ΕΛΛΗΝΙΚΗ ΔΗΜΟΚΡΑΤΙΑ
Εθνικόν και Καποδιστριακόν
Πανεπιστήμιον Αθηνών
ΙΔΡΥΘΕΝ ΤΟ 1837

SCHOOL OF PHILOSOPHY
DEPARTMENT OF MUSIC STUDIES

THESIS

Computational Analysis of Greek folk music of the Aegean islands

Anna-Maria – M - Christodoulou

Supervisors: **Christina Anagnostopoulou,** Associate Professor
Olivier Lartillot, Researcher

ATHENS

07 2022

THESIS

Computational Analysis of the Greek folk music of the Aegean islands

Anna-Maria M. Christodoulou
A.M.: 1569201700070

Four-member Committee:
Christina Anagnostopoulou, Associate Professor, NKUA
Areti Andreopoulou, Assistant Professor, NKUA
Anastasia Georgaki, Professor, President, NKUA
Olivier Lartillot, Researcher, UiO

Writer Note

This essay is a thesis written for the Music Department of National and Kapodistrian University of Athens and it was submitted for an exam in July of 2022. The writer confirms that the content of this essay is a result of personal work and that there has been the appropriate reference of other members' work, whenever that was necessary, according to the rules of academic integrity.

The opinions that are being presented in this thesis express exclusively the writer and not the supervisors.

Acknowledgements

Through the journey of my research, there are plenty of people who I would like to thank. First of all, I am very grateful to my main supervisor prof. Christina Anagnostopoulou, who supported me throughout my thesis and provided me with useful insights and comments. I am grateful for having someone that always believed in me and who always was happy to coach me on musicology and music technology-related matters. Your research path has been the main motivation for my research.

Also, I would like to thank my co-supervisor Olivier Lartillot who gave me the opportunity to work on his project “MIRAGE” and guided me through my first research steps. I am thankful for all the help, guidance, and constant sharing of knowledge. I was inspired by the way you work and by your passion and restless devotion on your projects.

I would also like to thank the University of Athens and specifically the Department of Musicology who shaped me and gave me all the opportunities to learn and grow as a student and as a person through these last five years. Special thank you to my Music Technology Professors Anastasia Georgaki, Areti Andreopoulou and Orestis Karamanlis for all the courses who helped me learn and appreciate music technology. I also want to thank the RITMO research center and people like Alexander, Anne, Ancha, Marit and Erik who accepted me for my last semester of my studies, where I worked on my thesis in a friendly and multi-cultural research environment. Special thank you to my colleagues and friends at RITMO Atilla, Maja, Julian, Connor, Remy, Alex, Finn and many more who gave me feedback and useful advice and my dear friends Penny, Hara and Georgia who haven’t stopped supporting me throughout my journey.

Finally, I would like to thank my family and especially my parents, my sister and my aunt Konstantina who have been my strength and my biggest support all these years.

Abstract

While newer, advanced computational music analysis models have been developed with the intentions of increasing available information in this field, very little research exists on the computational analysis of folk music in general and Greek folk music in specific. The aim of this study was to examine various types of musical features and patterns in the folk music of the Aegean islands and provide useful information about the structure and the content of this music style. In addition, to compare the tunes of Syrtos and Mpalos dances, but also the various island regions from which they originate, a total of 73 tunes were included in the constructed dataset and the analyses. Feature extraction and pattern analysis revealed that there are indeed melodic and temporal differences both between the two dance types and between the island regions, while there were also various important similarities throughout the whole dataset.

Keywords:

Music analysis, folk music, feature extraction, musical patterns, folk music dataset

Contents

Chapter 1: Introduction

Music analysis is one of the most important tasks in the field of musicology. It provides a better understanding of the concerned musical pieces, as well as a strong comprehension of their overall structure. Music analysis cannot be completely objective, but some musical aspects can be quantified, in order to achieve an analysis which is as neutral as possible. In today's musicology, as well as in many other scientific fields, technology has an important role in facilitating most human tasks. One of the most widely known and used technology tools today is Artificial Intelligence (AI). Artificial Intelligence is a field of Computer Science, focused on the developments of tools who resemble human intelligence, improving decision-making. In musicology, the use of Artificial Intelligence in music analysis and other related tasks, such as automated music transcription or classification, belongs in a field that is called Music Information Retrieval (MIR). Among the numerous MIR tasks, there is also a process called pattern recognition. Pattern recognition is a challenging piece of work, where the main focus is on the detection of motives. Machine Learning developments have turned out to be very useful for the optimization of the results of such attempts.

In this essay the focus is on the feature extraction and the melodic (closed) pattern discovery in Greek folk music and more specifically in songs of the Aegean islands, for circle or opposite (= couples facing each other) dances. More specifically, there have been 73 tunes selected from the dances of Syrtos and Mpalos, originated from 36 selected Aegean islands. Using the Greek Folk Music Dataset, which was created for the purposes of this project, there will be a feature extraction and analysis from the melodies of the selected data - which are in the form of wav and MIDI files - and then there will be a pattern discovery task, followed by a description and clustering of the resulting patterns. The aim is the discovery of the basic characteristics and melodic patterns of this folk music tradition and furthermore the discovery of any possible musical structures.

This project is part of my internship in the project MIRAGE of research center RITMO at the University of Oslo, coordinated by prof. Christina Anagnostopoulou and supervised by Olivier Lartillot, head of the MIRAGE project. One of the goals in MIRAGE is the examination of new computational methods in music analysis that will take it a step further and use these new methods to provide plenty of useful information about various music styles to the public. Folk music is generally underrepresented and there has not been much research in Greek folk music specifically, despite the rich cultural information that can be provided. The main hypothesis is that apart from major similarities across the dataset, there are plenty of differences both based on the dance style and based on the different geographical regions. Most importantly though, the goal is to practice on computational music analysis techniques and to test new ways to improve the performance of the various algorithms that are currently being used in the MIR field for this purpose.

Pattern discovery has been implemented in Indian music (Dutta & Murthy, 2014), Dutch folk music (Kranenburg & Conklin, 2016) and Greek folk music of Crete (Conklin & Anagnostopoulou, 2011), research attempts which are important for the theory and the implementation of such an attempt in the Aegean islands' folk tunes. Researchers from the University of Jyväskylä have performed feature extraction and unsupervised clustering of Finnish folk music (Toiviainen & Eerola, 2001), using the MIDI Toolbox that is also used in this thesis. Finally, relevant pattern discovery techniques (Lartillot O. , 2003) formed the base for the algorithm which was developed for this project.

So, in the theoretical part of this thesis, there will be a detailed explanation of what is a pattern, how is pattern recognition defined and how is it connected with new technologies, such as Machine Learning algorithms, as well as a thorough description of the processes and techniques that are being used in this field. In the practical part of the thesis, there will be a computational music analysis on the musical corpus, aiming for a comprehensive tune description. There will also be an attempt to test a pattern discovery algorithm in the corpus and then compare the results with previous attempts and relevant academic work. Finally, Machine Learning techniques will be implemented for the clustering of the data, both during the stage of the analysis and in the stage of the pattern discovery. More precisely, techniques such as Self-Organizing Maps and t-distributed Stochastic Neighbor Embedding will be performed for the unsupervised clustering of the data, based on some computed features and a clustering will be performed on the resulted patterns, in order to understand what type of relationships do these patterns form.

Chapter 2: Theoretical Background

2.1: Definition of patterns, description of pattern types and detailed explanation

A pattern is defined as a repetitive form or a regularly repetitive task. Patterns are characterized not only by repetition, but also periodicity (Schalkoff, 2007). Patterns can be descriptive, on condition that they appear more frequently in a corpus, compared to an anticorpus (Conklin & Anagnostopoulou, 2011). In music, a pattern can be detected in harmony (harmonic), melody (melodic), rhythm (rhythmic) or form (formic) and it is a multidimensional feature (Schwanauer & Levitt, 1993). Here, the focal point is the detection of closed melodic patterns in Greek folk music.

A pattern is characterized as closed on condition that it gets repeated more times than all the patterns that include it (Lartillot O. , 2014). A melodic pattern is simply a sequence of notes that is being frequently repeated in a piece or in a music piece dataset, and it is worth mentioning that the detection of such patterns is not always objective in every music analysis since the methods and the approaches can vary widely between analysts.

As mentioned above, the main interest of this essay is the feature extraction and detection of melodic patterns in music and more specifically, in Greek folk music. The dances that are focalized on here are Syrtos and Mpalos (Syrtos-Mpalos). These Greek dances come from the Aegean islands and their time signature is usually 4/4. Syrtos is one of the most popular folk dances in Greece to this day. The name comes from the verb syro (σύρω), which means drag. It is danced in circles, while people hold from their hands or from a handkerchief and there is usually a dancer who leads-“drags”-the circle and is free to improvise. Syrtos is danced in many Greek regions, such as Macedonia, Peloponnese, and the islands. Here, the interest is on Aegean islands’ Syrtos, also known as “nisiotikos” (>nisi/ νησί = island). Some of the dance categories whose songs will be analyzed later are Ikariotikos (or Kariotikos), which comes from Ikaria Island, Rhoditikos (from Rhode), Skyrianos (from Skyros) and many more.

Mpalos (or Syrtos-Mpalos) is a variation of Syrtos dance, and its origins are in the Aegean islands. It is danced in couples and in the moves, it incorporates the elements of flirting. This element is a depiction of the Greek society of the old times, when men could not approach women in any other way, other than social events and more specifically fairs and dancing. Therefore, in this dance men usually have more freedom in their moves, but in general its moves are not specific. Instead, both men and women are usually free to improvise based on a certain movement style, varying from island to island.

The reasons behind the selection of this music style are mainly two. First, pattern detection in folk music songs is not yet developed enough in the field of MIR (Music Information Retrieval) since there is a high level of difficulty in information extraction of such music genres. Also, folk music in general is underrepresented in the research domain, maybe due to the lack of enough sources and datasets for the researchers to work with. Therefore, there are many challenges that still need to be overcome and there is plenty of room for new research in this field, which provides a chance for representation of such genres, important to human tradition and music legacy. Second, it is interesting to discover which are those melodic patterns of a Southern European country like Greece and how different might or might not they be between the various islands and between the two dance types. Furthermore, it should be stated that the analysis of folk music and the detection of its melodic patterns is crucial to a deeper understanding of the musical structure, but also gain a clearer comprehension of the culture of the country it originates from. This applies to all folk music analyses. Finally, it is desired that through the highlighting of these discovered musical elements, this attempt will be inspiring to both next generation researchers and artists.

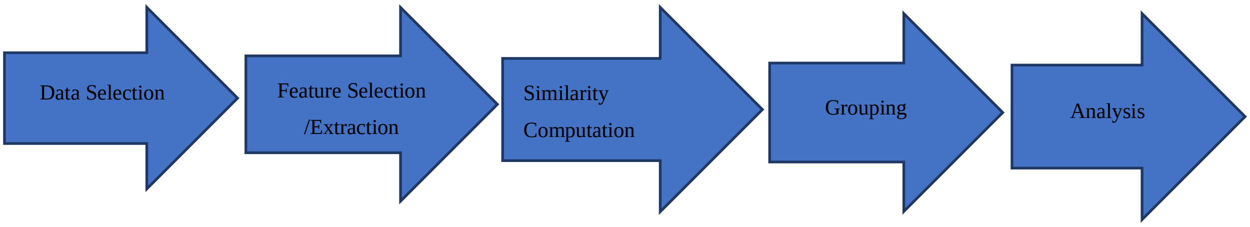
2.2: Definition of pattern recognition. Distinction between the terms «pattern discovery» and «pattern matching»

2.2.1: Pattern Recognition

Detecting patterns or repetitive motives in musical pieces are essential to a clear and delineated music analysis. The main reason behind this is that usually repetition in music suggests that the composer would like the audience to pay attention to this specific part of the music piece, though this theory is not always accurate (Meredith, 2006). There are also many examples in music history, where specific music patterns indicate a music movement or identify a music genre (Conklin D. , 2010). This is not a rule though since a pattern detected by algorithms or by analysts might not be intentionally placed or constructed by the composer. The techniques and the approaches behind these detections are called “pattern discovery” and “pattern matching” and they are mainly part of a bigger category, widely known as “pattern recognition” (Schalkoff, 2007).

