.

b) Computational cost: The computation time the algorithm requires is quite high and that is not suitable for QBH.

Finally, and to end up with this Discussion section, it is time to think about the results of the automatic music transcription approach.

The first point to mention is that the results obtained might contradict what has just been exposed: the results for the automatic music transcription approach (Tables V.ix and V.x) are considerably low compared to the other approaches (Tables V.xi and V.xii), especially taking into consideration what has just been explained.

The idea is that the work in the present Thesis, as commented in Chapter I, focuses on non-transcription approaches for QBH, reason why the transcription approach is more a proof of concept rather than an actual research task in this work, leading then to the implementation of a very simple approach.

With the aim of implementing this simple approach, a very basic coding algorithm for the automatically transcribed sequence was used (described in Chapter III) and, because of that, the issue found here is the same as in the previous cases: the sequences obtained are too generic (for instance, only 5 different values are used for coding rhythmic information) and, therefore, the system gets confused.

'We can only see a short distance ahead, but we can see plenty there that needs to be done.'

Alan Mathison Turing

VI Conclusions and Future work

This last Chapter acts as the closure unit for the work presented during the development of the present Thesis, first of all summarizing the outcomes and conclusions obtained and later proposing some possible future work to be carried out.

VI.1 Conclusions

Despite the conclusions that can be extracted from the present Thesis have already been commented, in a subtle way, in Chapter V, especially in the Discussion section (Section V.4) from the commented Chapter, it is important to remark them in a formal format, not only for a proper closure of the work done but also to give a clear idea of the reach of the present Thesis.

The first outcome is an important difference with the work by Duda et al. in 2007 in [DNS07] and the **statistical distribution used for the SAX coding approach**: in the mentioned work, it was said that an Exponential distribution could describe the statistical distribution of the pitch contours of their dataset, but in the present Thesis it has been proved that an Exponential distribution is not able to represent our dataset, being for instance a Gaussian distribution a better option.

A second conclusion to be commented is the fact that the **SAX similarity methods**, both the one for sequences with a similar length and the one based on sliding window, do not seem to be valid for the present work:

i) The **SAX similarity method for sequences with a similar length** did not seem to be appropriate from the first moment due to the fact that, in QBH, the sequences to be compared differ significantly in length. However, and as commented in Chapter V, it was implemented and evaluated, corroborating the initial idea with the low MRR results obtained.

ii) On the other hand, the **SAX similarity method based on sliding window** seemed more promising since it was already conceived for sequences differing in length. However, its implementation and later evaluation, despite improving the MRR scores compared to the previous approach, the results obtained were still low and, moreover, the algorithm turned out to be really slow, and therefore not suitable for QBH.

As commented, the **SAX similarity tools** do not seem to be valid in the scope of the present Thesis, but it may be too early to say that SAX might not be suitable for QBH since this assertion requires a deeper study.

A third idea to get from the work carried out is that the **time-series representations used are too generic**: when comparing a query to the whole music collection, as shown in Chapter V, the alignment/similarity method finds many 'clear' correspondences not only among different elements of the collection but also inside each element. Clearly, the more unique the representation turns to be (from **SAX**, way generic, to its **first** and **second extension**, which give more 'unique' sequences), the better the score gets, as previously shown. However, still the results are quite poor, reflecting the fact that the representations used do not produce sequences different enough from each other, at least for QBH.

VI.2 Future work

Considering both the Discussion section (Section V.4) in Chapter V and the Conclusions explained in the previous section, some future work lines may be proposed not only to solve the limitations, or at least some of them, found during the development of the Thesis but also to expand the reach of this work.

An initial point to take into consideration is the **statistical distribution** used for coding sequences with SAX: it was shown that, for the dataset described in Chapter IV, the best statistical distribution that described those sequences is not a **Gaussian distribution**, despite scoring a pretty good adjustment parameter. A **Cosine distribution** seems to get a better adjustment score, so it could be a proper option to try. Also, a **Gaussian Mixture Model** (GMM) might describe the sequences since each lobe of

the GMM would represent a certain note sung/hummed in the query.

Another possibility to research on is to consider **other time-series representations** different to SAX since, as it has been already commented, the sequences created with this approach do not seem to be unique enough for this case. Other representations that do not require the automatic music transcription step such as the ones introduced in Chapter II might be used with the dataset used in this work to check whether there is an improvement on the results.