Pattern recognition is a scientific field, where specific Machine Learning algorithms are used to identify repetitions in a corpus of data. It is widely used in Computer Science, but also many other fields, such as Medicine, Statistics, even Arts. Some of the most common implementations of pattern recognition algorithms are image processing, signal processing, data analysis, data compression and data classification. Due to the early approach on which it was based, pattern recognition is usually synonym with classification (Pavlidis, 1977). In music, pattern recognition belongs in the MIR (Music Information Retrieval) field.

Pattern recognition is “the description of measurements” (Schalkoff, 2007). Since the main interest in this essay is musical patterns, it would be useful to mention that pattern recognition in music is the description of measured music events. In contrast with pattern discovery and pattern matching, which will be further analyzed later, pattern recognition is not a technique, but an extensive body that contains various techniques and tools, in order to provide a descriptive information regarding (musical) events.



1. Pattern Recognition Process

There are three approaches for pattern recognition: structural (or syntactic), statistical and neural (Schalkoff, 2007). Structural PR (Pattern Recognition) is based on semi-supervised learning techniques (see Chapter 2.4), and it is mostly useful for describing the relationship between different music parts or pieces. As the name suggests, in this technique patterns are characterized using structural representations (Pavlidis, 1977), or more specifically patterns are represented by a sequence of numbers.

Syntactic techniques such as parse grammars are useful for the Syntactic approach. Therefore, one of the basic approaches in Structural PR is string-matching, where the data are represented as a sequence of characters (or numbers) in a string and then usually a search algorithm is implemented, which assists the description of the different strings. Since the focus here is on melodic pattern sequences, the notes of melodic patterns can be placed in order within a string-type sequence (Velankar & Dr. Kulkami, 2017). If the patterns are monophonic, there is no need for a multidimensional string representation. After creating such strings, a similarity computation must be done between them and then create graphs that will assist the string-matching method (Velankar & Dr. Kulkami, 2017).

Neural PR is based on ANNs (Artificial Neural Networks), and it is useful to detect similarities and repetitions in (music) data, using ML (Machine Learning) data (Jesan, 2004). During this approach, neural networks are trained to recognize specific patterns and then try to output these patterns, after locating them in each dataset (Schalkoff, 2007). Among their biggest benefits in pattern recognition are availability, low need in domain-specific knowledge (Velankar & Dr. Kulkami, 2017), adaptive learning, tolerance to errors and independent organization, while some important drawbacks are the learning speed and feature representation (Jesan, 2004). In a music dataset, the use of Neural Pattern Recognition would mean that an analyst could train a neural network to identify a certain set of melodic patterns for example that are considered to activate the feeling of happiness to people.

Statistical PR is based on supervised learning, and it is used to identify where does a music part belong, using statistics (Pavlidis, 1977). Many algorithms for Statistical PR are based on statistical methods like Bayes Theorem. In Statistical PR, the data are represented as vectors or strings of numerical values (Velankar & Dr. Kulkami, 2017). These can be extracted either as raw input (Pavlidis, 1977) or in a way that resembles a classification task (Velankar & Dr. Kulkami, 2017). This approach is the most useful for melodic pattern recognition, since a sequence of notes can easily be transformed into a string and then create a hierarchical structure, where the patterns can be recognized after computing the graph structure for example (Velankar & Dr. Kulkami, 2017). Another technique is the similarity computation of arrays based on geometrical distances. Here, the arrays consist of a number sequence that represents coordinates in a vector space (Pavlidis, 1977).

The most important parts of the pattern recognition algorithms are the descriptive and the explorative part (Fedorenko, 2020). The descriptive part is used to separate repetitive or common patterns in a music piece, while the explorative is used to identify and confirm these patterns. For example, a pattern recognition algorithm trained to recognize rhythmic patterns will try and detect repetitive rhythmic structures in a music dataset using its descriptive part, and then go back to its selection and decide which of these patterns are indeed significant, by activating the explorative part.

2.2.2: Pattern Discovery

Pattern discovery algorithms are designed to detect motives, repetitive patterns or themes that are of high importance. What is of high importance, though, is not predefined, but it must be decided by the analyst. In music, these algorithms are widely used to achieve a better understanding of the musical structure, while at the same time identify the characteristic patterns which define this overall structure. Patterns can be discovered intra-opus and inter-opus (Conklin D. , 2010). Intra-opus patterns are the ones that can be found multiple times in the same musical piece, while inter-opus are the patterns which appear frequently in multiple songs of a dataset. It is worth mentioning that sometimes in pattern discovery a pattern is defined as “a sequence of feature sets” (Conklin D. , 2010), where “feature set” means an assemblage of features that characterizes a music event.

One usual approach is the development of a suffix tree data structure. A suffix trie is a simple data structure that is used to store associative arrays, where every key is a string and every node is a prefix of that string (Crochemore & Lecroq, 2009). A suffix or PAT tree is a compressed trie that contains information about all the suffixes of a string and their positions (Crochemore M. L., 2009). Conklin & Anagnostopoulou, 2011, performed a discovery of multiple viewpoint patterns, using suffix trees. Before moving on to the description of this approach, it is important to explain that a viewpoint is a function that is extracted by the music analyst and encodes information about a specific music feature (Conklin & Anagnostopoulou, 2011). A viewpoint takes the music information and turns it into an abstract value (Conklin D. , 2010). A pattern discovery algorithm that uses suffix trees, transforms all musical pieces into a viewpoint sequence for every viewpoint and then inserts all generated suffixes from these sequences, creating a suffix tree. This suffix tree contains all sets of patterns that appear x times throughout the musical piece. If the patterns appear often, according to the specified statistical significance, then they are considered important and they get analyzed (Conklin & Anagnostopoulou, 2011).

String-based methods, where music data is transformed into string forms are widely preferred in pattern discovery (Meredith, 2006). More specifically, each musical piece is transformed into a set of strings, where every string can describe a music segment, for example a note sequence. Every string is a combination of substrings, which can represent for example a specific note (Meredith, 2006). String-matching algorithms are usually used for searching tasks, such as pattern matching in a dataset (Gimel'farb). This approach seems to be efficient for a simple repetitive motive discovery, but it becomes more complicated and inefficient when it comes to variated query patterns and polyphonic musical structures (Meredith, 2006). This problem can be solved by using algorithms who accept multidimensional music point sets as input, instead of single string forms.

MGDP (Maximally General Distinctive Pattern) discovery algorithm is yet another example of pattern discovery techniques. As mentioned before in this essay, a pattern is considered distinctive when it appears statistically often in a musical corpus (Conklin D. , 2010). What is interesting about this algorithm is that it has been previously applied in folk music (Conklin D. , 2010). This algorithm is a sequential pattern mining algorithm based on the depth-first strategy that was implemented by Ayres et al. (Ayres, Gehrke, Yiu, & Flannik, 2002). The MGDP discovery algorithm has three main frameworks. It contains a catalogue of viewpoints that is being employed to describe the musical data. Then, there is a predefined minimum value for the probability that defines whether a discovered pattern is worth to be reviewed as frequent. Finally, there is a minimum acceptable value for the interest, which defines whether a pattern is distinctive (Conklin D. , 2010). More specifically, given a traditional music dataset, if a specific pattern a-b-b-a appears more than a predefined number x in the dataset and not in a music piece, then it is considered distinctive. If this pattern does not belong to any other bigger patterns, or in other words, if it is the most general pattern, then it is considered Maximal General Distinctive Pattern.

Finding similar patterns in music based on cognitive perception is also an apt example of pattern discovery. After computing interval distances, which are also computed taken perceptual distance into account, an association is made with its occurrence frequency throughout the data that is placed in a hash-table (Lartillot O. , Discovering Musical Patterns through Perceptive Heuristics, 2003). Afterwards, there is a similarity test between the intervals and if a pair of intervals is like its precedent, then it means that a pattern is located in the table. It is worth mentioning that in this approach there is a distinction between pattern classes and pattern occurrences, where multiple pattern occurrences create a pattern class (Lartillot O. , Discovering Musical Patterns through Perceptive Heuristics, 2003), like graphs and nodes.

2.2.3: Pattern Matching

Pattern matching works exactly like pattern discovery, but also incorporates a matching process, where the algorithm attempts to detect whether a specific pattern already exists in the set of discovered data (Eikvil & Huseby, 2002). Pattern matching is considered as a tool of pattern recognition (Eikvil & Huseby, 2002). In more detail, such algorithms try to detect similarities between a query pattern and an existing pattern in the database. This happens by calculating the distance measure between them, such as the Euclidean or the city block distance (Eikvil & Huseby, 2002). In other words, pattern matching is based on the quantification of similarity between two sets of patterns (Schalkoff, 2007). A big benefit of the pattern matching algorithms is the resemblance of the relevant human music analyses (Meredith, 2006), while an important drawback is the high-computational costs and therefore a slow performance, which leads to the need of enhancement, usually utilizing heuristic algorithms.

An example of such an algorithm is SIA (Structure Induction Algorithm). The output of this algorithm is the detection of all Maximal Translatable Patterns (MTPs) for a note sequence for example in a dataset. When a Structure Induction Algorithm is used for music pattern matching, it accepts as input multidimensional music data (or points sets). Every musical pattern in the dataset is represented by a vector, on condition that this vector can map the information of this pattern to another pattern in the same dataset (Meredith, 2006). A Maximal Translated Pattern (MTP) of a vector is a pattern whose all points are interpreted by a vector to other points in the dataset (Meredith, 2006). By sorting all MTPs, SIA is able to detect all empty MTPs in the dataset and then generate non-empty ones in a list.

In pattern matching, there are two approached on which an analyst can be based: fuzzy and exact (Lartilllot & Toiviainen, 2007). In the fuzzy approach, the similarity of patterns is computed, by defining a numerical distance based on the musical intervals. Depending on the value of this distance that is supposed to characterize similarity, an analyst can determine whether a motive is also a repetitive pattern or not. This approach can be problematic since it takes into consideration only one musical dimension. In order to overcome such problems, the exact approach is followed, where pattern matching is applied through multiple music dimensions (Conklin & Anagnostopoulou, 2011; Lartilllot & Toiviainen, 2007).

Heuristic algorithms turn out to be useful in both pattern discovery and matching. More specifically, the importance of heuristic algorithms in pattern discovery lies in the detection of similar music segments and then the creation of data clusters, based on this similarity (Dannenberg & Hu, 2002). This can be done easily for large scale data, reducing the computational cost, while increasing speed and efficacy. At the same time, using heuristic algorithms in pattern matching can significantly increase speed, by compressing the data in a point-set matching algorithm (Meredith, 2006).

2.3: Music Similarity

Pattern recognition aims to determine whether a note sequence is being repeated in a music dataset (Eikvil & Huseby, 2002). Usually, this repetition is being detected by computing music similarity (Schalkoff, 2007). In musicology, music similarity depends on many music dimensions, some of which are objective, such as pitch or frequency, while others are subjective, like timbre or rhythm.

In this essay, the focus is the discovery of similar melodic patterns. One approach for similarity computation between two melodic parts is the use of graphs (Orio & Roda, 2009). More specifically, each melody can be represented as a node, while the whole music dataset can be the graph. The graph could contain hierarchical trees that represent patterns (Velankar & Dr. Kulkami, 2017). Trying to detect the closest way between the nodes in this graph is a way to detect similarities between these nodes (melodies). Melodic profile is the usual dimension that is used to compute similarity (Orio & Roda, 2009). In case of melodic patterns, the notes sequence pattern is generally represented as ordered list of notes with string type data structure.

Before the major development of Neural Networks, Statistics was the main tool for Pattern Recognition. The base of this approach is the search of statistically important motives that are being repeated in a dataset a prefixed number of times. As it has been thoroughly mentioned above, music similarity can often be computed by calculating the repetition frequency of a note sequence. This can be done with Statistics and more precisely methods and tools like Markov Models, Bayes Theorem or even Machine Learning algorithms such as Random Forest.

A Markov Model is a set of functions that models pseudo-randomly changing systems. The simplest Markov Model is a Markov Chain, which is a stochastic model that describes the possible outcomes of a certain event, based on its last state, or a specific predefined number of states (Dubnov, Assayag, Lartillot, & Bejerano, 2003). A Hidden Markov Model (or HMM) is a statistical model that works like a simple Markov Model, but, as the name suggests, its Markov processes are hidden. In music, Markov models are mostly useful for music generation, but it has been quite useful for pattern recognition as well. For example, a HMM could be used to recognize specific type of patterns in a music corpus (Pikrakis, Theodoridis, & Kamarotos, 2002) (Sinith & Rajeev, 2004) or harmonic content could be discovered through tools based on HMMs (Wang & Dubnov, 2015).

Bayes Theorem is a statistics theorem, which calculates how likely is an event to occur, based on previous knowledge or related conditions. In music, it is mainly used for music classification tasks since it is useful for similarity detection. Such classifiers use Bayer Theorem usually in the form of Bayes rule:

$$P(A|B) = (P(B|A) * P(A)) / P(B)$$

What this rule says is that the probability of an event A happening under condition B depends on the probability of an event B happening under condition A times probability of A and all of this divided by the probability of B. This rule is also useful in pattern recognition, because if the query is melodic pattern detection, then the similarity of the various music segments is essential to understand and discover repetition and to a larger extent, patterns.

A Decision Tree is a flowchart-like structure that is used to model all possible outcomes of a specific decision. Random Forests are a set of Decision Trees that randomly choose which trees should be generated and where, as well as where should each tree be split (Breiman, 2001). Another formal definition is “A classifier consisting of a collection of tree-structured classifiers {h(x, k), k = 1,...} where the {k } are independent identically distributed random vectors and each tree casts a unit vote for the most popular class at input x.” (Breiman, 2001). Two main methods of Random Forest building are “Bagging” and “Random subspaces” (Ali, Khan, Ahmad, & Maqsood, 2012). In these methods, Random Forests are fed with randomly selected features (Breiman, 2001). Amongst the biggest advantages of using Random Forests for data clustering are the minimum sensitivity of outer data during training process, the absence of overfitting and accuracy definition, as well as the fact that they don’t need any pruning (Ali, Khan, Ahmad, & Maqsood, 2012). Random Forests are therefore useful for pattern recognition tasks (Dapogny, Bailly, & Dubuisson, 2015).

It is possible and fairly common to measure similarity using string-matching techniques, like, for example, computing edit distance. Edit distance is a way to compute similarities of strings, where in this case, a string is a sequence of notes. As mentioned before, this method is not equally efficient in multidimensional or polyphonic music analysis, but mostly for monophonic. One possible way to improve this method is to calculate entropy, the level of certainty or uncertainty in a corpus (Xiaoqing, Xueqian, Wei, & Wanggen, 2010). Entropy is widely used in music similarity tasks, and it depicts the level of resemblance possibility between a sequence s and a model m (Laney, Samuels, & Capulet, 2015). Cross entropy measures musical contrast, using the IDyOM framework, which is a framework based on multi-order Markov models (Laney, Samuels, & Capulet, 2015). Quadratic entropy suggests that a sequence x is similar to a sequence y if their average entropy is maximum (Çataltepe & Altinel, 2007).

Note sequences can be transformed into strings, graphs, vectors, or even Markov Chains. Another possible transformation that is very useful for music similarity is the conversion of pitch sequences into n-grams. An n-gram is an n-length combination of things, in this case pitches or pitch intervals (Eikvil & Huseby, 2002). Then, it is possible to compare the n-grams and find similarities or irregularities in the extracted features. For melodic patterns in specific, one can compare occurring patterns with main melodies in a musical piece or concatenate a melodic sequence into smaller melodic events (Velankar & Dr. Kulkami, 2017). For the second approach, there are some assigned weights on each pitch or pitch interval and note sequences are being transformed into n-grams and later compared using string-matching techniques (Velankar & Dr. Kulkami, 2017).

Music similarity can be computed in many ways. Indicatively, an option for music similarity computation is the effort to reach the most basic form of the melodies, using music analysis theories like GTTM or Schenker (Orio & Roda, 2009), which are theories based on music hierarchy. This approach though is not objective, since the music hierarchy is not defined by certain rules, but instead it is open to the analyst. Another way to compute music similarity is to choose and isolate one characteristic, such as music pitch, and then classify it into many subcategories, such as natural, sharpened or flattened. Furthermore, Brute-Force algorithms can be useful and fast, since they compute the distance between all sets of points in a dataset, aiming to find the closest solution to the query (Dannenberg & Hu, 2002).

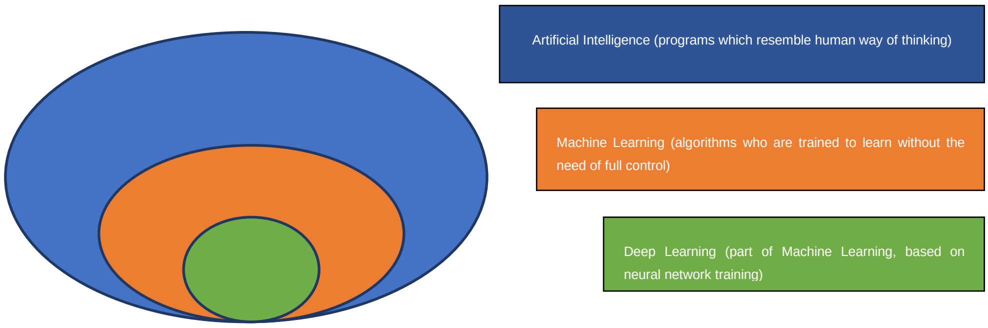
2.4: Machine Learning

2.4.1: General Information. Supervised vs. Unsupervised Machine Learning

Machine Learning is a core component of pattern recognition. The term Machine Learning is very often confused with Deep Learning and Artificial Intelligence, so it is important to make a distinction before jumping to the analysis of its use in pattern recognition tasks. Artificial Intelligence (AI) is a Computer Science field, focused on the design of computer systems which resemble human behavior, but also exceed human capacity (Pavlidis, 1977). It is widely used in many technology and non-IT fields, benefitting human tasks by significantly reducing time-consuming or repetitive tasks. Machine Learning is a branch of AI. It is a set of algorithms that show the machines' capabilities to mimic human behavior. On the other hand, Deep Learning is a part of Machine Learning methods, based on the human brain's structure and functions. In Figure 2, there is a distinction specification of these three terms.

Machine Learning is divided in two categories: supervised and unsupervised (Velankar & Dr. Kulkami, 2017). What is meant by supervised learning is that there is a known relationship between input and output data and the aim is to train the algorithm to understand and approximate the output data, using a function (Soni, 2018). In other words, supervised learning algorithms are trained using labeled input data, aiming for a specific output. For example, supervised learning can be used to predict if a specific song will interest a person, based on their previous listening data. In unsupervised learning there are no data labels, and the goal is to discover the inherent structure within a dataset (Soni, 2018). For example, unsupervised learning algorithms are useful to provide a description to unlabeled music data. There is also a mixture of the two categories, known as semi-supervised learning and it basically contains a small set of labeled data and a larger set of unlabeled data.

As stated above, machine learning is exploited in pattern recognition, in order to discover repetitive music parts in a music corpus. All of today's pattern recognition techniques are based on Machine Learning algorithms and there has been much research on the implementation of such algorithms in music, but not enough in folk music and specifically Greek.



2 Artificial Intelligence - Machine Learning - Deep Learning

2.4.2: Compression Algorithms

2.4.2.1: Heuristic Algorithms

As mentioned previously, heuristic algorithms are useful for pattern recognition techniques, mainly due to the compression of large-scale data. What these algorithms actually do is to speed up the computation of a result by searching for approximate solutions (Kokash, 2005). This means that the result is not the best solution, but the fastest one and the closest to the best one. For this reason, heuristic algorithms belong in the category of approximate algorithms. According to the level of complexity that a heuristic algorithm has to face, there are the relevant difficulty categories, or so called "classes" (Kokash, 2005). The main classes are P and NP, where in class P the problems are solved in a deterministic Turing machine in polynomial time, where in class NP problems can be solved in a non-deterministic Turing machine, but still in polynomial time (Kokash, 2005). What is meant by polynomial time is time complexity and it describes how much time does a machine need to complete a task.

Among some of the heuristic techniques, there is simulated annealing, tabu search, swarm intelligence, genetic algorithms, ANNs (Artificial Neural Networks) and SVMs (Support Vector Machines) (Kokash, 2005). Simulated annealing is a heuristic process that aims to detect the best solution, by approximating the global optimum. In melodic pattern recognition simulated annealing could be used to find the fastest way to compute similarity between intervals in a dataset. Tabu search iterates through a given dataset and stores in memory the evaluated data, which after evaluation are considered "tabu". It is mostly useful for its adaptive memory, which gives an advantage to search problems. Swarm intelligence is divided in two parts-ACO and PSO-and it resembles ant and swarm behavior respectively. In ACO (Ant Colony Optimization) the algorithm climbs up to a graph and alters the nodes in such a way that PSO can solve the problem in the best way possible. Also, PSO can deal with local optima, which gives it an extra advantage. In melodic pattern recognition swarm intelligence could be very useful to reduce the music data dimensions and achieve a faster clustering. Genetic algorithms are also useful for optimization problems and their structure resembles, as their name suggests, Darwin's selection. ANNs work like human brains, in the sense that they are trained to "learn". In pattern recognition, ANNs are widely used in feature extraction and classification due to their high efficiency in this field (Kim, 2010). Finally, SVMs are mainly used in classification and regression tasks, and they are based on supervised learning. An important benefit of SVMs is their high efficiency in multidimensional data and the use of subsets for the decision functions, which makes them fast and low cost. A minor disadvantage is overfitting, which can easily be avoided by using Kernels, which are functions that assist complicated computations.

2.4.2.2: Lempel-Ziv-Welch Algorithm

Lempel-Ziv-Welch Algorithm (LZW) is used for the compression of big computer files, in order to save up space or to get rid of redundant files in a computer system (Kotze & Kuhn G., 1999). In a pattern recognition problem example LZW compression could be used to detect new patterns in a pattern sequence in an optimal and fast way. Note sequences can be transformed into strings and then encoding the strings into codes, which need less memory space. Dubnov et al., 2003 used this algorithm to calculate the probabilities of a certain pattern appearing in a sequence. More specifically, they created a pattern dictionary and a tree that contained a pattern string on every node (Dubnov, Assayag, Lartillot, & Bejerano, 2003). Then, they created an algorithm that is going through the sequence, trying to test whether a prefix has already been detected or needs to be added in the sequence.

2.5: Feature Extraction

Feature extraction is a method where information is being extracted by a larger corpus and compressed, to gain some more specific information about the dataset. Mathematically, the principal of feature extraction is the transformation of an n-dimensional vector x into an m-dimensional vector y, using a mapping f (Ding, Zhu, Jia, & Su, 2012). Feature extraction is a very important step in pattern recognition in general. Some musical features that are usually being extracted in musical pattern recognition are time duration, dynamics, timbre, and pitch. Depending on the knowledge gain goal of each researcher, the relevant feature or set of features are extracted each time. In this project melodic sequences will be analyzed, therefore the extracted feature that is of main concern is pitch. Pitch is a musical feature that provides information about how “high” or “low” a sound is perceived, and it is correlated to the fundamental frequency (Eikvil & Huseby, 2002).

Musical features can be classified into three main categories: low, middle, and high, and pitch belongs in the middle-level category (Velankar & Dr. Kulkami, 2017). It is important to choose one specific feature for recognition before proceeding to a broader recognition task. For example, pitch is a musical feature that could be a starting point in pattern recognition, especially if the musical structure is not known, because pitch alterations doesn’t seem to affect the overall musical information expressed in a piece (Eikvil & Huseby, 2002).