Related to **time-series representation**, more complex representations for **automatic music transcription** than the one used in the present work might improve the overall score. The work by Urbano et al. in 2012 [ULMSC12] obtained the best results in MIREX 2012 by representing melodic contours, which were obtained directly from score information, as spline curves. Also, the use of perception models based on the studies by Narmour and the Implication/Realization model, which has already been used in QBH [GAdM05], might improve the performance in the similarity task.

Despite the **Smith-Waterman** algorithm seems to be a proper algorithm for QBH for both performing subsequence matching and dealing with time warps, other approaches as the ones introduces in Chapter II might be implemented for checking whether the overall QBH performance can be improved by changing the similarity/alignment method.

Speed issues, not considered in the present Thesis unless the execution time turned out to be excessive, are quite important in QBH since this task is thought for real-time interaction with a user. **Indexing structures** might be an important point to develop in the future so that a query is not compared to every single element in the music collection but only to possible candidates. Hashing, as in the work by Ryynänen and Klapuri in 2008 [RK08], might be a possibility to take into consideration.

Leaving apart all the mechanisms for time-series coding and comparison, it seems also interesting to study the **way queries are produced by people and its influence on the QBH system performance**, which is something that has not been considered in the present work either. In the work by Duda et al. in 2007 [DNS07] it can be seen a clear distinction between queries produced by humming and singing and the difference in performance between the two. Also, the work by Salamon et al. in 2012 [SSG12] includes a study of the influence of the tuning of the queries on the system performance, study that has been extended by Filippo Morelli in his Master Thesis developed in parallel to the present work.

Finally, a last proposal is to extend the presented work for **more datasets** to check the behavior of the approach independently of the data to process. For instance, a proper candidate might be the dataset by Duda et al. in 2007 [DNS07] so that results could be compared not only qualitatively as done in Chapter V but also quantitatively.

Bibliography

[BGC+11] Costas Boletsis, Anna Gratsani, Dimitra Chasanidou, Ionnis Karydis, and Kermanidis Kermanidis. Comparative analysis of content-based and context-based similarity on musical data. In Lazaros Iliadis, Ilias Maglogiannis, and Harris Papadopoulos, editors, *Artificial Intelligence Applications and Innovations*, volume 364 of *IFIP Advances in Information and Communication Technology*, pages 179–189. Springer Berlin Heidelberg, 2011.

[Can98] Pedro Cano. Fundamental frequency estimation in the sms analysis. In *International Conference on Digital Audio Effects (DAFX)*, 1998.

[CVG+08] Michael A. Casey, Remco Veltkamp, Masataka Goto, Marc Leman, Cristophe Rhodes, and Malcolm Slaney. Content-based music information retrieval: Current directions and future challenges. *Proceedings of the IEEE*, 96(4):668 –696, Apr. 2008.

[DBT+04] Roger B. Dannenberg, William P. Birmingham, George P. Tzanetakis, Colin P. Meek, Ning P. Hu, and Bryan P. Pardo. The musart testbed for query-by-humming evaluation. *Comput. Music J.*, 28(2):34–48, June 2004.

[dCK02] Alain de Cheveigné and Hideki Kawahara. YIN, a fundamental frequency estimator for speech and music. *The Journal of the Acoustical Society of America*, 111(4):1917–1930, 2002.

[DNS07] Alexander Duda, Andreas Nürnberger, and Sebastian Stober. Towards query by singing/humming on audio databases. In Simon Dixon, David Bainbridge, and Rainer Typke, editors, *ISMIR*, pages 331–334. Austrian Computer Society, 2007.

[GAdM05] Maarten Grachten, Josep Lluis Arcos, and Ramon López de Mántaras. Melody retrieval using the implication/realization model. 2005.

[GB13] Emilia Gómez and Jordi Bonada. Towards computer-assisted flamenco transcription: An experimental comparison of automatic transcription algorithms as applied to a cappella singing. *Computer Music Journal*, 37:73–90, 2013.

[GLCS95] Asif Ghias, Jonathan Logan, David Chamberlin, and Brian C. Smith. Query by humming: musical information retrieval in an audio database. In *In ACM Multimedia*, pages 231–236, 1995.

[HR09] Pierre Hanna and Matthias Robine. Query by tapping system based on alignment algorithm. In *Acoustics, Speech and Signal Processing, 2009. ICASSP 2009. IEEE International Conference on*, pages 1881–1884, 2009.