2.6: Data Categorization

2.6.1: Classification

After extracting the desired features from the selected dataset, it is important to proceed to the data categorization task. This task is helpful, because mapping the data into categories makes it easier for their analysis and description. In order to do this splitting, a training set is needed, that contains certain labels which describe the data (Lippmann, 1989). Then, there is a needed testing set, until a conclusion is reached. In pattern recognition for example, the data categorization can start by a predefined set of melodic features, where predefinition is the desired labeling. Then, after extracting pitch information, the resulting melodic patterns get classified and matched to the pre-defined features, separating them into multiple feature categories. This task is called a classification task and is mostly used in pattern recognition tasks.

There are four types of classification: binary, multi-class, multi-level and imbalanced. In binary classification, the algorithm can map the data into two categories, for example ascending and descending melodic sequencies. In multiclass, there are more than two possible classes, and the data can be mapped to one certain class. For example, melodies can be major, minor, Lydian or Hijaz, but not two of them at the same time. In multi-level classification a pattern can be mapped into more than one category, for example a certain interval can belong to the music of more than one islands. Most of the times, all the above classification types assume that all classes are equal, or all classes in the same classifier is given an equal amount of data. This is not the case in real world problems, so imbalanced classification is used with unequal classes during the training task.

Though pattern recognition is based on supervised Machine Learning, there are also combined supervised/unsupervised learning techniques (Lippmann, 1989). Some of the classifiers that are commonly used in Machine Learning for this task are: Linear, such as Naïve Bayes, Logistic Regression or Fisher’s linear discriminant, Support Vector Machines (SVMs), Kernel-based classifiers, Decision Trees, such as Random Forests and Neural Networks. Linear classifiers such as Logistic Regression are designed just for the task of classification, therefore have a high performance, but can only be implemented in binary type classification. Support Vector Machines (SVMs) can be applied to multi-class classification, since it works with multi-dimensional data and is memory-efficient, but cross-validation is needed for the probability estimation. Kernel-based classifiers, such as K-nearest neighbors, are simple and effective for large datasets, but they also have high computational costs. Decision Trees are effective in all types of data and have a simple structure, but they can often create too complex trees. Random Forests, which are a special type of decision trees can overcome the problem of overfitting and complexity and can therefore become more accurate than the rest of the Decision Trees, but they are slow and difficult to implement. Last but not least, Neural Networks such as Artificial Neural Networks (ANNs) are usually implemented for classification tasks, since they have big accuracy and high speed, but they can also be time consuming to develop and computationally costly.

2.6.2: Clustering

Data clustering is yet another example of data categorization. Here, as the name suggests, the selected algorithm segments the data into clusters. This segmentation is based on similarity and distance measurements (Xu & Tian, 2015). Clustering is an unsupervised task and can be used mostly in pattern discovery. More specifically, in contrast to pattern classification where one has a set of labeled data with which they train a model to map new data into this set, in data clustering in general and pattern clustering in specific there is not any known information regarding the data relationships and the algorithm has to figure them out itself.

Clustering algorithms can be divided into five main categories: centroid-based, density-based, distribution-based, hierarchical, and neural network-based (Jain, Murty, & Flynn, 1999). Centroid-based clustering algorithms, such as K-means clustering algorithm, are creating clusters based on the distance of data points from the specified clusters. K-means is a very simple and efficient algorithm, but it is also slow on big datasets, since it iterates through all data points.

Density-based clustering algorithms, such as DBSCAN, create clusters where they detect a dense number of data points. These algorithms are useful, because they don't have a shape constraint, but they seem to have a difficulty when they are implemented in high dimensional data, since the lack of cluster outliers can sometimes get them invisible.

Distribution-based clustering algorithms, like Gaussian Mixture Model, divide the data into clusters, according to the distance of a data point from the distribution center. This means for example that the further away a data point is from a distribution center, the less likely it is that it belongs to this cluster. The disadvantage of this algorithm is that one can only use it provided they know about their data distribution.

Last but not least, hierarchical clustering algorithms, such as BIRCH, Agglomerative or Mean-Shift follow a top-down approach and creates cluster trees. Affinity Propagation algorithm is a special type of clustering algorithm that can be used if the data relationships, its distribution, and its patterns are unknown.

2.6.3: Regression

In Machine Learning there is yet another way to classify the data into categories. An algorithm based on supervised Machine Learning techniques is called Regression. What this algorithm basically does is detecting the relationship between different variables with predicted unknown variables, without having any predefined models or further information. In music pattern recognition, this algorithm could be used for example to estimate the relationship between string plucking and sound amplitude in beginners and professional guitarists.

Regression can be either Linear or Logistic. Linear regression can be simple Linear or Bivariable, where the goal is to determine the relationship between an independent input and a dependent output variable, using a line. Mathematically, the simple form of Linear Regression is expressed as: $\mathbf{Y = a + bX + u}$, where Y is the dependent variable, a is the intercept, b is the slope, X is the independent variable and u is the regression residual. If the goal of the linear model is to predict the relationship between multiple independent input variables and one dependent output variable, then the process is called Multivariate Linear Regression, mathematically depicted as $\mathbf{Y = a + b1X1 + b2X2 + b3X3 + ... + btXt + u}$. Regression can also be nonlinear in case the parameters of the data are nonlinear. Instead of a line, nonlinear models use a curve. Mathematically, it is expressed as: $\mathbf{Y = f(X,\beta) + \epsilon}$, where Y is the output dependent variable, X is a vector of P predictors, β is a vector of k parameters and ϵ is the error term.

Logistic regression is used as a solution to a classification problem. The output can be either two options (Binary Logistic Regression) or multiple. If the multiple results options have no order, then regression is called Multinomial, while if the multiple results are ordered, it is called Ordinal.

Chapter 3: Related Work

3.1: Computational Folk Music Analysis

Computational music analysis has been performed in other folk music styles, such as Turkish makam (Bozkurt, Ayangil, & Holzapfel, 2014). In this research, the task contains computational pitch, rhythm, and timbre analysis. For pitch, the estimation is autocorrelation-based, and histograms are used for observation of melodic sequences. The recordings of the files are multi-instrumental and simplified, due to the microtonal nature of the tunes and the files are both symbolic and audio. It is worth to mention though, that today there is an automated transcription tool available for Turkish micromelodic tunes (Benetos & Holzapfel, 2015). As it happens in this thesis, rhythm is analyzed based on onset location, while downbeats are important. A major difference though is that in Turkish makam music analysis timbre is an important factor, while the analysis of this thesis does not consider this aspect. Furthermore, in the relevant research attempt recognition is performed, based on features such as key signatures and rhythm.

Feature extraction has also been implemented in Essen collection of folk music (Toiviainen & Eerola, 2001). Features such as pitch class distribution and key strength are also computed in this thesis, while the main difference is that all tunes are transposed in the same key prior to feature extraction, while in this project the tunes remained in their original form. Furthermore, Toiviainen and Eerola used Self-Organizing Maps for the representation of data, based on selected features and this is a technique that was also implemented in this thesis, due to the high dimensionality of the data features' results.

3.2.: Pattern Discovery in Greek Folk Music

Pattern discovery and more specifically its applications in folk music styles has been the interest of many other researchers in the past. A characteristic example is the detection of melodic patterns in Cretan folk music from MIDI files (Conklin & Anagnostopoulou, 2011). In this case, there was a large dataset containing 106 Cretan folk song MIDI transcriptions in the final process, that were analyzed and split into four main categories, in order to detect whether there are any melodic similarities within these categories. These four categories formed the knowledge base of the corpus and consisted of: song type, song hypertype, area and hyperarea. To clarify the content of these categories, song hypertype can be defined as a broad song type category, such as Tavla, while song type can be defined as a specific Tavla song, which has a unique characteristic. Same explanation can be given for the categories area and hyperarea.

The purpose of this research was to detect melodic patterns within the corpus, by following the tact of Maximally General Distinctive Pattern (MGDP) detection (Conklin & Anagnostopoulou, 2011). The definition of this pattern type, as well as the process of reaching it, has been analyzed above. After ending up in some maximally general distinctive patterns, they are ranked by their level of distinctiveness (Conklin & Anagnostopoulou, 2011). One of the main differences between this approach and the one that is suggested in this thesis, is that here the metrical position of the patterns is not taken into consideration.

3.3: Pattern Discovery in Folk Music of the world

Similar work using MIDI transcriptions of folk music has been also done in Native American music (Neubarth, Shanahan, & Conklin, 2017), using Supervised learning techniques. The goal of this approach is the detection of so-called “interesting” patterns. A pattern can be considered interesting if it is distinctive and statistically significant, according to a predefined set of threshold values (Neubarth, Shanahan, & Conklin, 2017). A similar approach will be followed in this thesis for the post-processing of pattern detection, before moving on to the clustering process.

Pattern discovery techniques have also been applied to Dutch folk music, by analyzing human-annotated musical data (Kranenburg & Conklin, 2016). This project followed a viewpoint approach. As mentioned earlier, viewpoints in music analysis are functions that encode descriptive information about a specific music feature. Viewpoints can also be divided in three main categories, basic, derived and constructed (Kranenburg & Conklin, 2016). Derived viewpoints result from other viewpoints and can be used for a more complicated attribute description. In this research, a derived viewpoint named “phrpos” is used, which provides the analyst information related to the pattern position within a phrase. Viewpoints are not in the focus of this essay, but the idea behind providing the information of pattern position is of high interest.

Yet another example of pattern discovery in folk music is the motif detection in Indian ragas (Dutta & Murthy, 2014). In this example the dataset consists of vocal audio files. The main purpose in this research is the detection of Rough Longest Common Subsequence, or RLCS within a specific raga. So, the patterns that are being discovered here are intra-corpus. An RLCS algorithm compares two musical segments or, more specifically, note sequences, taking into consideration their local similarity with respect to the longest common sequence.

3.4: Pattern Discovery from XML files

It is worth mentioning that pattern discovery cannot only be applied on MIDI data, but also other forms of symbolic music representation, such as Music XML (Nuttall, Casado, Ferraro, Conklin, & Repetto, 2021). In this approach, the first step is data representation. Here the data is in the form of n-grams, simulating a Natural Language Processing (NLP) method of pattern discovery. More specifically, each XML file is a so-called “bag-of-patterns”, containing n-grams. For the purposes of this research, temporal and rest information are discarded, but for this thesis both information is useful.

After data representation, there are two methods that are used in this research. On the one hand, there are statistics, which are implemented to calculate how often does a pattern appear through the score and what how is that calculation interpreted for the overall score characterization. On the other hand, SIA algorithm is implemented, in order to find the MTPs of each note sequence in a score, using the geometric approach. In this essay, SIA would not be helpful, since patterns of the same interval distance but different musical key would not be put in the same cluster. Therefore, SIATEC algorithm would be more appropriate. Finally, based on a predefined minimum accepted frequency of occurrence (MFO), most frequent patterns are selected (Nuttall, Casado, Ferraro, Conklin, & Repetto, 2021).

3.5: Pattern Discovery from Audio files

An approach that is worth to mention is pattern discovery directly from audio files (Wang & Dubnov, 2015). This research focused on extracting repetitive patterns, or themes, from polyphonic classical music audio files. The implementation was based on the Variable Markov Oracle (VMO) data structure, which can extract information from an audio signal, cluster this information and create a symbolization (Wang & Dubnov, 2015). The VMO is acquiring this information through the Factor Oracle (FO) and the Audio Oracle (AO). The Factor Oracle is a suffix tree that can automatically detect patterns through a sequence, while the Audio Oracle is FO's extension, applied in audio signals (Wang & Dubnov, 2015). Feature extraction is important for this approach. Furthermore, a pattern evaluation is being held, in order to pick out only the patterns who appear with a statistical occurrence above the prespecified threshold.