[IKMI10] Akinori Ito, Yu Kosugi, Shozo Makino, and Masashi Ito. A query-by-humming music information retrieval from audio signals based on multiple f0 candidates. In *Audio Language and Image Processing (ICALIP), 2010 International Conference on*, pages 1–5. IEEE, 2010.

[JMC09] Woojay Jeon, Changxue Ma, and Yan Ming Cheng. An efficient signal-matching approach to melody indexing and search using continuous pitch contours and wavelets. In *ISMIR*, pages 681–686, 2009.

[Kel12] Matthew Brian Kelly. Evaluation of melody similarity measures. Master's thesis, Queen's University, Kingston, Ontario, Canada, August 2012.

[KPH+12] Alexios Kotsifakos, Panagiotis Papapetrou, Jaakko Hollmén, Dimitrios Gunopulos, and Vassilis Athitsos. A survey of query-by-humming similarity methods. In *Proceedings of the 5th International Conference on PErvasive Technologies Related to Assistive Environments*, PETRA '12, pages 5:1–5:4, New York, NY, USA, 2012. ACM.

[LKWL07] Jessica Lin, Eamonn Keogh, Li Wei, and Stefano Lonardi. Experiencing sax: a novel symbolic representation of time series. *Data Mining and Knowledge Discovery*, 15:107–144, 2007.

[LML+03] Micheline Lesaffre, Dirk Moelants, Marc Leman, Bernard De Baets, Hans De Meyer, Gaë tan Martens, and Jean Martens. User behavior in the spontaneous reproduction of musical pieces by vocal query. In *Proceedings 5th Triennial ESCOM Conference*, pages 208–211, 2003.

[LPHA10] Jefrey Lijffijt, Panagiotis Papapetrou, Jaakko Hollmén, and Vassilis Athitsos. Benchmarking dynamic time warping for music retrieval. In *Proceedings of the 3rd International Conference on PErvasive Technologies Related to Assistive Environments*, PETRA '10, pages 59:1–59:7, New York, NY, USA, 2010. ACM.

[LRP07] David Little, David Raffensperger, and Bryan Pardo. A Query by Humming System that Learns from Experience. pages 335–338, Vienna, Austria, September 2007. Österreichische Computer Gesellschaft.

[MB94] Robert C. Maher and James W. Beauchamp. Fundamental Frequency Estimation of Musical Signals using a two-way Mismatch Procedure. *Journal of the Acoustical Society of America*, 95(4):2254–2263, 1994.

[MD01] Dominic Mazzoni and Roger B. Dannenberg. Melody matching directly from audio. In *Indiana University*, pages 17–18, 2001.

[Mur89] James A. H. Murray. The oxford english dictionary, 1989.

[NTN10] Andrzej Czyżewski Ngoc Thanh Nguyen, Aleksander Zgrzywa, editor. *Advances in Multimedia and Network Information System Technologies*. Springer, 2010.

[Pap10] Panagiotis Papiotis. Real-time accompaniment using lyrics-matching query-by-humming (qbh). Master's thesis, Universitat Pompeu Fabra (UPF), 2010.

[PB03] Bryan Pardo and William P. Birmingham. Query by humming: how good can it get. In *Workshop on Music Information Retrieval. Toronto, Canada*, pages 71–111, 2003.

[PEE+07] Graham Poliner, Daniel Ellis, Andreas Ehmann, Emilia Gómez, Sebastian Streich, and Beesuan Ong. Melody transcription from music audio approaches and evaluation. *IEEE Transactions on Audio, Speech and Language Processing*, 15:1247–1256, 2007.

[QLL11] Jing Qin, Hongfei Lin, and Xinyue Liu. Query by humming systems using melody matching model based on the genetic algorithm. *JSW*, 6(12):2416–2420, 2011.

[Ric01] John A. Rice. *Mathematical Statistics and Data Analysis*. Duxbury Press, 3 edition, April 2001.

[RK08] Matti Ryynänen and Anssi Klapuri. Query by humming of midi and audio using locality sensitive hashing. In *ICASSP*, pages 2249–2252. IEEE, 2008.

[SBY02] Jungmin Song, So Y. Bae, and Kyoungro Yoon. Mid-Level Music Melody Representation of Polyphonic Audio for Query-by-Humming System. Paris, France, October 2002. Ircam - Centre Pompidou.

[SG12] Justin Salamon and Emilia Gómez. Melody extraction from polyphonic music signals using pitch contour characteristics. *IEEE Transactions on Audio, Speech, and Language Processing*, 20(6):1759–1770, Aug. 2012.