Another interesting pattern discovery attempt directly from audio files is based on heuristics (Dannenberg & Hu, 2002). In this work, a distance function is computed between two music segments, to detect similarity. For monophonic analysis, when transcribed pitch analysis is possible, dynamic programming is used to iterate through a note matrix and detect patterns. Otherwise, a spectral analysis is implemented for similarity detection. For polyphonic analysis, audio files were transcribed to MIDI.

Pattern discovery from audio can be quite successful when combined with music segmentation techniques (Nieto & Farbood, 2014). Nieto & Farbood, 2014 developed a technique for pattern discovery from polyphonic audio files. After creating a harmonic representation from the audio files' spectrograms, they used a beat tracker which provided information regarding synchronous patterns that are not only located at the beginning of a beat. They also computed the transposition invariant SSM (self-Similarity Matrix) to maintain the algorithm invariant to key transpositions of the patterns. Then, the goal was to detect all repeated patterns in the dataset and create an algorithm that calculated the number of pattern repetitions per path in S.

3.6: Greek Music Dataset

As mentioned above, there are many researchers across the globe who have been interested in extracting patterns from music, from classical to folk music and from simple symbolic representation files to complex polyphonic audio files. Another thing that is also important is the creation of folk music datasets. This will help both researchers who wish to investigate a folk music tradition and extract useful information and music performers from a different music tradition. In the case of Greek folk music there are not enough digital archives or datasets. There has been another dataset consisting of MIDI files of Greek music though, which is called "Greek Music Dataset", or GMD (Makris, Karydis, & Sioutas, 2015). For the purposes of this essay, a new dataset was created.

Chapter 4: Practical Implementation

4.1: Dataset Description

The dataset that was created for the purposes of this project is named “Greek Folk Music Dataset”. It consists of 73 traditional tunes of Syrtos and Mpalos from the Aegean islands. These songs were transcribed using the open-source software MuseScore, after listening to online performances of the selected tunes and they were afterwards exported as mscz (sheet), wav (audio) and MIDI files.

The reason behind selecting MIDI files and non-automated transcription by ear is that the available sound files were neither enough nor appropriate for using a transcription program. Furthermore, there were important differences between the various performances for the same tune. Also, even though it could be possible if the files were enough to train a neural network to perform an automated transcription, many of the performances available online had too much noise and blurry sound, since they performed in fairs and there were sounds from people cheering, speech and clapping. This could be a future project, where the audio files found online could be cleaned and the instruments could be separated and then fed into an algorithm for an automated transcription. Still, often enough the instruments could play out of tune or there could be instruments that play microtonal melodies that are not easy to transcribe in a traditional music transcription program. Both the instrumental and the vocal music tradition of Greek folk music is characterized by the microtonal melodic production, therefore it would be an interesting attempt to develop a new tool designed specifically for this type of music. It should be mentioned though that there has been a tool for automated music transcription through pitch detection for example, such as similar research done in traditional melodies of Norwegian folk music (Lartillot, Thedens, & Jensenius, 2018) and a tool for microtonal melodic transcription of Turkish music (Benetos & Holzapfel, 2015). Additionally, both the vocalists and the instrument performers were improvising very often, since all the tunes are meant to be danced and improvised as well. This meant that, in order to distinguish these passing music events and freedom expressions, there should be a human analyst to set the limits while having a good understanding of the music, in order to achieve a transcription that is as loyal and neutral as possible. The element of improvisation could be observed in a later effort.

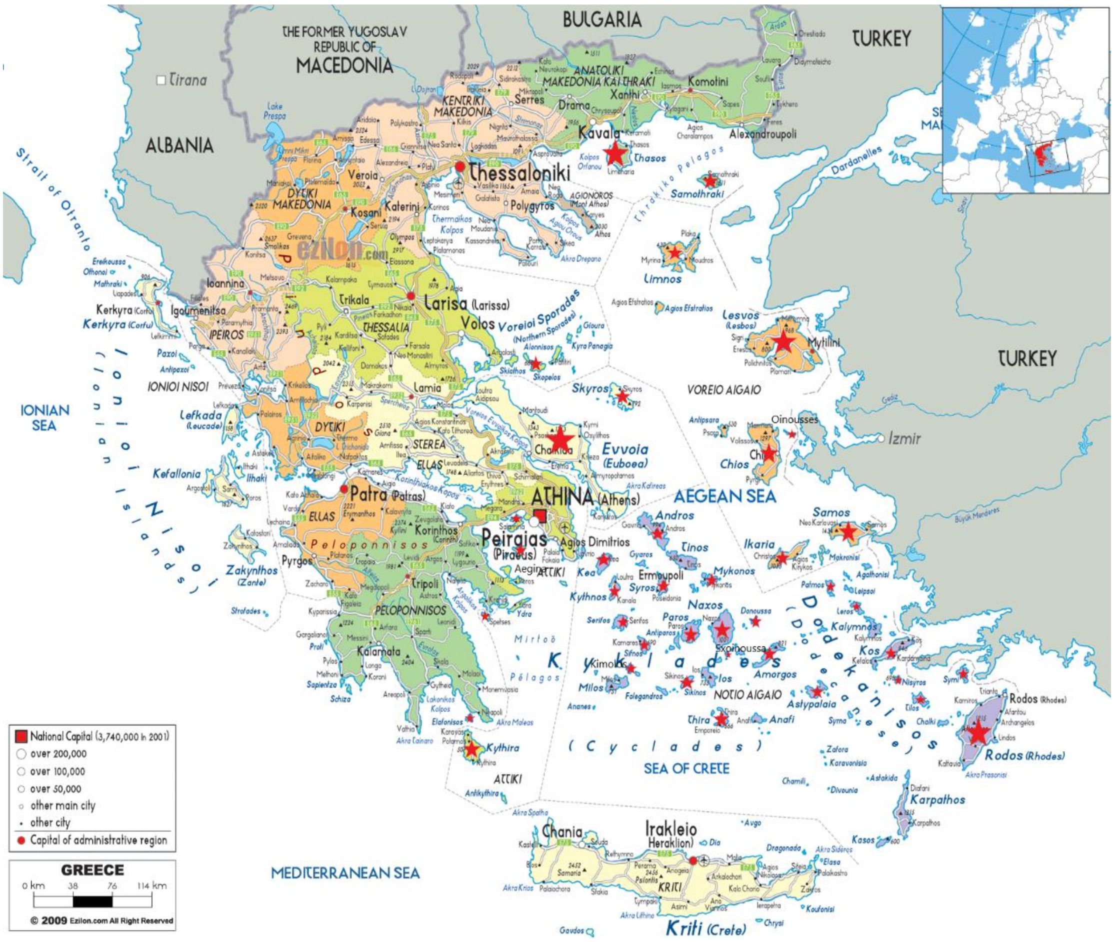
The music tunes that were selected were divided according to the dance type and were named according to the island from which they originate. It is crucial to mention though that for this type of music it is difficult to be absolutely confident about the specific origin of every song, especially since many islands share the same traditions and music styles. The information given from the MIDI files is pitch, duration, tempo, instrument type (piano, violin, or voice). The islands from which the songs were selected can be seen in table 1 and Figure 3.

Table 1 Dataset Description

Island	Island Region	Syrtos	Mpalos
Aegina	Saronic	1	1
Amorgos	Cyclades	1	1
Andros	Cyclades	1	1
-	Cyclades	1	2
Donousa	Cyclades	1	1
Elafonisos	Evia and surrounding islands	1	0
Evia	Evia and surrounding islands	1	1
Hios	NE Aegean	1	1
Ikaria	NE Aegean	2	1
Kea	Cyclades	1	1
Kimolos	Cyclades	1	0

Kos	Dodecanese	1	1
Kythira	Crete and surrounding islands	1	1
Kythnos	Cyclades	1	1
Leros	Dodecanese	1	0
Lesvos (Mytilene)	NE Aegean	1	1
Limnos	NE Aegean	1	1
Mykonos	Cyclades	1	1
Naxos	Cyclades	4	1
Nisyros	Dodecanese	1	0
Oinousses	NE Aegean	1	1
Paros	Cyclades	2	2
Patmos	Dodecanese	1	1
Rhodes	Dodecanese	1	0
Salamis	Saronic	1	1
Samos	NE Aegean	1	1
Samothrace	NE Aegean	1	1
Santorini (Thira)	Cyclades	1	1
Serifos	Cyclades	1	0
Sifnos	Cyclades	1	1
Sikinos	Cyclades	1	1

Skopelos	Sporades	1	1
Skyros	Sporades	1	1
Spetses	Saronic	2	0
Symi	Dodecanese	1	0
Syros	Cyclades	1	0
Thasos	NE Aegean	1	1



3Songs Map

As it seems from the table above, the island regions from which the songs were selected are six: Crete and surrounding islands region, Cyclades, Dodecanese, Evia and surrounding islands region, Saronic and Sporades. Some of the songs that originated from Cyclades region, but it is not known from which specific island, were put in the Cyclades region category, without specifying the island name.

One aspect that helped recognize the origins of the different tunes was most of the times the names. Of course, in most cases there is a certain name for the song that has nothing to do with its origin, but usually in instrumental tunes there is such a distinction. For example, there is Ikariotikos (or Kariotikos) Syrtos, which comes from Ikaria, Messaritikos Syrtos, which comes from Messaria, a town in Kythira Island, Marathokampitikos Syrtos from Marathokampos, Samos, and Platanisios Syrtos from Platana, Samothrace. Furthermore, some of the names that are given are due to the dancing style, such as Mantilianos (<mantili, μαντήλι = handkerchief) Syrtos from Kos Island. This dancing tune is also called Travichtos (<travo, τραβώ = pull), because the dancers hold each other from handkerchiefs and the leading dancer holds the second one from the left shoulder with his left hand, giving the impression of dragging the circle. As mentioned in the beginning, Syrtos dance is usually slower paced than Mpalos, due to the more dragging effect of the dancing style. One exception to this rule is Rhoditikos Syrtos (or Pidichtos < pido, πηδώ = jump), which is inspired from Cretan fast paced Syrtos dance.

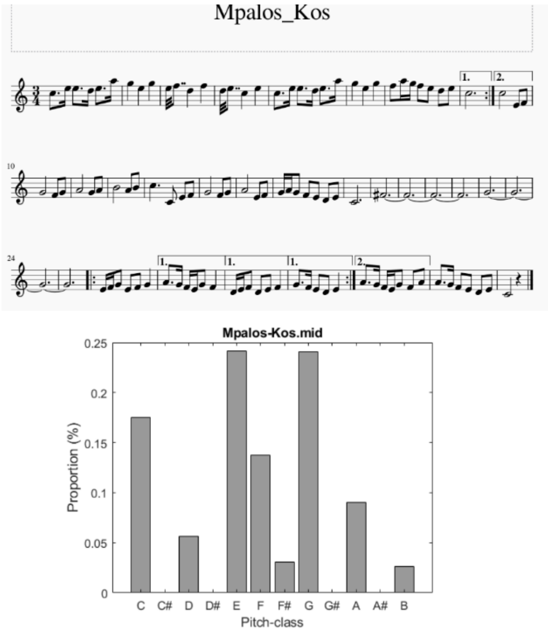
The same logic applies also to Mpalos dances. There are Mpalos tunes that are named after the region from which they originate, such as Koulouriotikos Mpalos, from Koulouri (Salamina) and Thermiotikos Mpalos from Thermia (Kythnos). The latter one though has also gotten a second name, due to the dancing style. More specifically, Thermiotikos Mpalos is also called Fourles, which is a name for spinning. Two really distinctive dance tunes that have special type of characteristics are Mpalos from Kos Island and Myloniatikos Mpalos from Evia. The first one, is the exception in terms of tempo since it is the only Mpalos dance with a rhythmic meter of 3/4 instead of 4/4. It is actually the oldest and most original form of Mpalos tune. Myloniatikos Mpalos from Evia is the only dance which stands in the line between Syrtos and Mpalos, something which is expected to be found later in the pattern analysis and clustering.

4.2: Feature Analysis

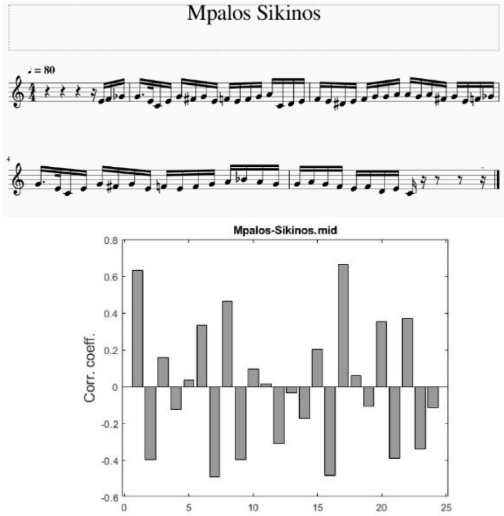
When trying to understand a dataset, in order to proceed to a motivic analysis, it is essential to identify what kind of information is present. Features are useful in this procedure. More specifically, there are some aspects in the data that can provide us with useful insights regarding their characteristics and measurable properties. The analysis of these aspects or so-called “features” is called “feature analysis” and it was one of the most important tasks in this project. Among the numerous features that were computed from the MIDI files were: pitch class distribution, key strength, onset distribution and onset autocorrelation.

Before proceeding to a more detailed description of these features and their analysis’ results, it is important to mention that for these computations, the MIDI toolbox was used (Toiviainen P. &., 2016). Starting from the pitch class distribution, it is a function that is used for the estimation of a tune’s key signature. This function calculates the proportion of a specific note within a tune. For example, as it seems in figure 4, on the x-axis there are twelve different notes, according to the chromatic pitch scale (C-C#-D-D#-E-F-F#-G-G#-A-A#-B), while on the y-axis there is a percentage which corresponds to each note’s proportion in the selected tune. In Mpalos tunes, there were some examples with a clear key signature indication: Mpalos Kos and Kythira (C major – dominant C-E-G), Mpalos Limnos (F major – dominant F-A-C), while others like Mpalos Hios who could indicate another key signature, but is actually another one (indicates D minor, but is F major). In Syrtos tunes, the most characteristic examples of clear key signature indication are: Syrtos Ikaria and Symi (D minor – dominant D-F-A) and Syrtos Sikinos (C major – dominant C-E-G).

Key strength is yet another function that was used in order to perform feature analysis. It is based on the K-S key-finding algorithm and it is useful for key profile characterization. As seen in figure 5, on the x-axis there are twenty-four different key profiles, twelve major and twelve minor, while on the y-axis there is a value for the correlation coefficients. More detailed, the twenty-four key profiles are correlated to the pitch class distribution of the tune, which is weighted according to the duration of each key profile. In picture 3, the most dominant notes are C, E and G, implying C major, which is clearly indicated in the score as well.

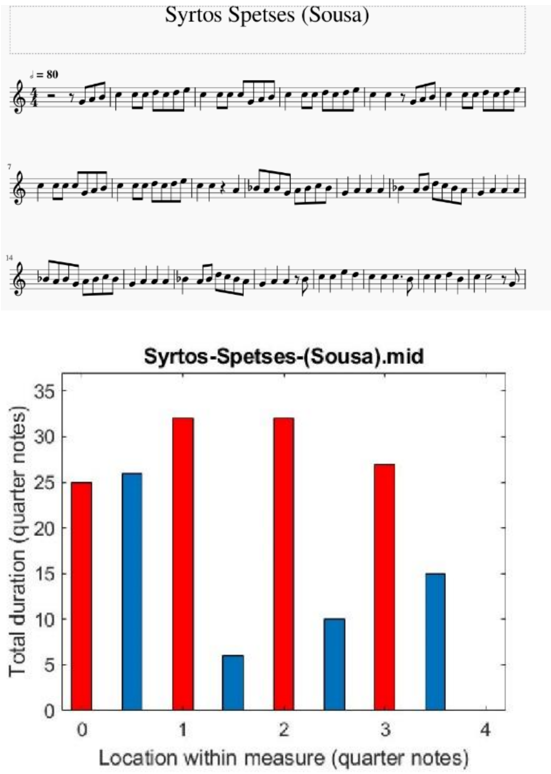


4 Pitch Class Distribution

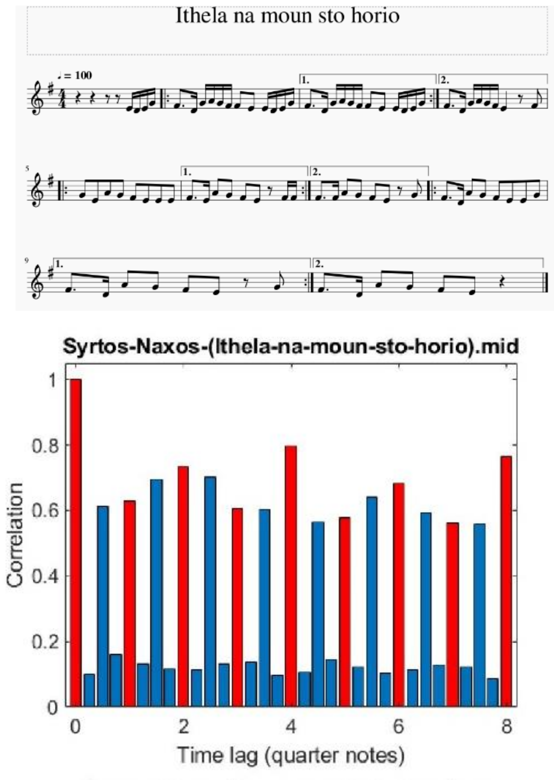


5 Key strength

Onset distribution is yet another feature that was computed using the MIDI toolbox. This feature corresponds to the beat distribution per measure, or in other words, how many quarter notes are there in every measure. Onset autocorrelation is a useful function for periodicity detection. In figure 7 there is such a periodicity detection per beat.



6 Onset Distribution

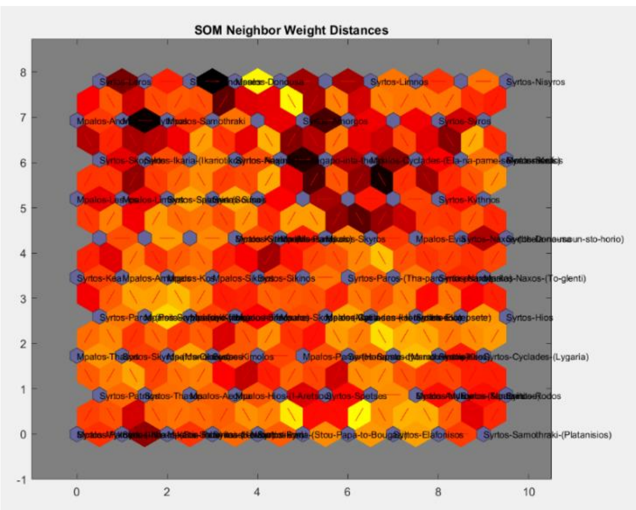


7 Onset Autocorrelation

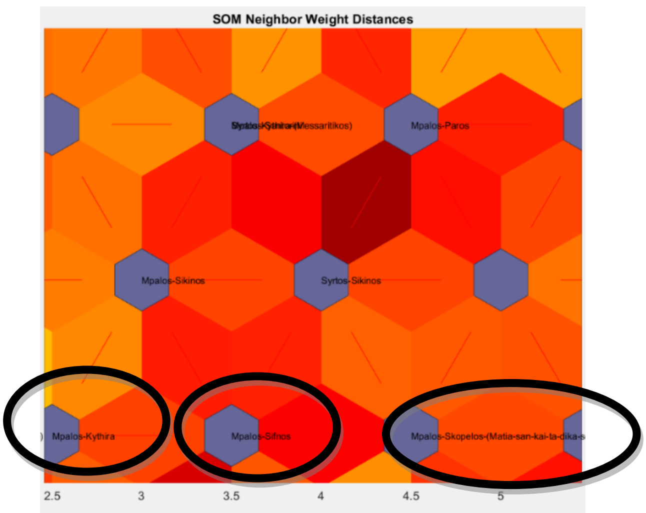
4.3: Unsupervised Clustering

Another form of better understanding the contents of a dataset is to cluster all data using unsupervised learning methods. Here, the two methods that were selected were Self-Organizing Map (SOM) and t-distributed Stochastic Neighbor Embedding (t-SNE). Both of them are dimensionality reduction techniques used to represent data distribution in a 2-dimensional space. The Self Organizing map is an artificial neural network, that reduces data dimensionality to display its distribution in a two-dimensional map. On the other hand, t-distributed Stochastic Neighbor Embedding is a nonlinear dimensionality reduction algorithm, following some statistics principals, such as probability distribution.

Here, Self-Organizing Map was created using nctool command in MATLAB, which is an open neural network clustering application. The input data that was selected for the example in figure 8 was a double array of 74 observations (74 tunes) and 24 features (24 key signatures). Map size was the default 10 by 10, and the training of the network was by default until reaching maximum number of epochs (Appendix 1). In the map of figure 11 neighbor distances were plotted. Each neuron corresponds to a tune and the neighbor patches have color shades from yellow to black, based on whether the neuron's weight vector is close to its neighbor one, or in other words, how similar the tunes are based on their key strength values. Similar key profiles were detected between the Mpalos tunes of Sifnos, Kythira and Skopelos, two of which actually partly share the same melody (Figures 8, 9 & 10).



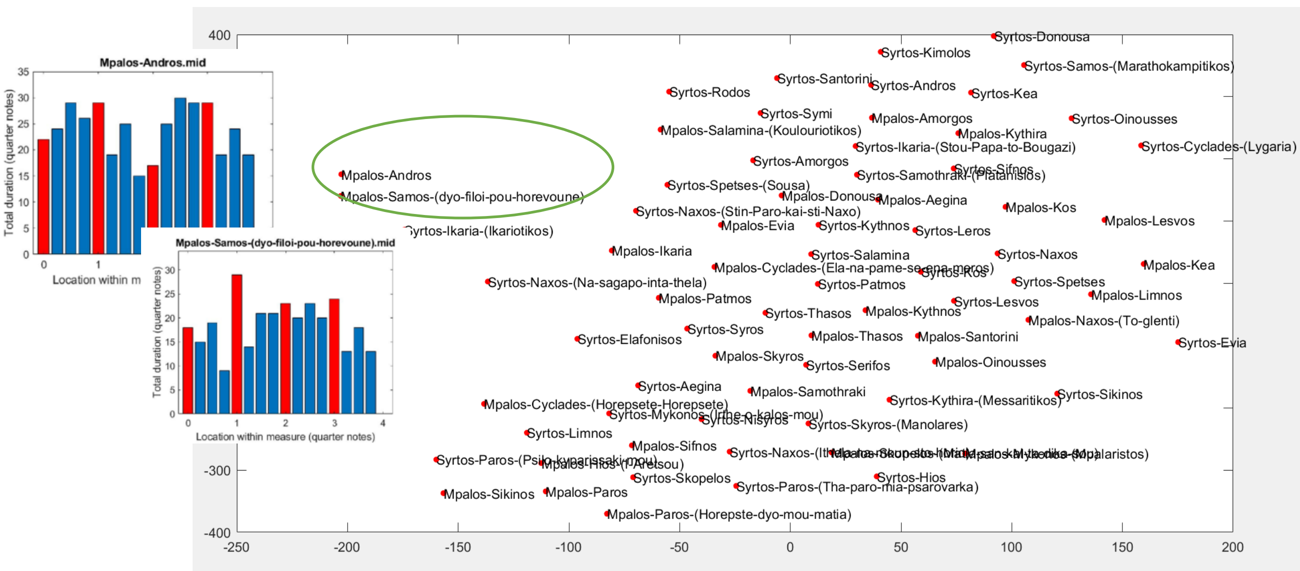
8 Self-Organizing Map (key strength)



9 Key strength Similarities

10 Key strength Similarities

The t-SNE map in figure 11 displays the tunes' distribution, based on their onset distribution. This map was also computed using the built-in MATLAB function. The input data was a double array of 73 observations (73 tunes) and 16 features (16 distribution values). In the example of figure 11 it is obvious that two of the tunes that were considered similar had indeed similar onset distribution values.