[SKSA11] Joan Serrà, Holger Kantz, Xavier. Serra, and Ralph G. Andrzejak. Predictability of music descriptor time series and its application to cover song detection. *IEEE Transactions on Audio, Speech, and Language Processing*, 20:514–525, 12/2011 2011. The supplementary file can be downloaded from http://mtg.upf.edu/system/files/publications/IEEEPredictability Supplementary.pdf
To retrieve the published version please go to http://dx.doi.org/10.1109/TASL.2011.2162321.

[SPB77] Thomas L. Magnanti Stephen P. Bradley, Arnoldo C. Hax. *Applied mathematical programming*. 1977.

[SSA09] J. Serrà, Xavier Serra, and R. G. Andrzejak. Cross recurrence quantification for cover song identification. *New Journal of Physics*, 11:093017, 09/2009 2009. Free access in New Journal of Physics: http://dx.doi.org/10.1088/1367-2630/11/9/093017.

[SSG12] Justin Salamon, Joan Serrà, and Emilia Gómez. Tonal representations for music retrieval: from version identification to query-by-humming. *International Journal of Multimedia Information Retrieval*, Dec. 2012.

[SW81] Temple F. Smith and Michael S. Waterman. Identification of common molecular subsequences. *Journal of molecular biology*, 147(1):195–197, March 1981.

[TTM12] Wei-Ho Tsai, Yu-Ming Tu, and Cin-Hao Ma. An fft-based fast melody comparison method for query-by-singing/humming systems. *Pattern Recogn. Lett.*, 33(16):2285–2291, December 2012.

[TWT10] Rainer Typke and Agatha Walczak-Typke. Indexing techniques for non-metric music dissimilarity measures. In *Advances in Music Information Retrieval*, pages 3–17. 2010.

[Typ07] Rainer Typke. *Music Retrieval based on Melodic Similarity*. PhD thesis, Universiteit Utrecht, February 2007.

[ULMSC12] Julián Urbano, Juan Lloréns, Jorge Morato, and Sonia Sánchez-Cuadrado. MIREX 2012 Symbolic Melodic Similarity: Hybrid Sequence Alignment with Geometric Representations. Technical report, Music Information Retrieval Evaluation eXchange, 2012.

[Wie07] Frans Wiering. Can humans benefit from music information retrieval? In *Proceedings of the 4th international conference on Adaptive multimedia retrieval: user, context, and feedback*, AMR'06, pages 82–94, Berlin, Heidelberg, 2007. Springer-Verlag.

[WLL+06] Xiao Wu, Ming Li, Jian Liu, Jun Yang, and Yonghong Yan. A top-down approach to melody match in pitch contour for query by humming, 2006.

[XS13] Emmanouil Benetos Magdalena Chudy Simon Dixon Arthur Flexer Emilia Gómez Fabien Gouyon Perfecto Herrera Sergi Jorda Oscar Paytuvi Geoffroy Petters Jan Schlüter Hugues Vinet Gerhard Widmer Xavier Serra, Michela Magas. *Roadmap for Music Information ReSearch*. Creative Commons BY-NC-ND 3.0 license, 2013.

[ZKT02] Yongwei Zhu, M. Kankanhalli, and Qi Tian. Similarity matching of continuous melody contours for humming querying of melody databases. In *Multimedia Signal Processing, 2002 IEEE Workshop on*, pages 249–252, 2002.

A
List of acronyms

AMDF Average Magnitude Difference Function

ASA Auditory Scene Analysis

CLT Central Limit Theorem

DP Dynamic Programming

DTW Dynamic Time Warping

FFT Fast Fourier Transform

GMM Gaussian Mixture Model

HMM Hidden Markov Model

IOI Inter-Onset Interval

IOIR Inter-Onset Interval Ratio

LogIOIR Logarithm Inter-Onset Interval Ratio

LSH Local Sensitive Hashing

MFCC Mel-Frequency Cepstrum Coefficient

MIDI Musical Instrument Digital Interface

MIR Music Information Retrieval

MIREX Music Information Retrieval eXchange

MRR Mean Reciprocal Rank

PAA Piecewise Aggregate Approximation

QBH Query-by-Humming

SAC Spectral Auto-Correlation

SAX Symbolic Aggregate Approximation

STFT Short-Time Fourier Transform

TWM Two-Way Mismatch