11 t-SNE map

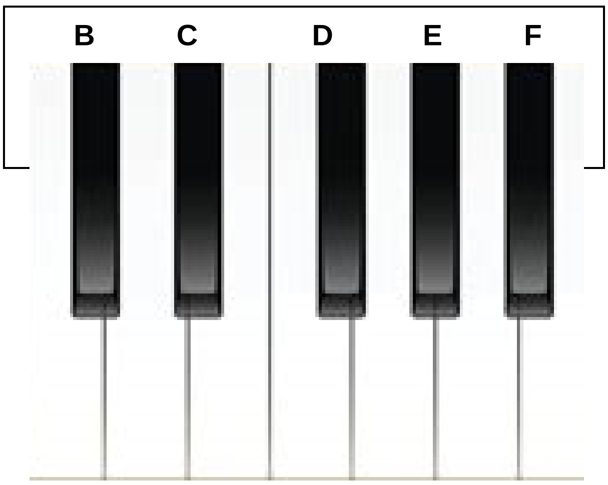
4.4: Pattern Discovery Algorithm

The algorithms that were used for the discovery of closed melodic patterns were developed by Olivier Lartillot as part of the project MIRAGE, adjusting a previously existing algorithm for pattern discovery (Lartillot O. , 2015) specifically for this new dataset. The focus on the closed patterns was essential for the implementation of a compact pattern description. As mentioned earlier, a closed pattern is a pattern that is enclosed in a larger one, but is repeated more often than this, so by detecting such motives we avoid information loss. There was a different algorithm developed for the discovery of patterns intra corpus, so patterns that exist in every tune individually, and another one for the discovery of patterns inter corpus, so across the whole dataset.

The basic files of this algorithmic process are called “analyze.m” and “analyzeMIDI.m” and they are MATLAB script files. In order to compute patterns for each file separately, one should call analyzeMIDI and provide information regarding filename, spell and beats per bar, using for example the following command if the tune is a Mpalos from Thasos Island:

analyzeMIDI ('Mpalos-Thasos.mid', [NaN,2, NaN, NaN,6], 4)

The spelling and beats per bar information were provided manually in a csv file. Spell indicates what type of accidental notes are there in the score since MIDI cannot provide this type of information. For example, as seen in figure 12, depending on the number placed in the sheet bars B to F, a note can be either sharp or flat. This information is useful for key signature declaration. Non-available spell means that there are no accidental notes present in the MIDI file. Beats per bar indicate time signature and it is 4 when time signature is two or four beats, 3 when time signature is three beats and 3.5 when time signature is 7 beats.

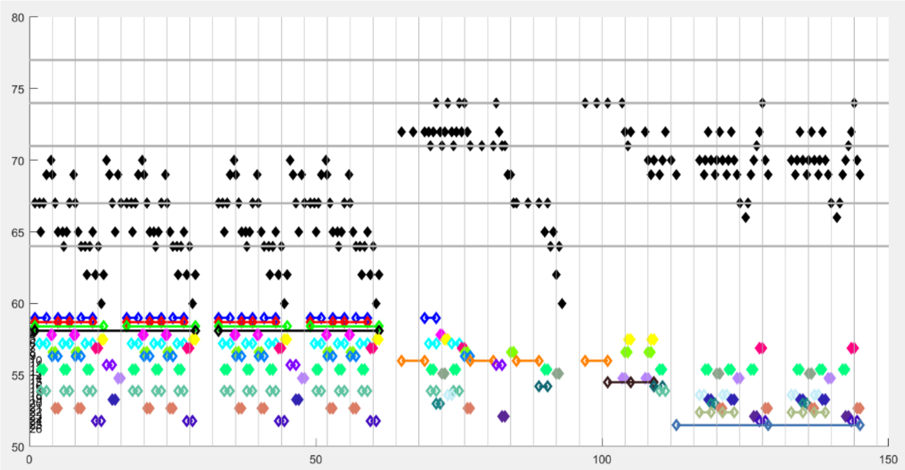


12 Spelling Information

The analyzeMIDI function also needs information such as patterns and file ID, which is provided by analyze function. Before proceeding into a more detailed description of analyzeMIDI, it is important to understand what happens in analyze function. First, it accepts as input a sequence of MIDI notes and creates a parametrical space with one dimension. Then, the goal is to create a pattern tree, so the algorithm checks whether the pattern sequence is empty and initiates the root of the pattern tree, while it also creates the "occurrence" of the root, from which all new pattern occurrences will extend. Every note is an occurrence of a note pattern, and these are used for the detection of interval patterns. Then, an empty sequence is initialized and for every element in this sequence, a pattern event is created.

For the analysis of the pattern sequences there are two main processes that happen in the algorithm. There is an attempt to extend any pattern occurrence which ends at the previous event, by memorizing the new note as an extension of the pattern event and checking whether this new event can extend the root occurrence and create a new pattern occurrence or not. In other words, there is a pattern recognition task during which the algorithm tries to detect if a pattern occurrence exists already and a pattern discovery task during which the algorithm tries to detect whether the pattern occurrence is new and if it can be extended into an occurrence of a new child of the pattern.

Moving back to analyzeMIDI function, the first thing that the algorithm is doing is to load the MIDI file and then after completing the analysis of the resulting patterns by calling analyze.m it provides the user with a display of these features, as seen in figure 13, an example of Koulouriotikos Mpalos from Salamina Island. On the x-axis there is the time value and on the y-axis the MIDI note value. There are five horizontal lines which resemble the five lines of the musical stave, a feature which makes the pattern observation and analysis easy to follow. Same patterns are depicted with the same colors, and it is visible that in a big pattern there can be many smaller ones, and they are indicated especially if they appear more times than the more general pattern they belong to. A major strategy of this approach is that the patterns are not only detected based on their pitch and interval distances, but also based on their metrical position.



13 Intra-corpus patterns – Salamina Island Example

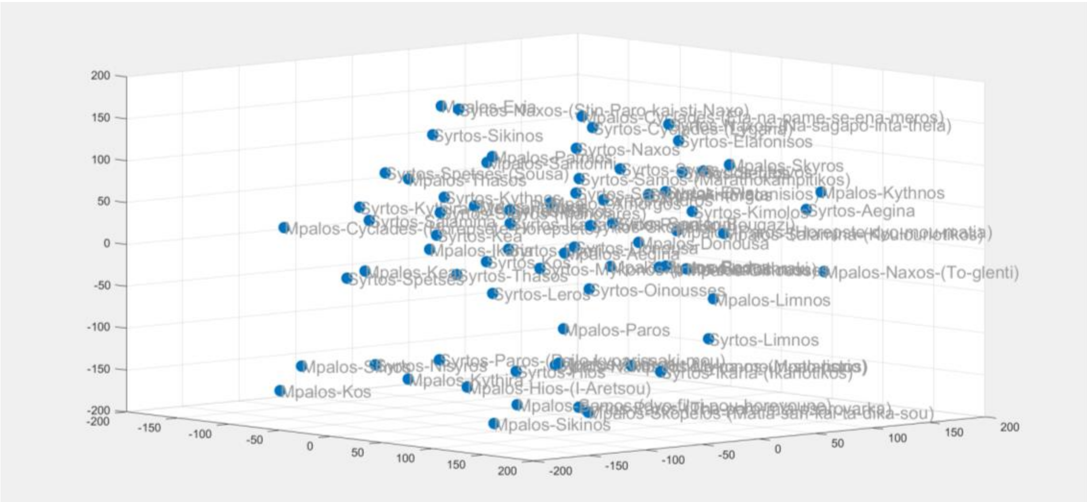
As mentioned before, patterns can also be discovered inter-corpus, between two or more files of the same dataset. In this project, there was an attempt to discover patterns in two separate folders, across Syrtos dance and across Mpalos, but also across the whole dataset. A new script was used for the purposes of this task, which used the key signature csv file, containing the relevant spell and beats per bar information for all the files in the dataset. The rest of the process follows a similar approach to the single-file pattern discovery and pattern display, but the patterns are displayed in text, as seen in figure 14. In the results, there is the pattern number given, based on the sequence it was found and then in which tune, in which beat it was detected and, in the parenthesis, it is indicated how many times was it detected.

```
Command Window
Pattern: b:2.5 ; d-1 b:3 ; d+1 b:3.5 ; d+1 b:0 ; d-2 b:0.5 ; d+1 b:1  (6)
TuneSyrtos/Syrtos-Naxos-(Na-sagapo-inta-thela).mid, Beat 102.5
TuneSyrtos/Syrtos-Naxos-(Na-sagapo-inta-thela).mid, Beat 110.5
TuneSyrtos/Syrtos-Naxos-(Na-sagapo-inta-thela).mid, Beat 118.5
TuneSyrtos/Syrtos-Naxos-(Na-sagapo-inta-thela).mid, Beat 126.5
TuneSyrtos/Syrtos-Syros.mid, Beat 10.5
TuneSyrtos/Syrtos-Syros.mid, Beat 26.5
== 748
Pattern: b:1 ; d+1 b:1.75 ; d-1 b:2 ; d-1 b:2.5 ; d-1 b:3 ; d-1 b:3.5  (7)
TuneSyrtos/Syrtos-Ikaria-(Stou-Papa-to-Bougazi).mid, Beat 5
TuneSyrtos/Syrtos-Ikaria-(Stou-Papa-to-Bougazi).mid, Beat 13
TuneSyrtos/Syrtos-Ikaria-(Stou-Papa-to-Bougazi).mid, Beat 21
TuneSyrtos/Syrtos-Ikaria-(Stou-Papa-to-Bougazi).mid, Beat 37
TuneSyrtos/Syrtos-Ikaria-(Stou-Papa-to-Bougazi).mid, Beat 45
TuneSyrtos/Syrtos-Ikaria-(Stou-Papa-to-Bougazi).mid, Beat 53
TuneSyrtos/Syrtos-Syros.mid, Beat 13
== 749
Pattern: b:3 ; d-1 b:3.5 ; d+1 b:0 ; d+0 b:0.5 ; d+1 b:1  (5)
TuneMpalos/Mpalos-Lesvos.mid, Beat 19
TuneMpalos/Mpalos-Lesvos.mid, Beat 55
TuneSyrtos/Syrtos-Syros.mid, Beat 31
TuneSyrtos/Syrtos-Syros.mid, Beat 39
TuneSyrtos/Syrtos-Syros.mid, Beat 47
fx >>
```

14 Inter-corpus patterns

4.5: Pattern Clustering

An important process for the analysis of the discovered patterns was the grouping of the resulted data. More precisely, a pattern clustering needed to be done, in order to gain a better understanding of the data distribution. Patterns are high-dimensional non-linear data, something which created the need for a simpler representation. This was achieved by unsupervised clustering and more specifically by the construction of a t-distributed Statistic Neighbor Embedding (t-SNE) map. Since this process was also helpful with understanding the data based on their computed features (key strength / pitch class distribution – based maps), it was decided that it makes sense to perform the same form of clustering in the patterns as well. One main difference in this approach is that first there was a similarity and dissimilarity matrix computed for all the tunes in the dataset and then the t-SNE map was computed based on these. Furthermore, a three-dimensional map was also created, in order to achieve a more comprehensive analysis of the resulting patterns (figure 15).



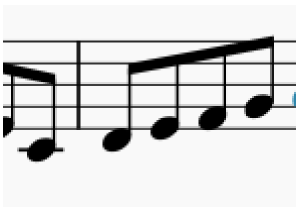
15 3D t-SNE map

Chapter 5: Results

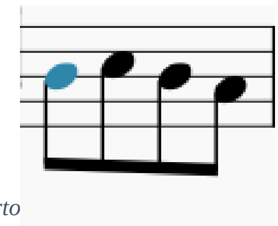
The total number of patterns discovered in the Greek Folk Music Dataset were seven hundred and forty-nine, based on the minimum pattern length, which was set in 3 notes. The minimum number of occurrences for a pattern was 2 times, while the most frequent pattern appeared two hundred and sixteen times (figure 16). The number of occurrences was computed across the dataset, which means that if a pattern was detected twice, then it was found in two different tunes. Most of the patterns that were detected belonged to both dance styles, while there were seven patterns detected only in Mpalos dance, with the most frequent Mpalos pattern in figure 17. Also, at least 23 patterns were detected only in Syrtos dance and the ones appearing more often are illustrated in figures 18 and 19.

Regarding the regions, there were islands who shared similar patterns for both of their dances, like Donousa, Ikaria, Aegina, Kea and Kythnos, while there were also common patterns between islands of Cyclades and Saronic. Region did not seem to be the major identifier, but it was in many island regions a common element. Furthermore, there have been multiple patterns detected between specific pairs of songs, indicating high melodic and rhythmic similarity. For example, multiple similar patterns were detected between Mpalos of Salamina Island and Syrtos of Ikaria Island, between the Mpalos of Samos, Andros and Skopelos and between the Syrtos of Samos and Mykonos.

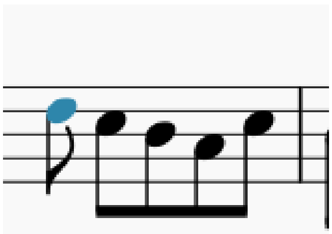
16 Most repeated pattern



17 Mpalos pattern

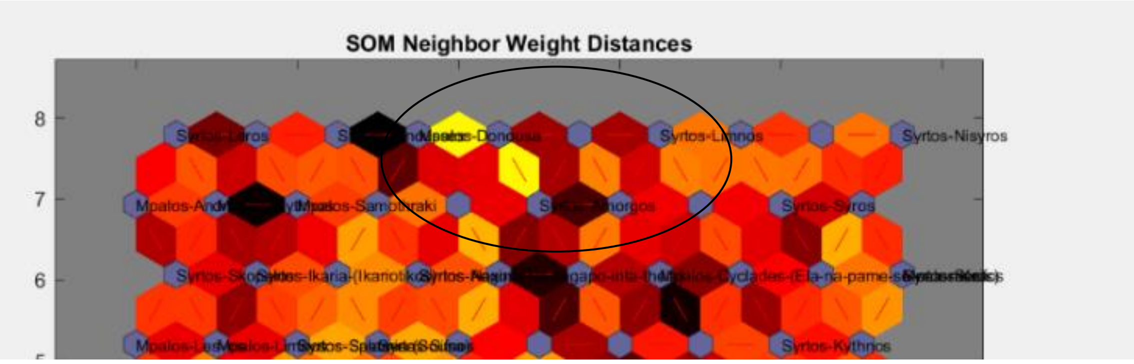


18 Syrto

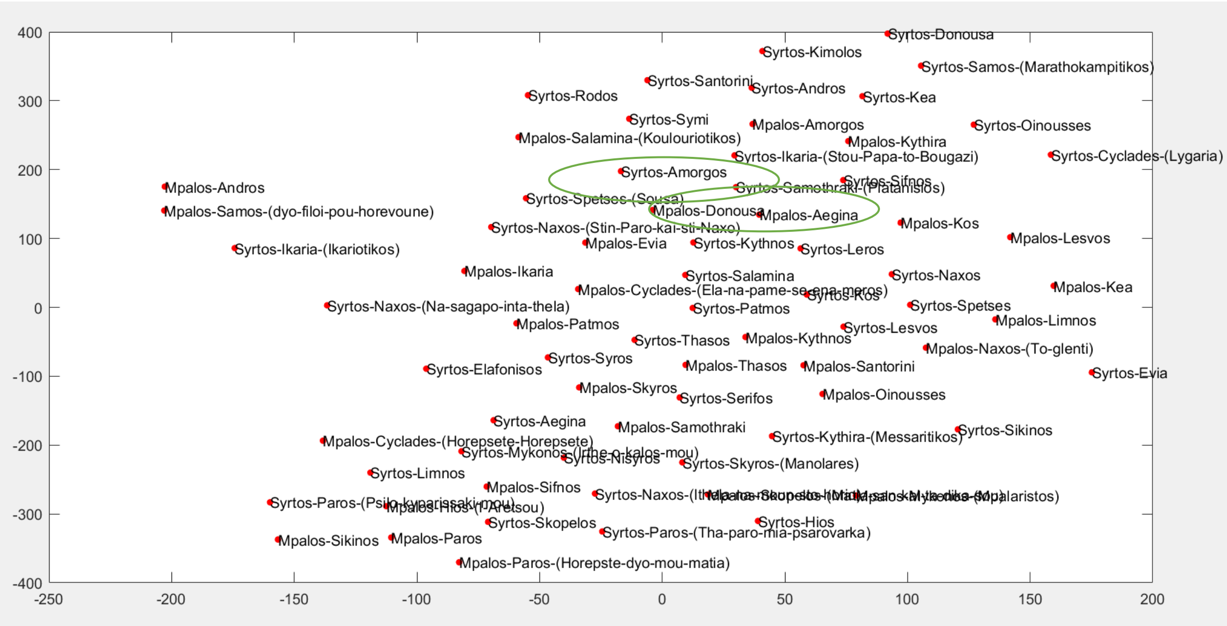


19 Syrtos pattern b

The melodic similarities that were detected from the patterns seem to appear also in the feature analysis results. For example, the similarity detected in figure 15 in the t-SNE map between the Mpalos tunes of Samos and Andros was also detected through the pattern analysis. Also, there were multiple similar patterns between Mpalos tunes of Samos, Skopelos and Kythira, something which makes sense, since it is known that they share parts of the same melody and this was also detected from key strength analysis. Furthermore, similar patterns that were detected between Mpalos tunes of Aegina and Donousa and Syrtos of Amorgos can be detected both in the SOM and in the t-SNE (figures 20 and 21).



20 pattern similarities detected in SOM



21 Pattern similarities detected in t-SNE

Chapter 6: Discussion

Computational music analysis is one of the most promising fields in musicology. It can provide a more objective music description, without including human errors in the execution process. Of course, there is a prerequisite for a basic technology background for the use and especially the development of such tools and there is without a doubt a large number of errors in the algorithm's decision making that the analyst or the developer should always have in mind. The various tasks and methods of Music Information Retrieval are the core of the future musicology and music analysis. Pattern recognition in specific can provide significant results, since through repetitions a musical piece or a musical style in general can be characterized and understood.

Earlier in this thesis there has been an analysis and pattern discovery in Greek folk music tunes of Aegean Syrtos and Mpalos dances. Both the feature and the pattern analysis proved that there are indeed numerous similarities between the tunes of the dataset, while at the same time important differences between the two dances and the various island regions. The feature analysis that was implemented was similar to Turkish makam analysis (Bozkurt, Ayangil, & Holzapfel, 2014), in the way that music was characterized and grouped based on pitch and rhythmical aspects, but the element of patterns was added to provide an extra level of information. Also, compared to the feature extraction and analysis in Finnish folk music (Toiviainen P. &., 2016) there was also a t-SNE map added to the distribution of the data apart from SOM.

Regarding pattern discovery and more specifically in Greek folk music (Conklin & Anagnostopoulou, 2011) the results here were closer to the human perception of repetitive patterns since the metrical position here was a main factor. There were not clear categories resulted from the patterns though, apart from some patterns that were discovered exclusively in one of the two dances. Supervised learning could not be applied in this task like it happened with Native American music (Neubarth, Shanahan, & Conklin, 2017) and RLCS algorithm (Dutta & Murthy, 2014) would not give the optimum results, since the interest was also between multiple tunes.

It is important to mention some important limitations in this attempt. First of all, the melodies were simplified, something that could mean important loss of melodic information. Furthermore, since the transcriptions were done manually and from online audio sources, there is always the concern for human mistakes and biases. Concerning the feature analysis, the wav files that were used were coming from MuseScore transcriptions, something that could also indicate major loss of audio information. On the other hand, live performances include noise and extra audio information that should in any case be discarded. Lastly, folk music origins can be sometimes quite questionable, since there are many tunes that are common between more than one island or the same tune can travel from one island region to another, obtaining some new features but keeping parts of its original shape, such as Mpalos tune from Skopelos and its variation from Kythira.

Chapter 7: Conclusions

Greek Folk Music Dataset is just the first step towards the computational analysis of Greek folk music. There could be the possibility to perform a feature and pattern analysis and the dataset of Greek folk music from Crete using these new techniques to compare the results. It would be of high interest to enrich and improve this dataset, first by replacing the online recordings with professional ones, possibly also advised by experienced ethnomusicologists in this music style. Second, more Greek regions could be added for Syrtos dance, such as Ionian islands, Peloponnese, and Macedonia, which could possibly offer a wider variety of melodic and rhythmic motives and structures. Third, gradually more folk tunes could be added, both from other Syrtos dance variations, and from other dances like sousta, karsilamas, hasapiko and more. This would gradually enable the analysis of the whole Greek folk music corpus and also create the possibility of a complete mapping of this music style.

In the current dataset, there could be plenty of other features computed. For example, there is the possibility to compute tempo directly from the original audio files, an element that could provide useful information regarding the overall temporal profile of the two dances. Also, the mode of each tune could be computed, by altering some aspects of the keystrength algorithm. It would also be quite interesting to test a feature analysis on the tunes, after transposing them in the same key, following the example of Toiviainen & Eerola, 2001.

Automated transcription of folk music and Greek music in specific is yet another future goal of this project. Microtonal melodies are really challenging to transcribe in a conventional transcription program and even though there has been some research in other folk music styles' transcription, such as Norwegian (Lartillot, Thedens, & Jensenius, 2018) and Turkish (Bozkurt, Ayangil, & Holzapfel, 2014), it would be interesting to explore the possibilities in Greek folk music. Instruments such as tsampouna and kanonaki for example could be detected from audio recordings and there could be interesting attempts to train a program to distinguish and transcribe the melodies that they produce.

Appendix 1.

SOM code:

```
%function y=mykeysom(x)
% Solve a Clustering Problem with a Self-Organizing Map
% Script generated by Neural Clustering app
% Created 30-May-2022 11:37:28
%
% This script assumes these variables are defined:
%
%   b - input data.

load export.mat %mat file containing keystrength data

x = b';

% Create a Self-Organizing Map
dimension1 = 10;
dimension2 = 10;
net = selforgmap([dimension1 dimension2]);

% Train the Network
[net,tr] = train(net,x);

% Test the Network
y = net(x);

% View the Network
view(net)

% Plots
% Uncomment these lines to enable various plots.
%figure, plotsomtop(net)
%figure, plotsomnc(net)

[rows,columns] = size(y);
pos = net.layers{1}.positions;
[l,m] = size(pos);
figure, plotsomnd(net, x)

for j=1:rows
for k=1:columns
    val = y(j,k);
    if val == 1
        point = [j,k];
        test = j;
        n = filenames{k};
        [filepath,name,ext] = fileparts(n);
        n = name;

        for w=1
        for z=test
            val2 = pos([w,w+1],z);
            text(val2(1),val2(2),0,n,'FontSize',8, 'Color', 'k')
        end
        end
    end
end
end
```

Bibliography

Ali, J., Khan, R., Ahmad, N., & Maqsood, I. (2012). Random Forests and Decision Trees. Retrieved from https://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.402.3863&rep=rep1&type=pdf

Ayres, J., Gehrke, J., Yiu, T., & Flannik, J. (2002). Sequential Pattern Mining using a Bitmap Representation . Retrieved from https://www.cs.cornell.edu/johannes/papers/2002/kdd2002-spam.pdf

Benetos, E., & Holzapfel, A. (2015). Automatic transcription of Turkish microtonal music. Retrieved from https://asa.scitation.org/doi/10.1121/1.4930187

Bozkurt, B., Ayangil, R., & Holzapfel, A. (2014). Computational Analysis of Turkish makam music: Music: Review of State-of-the-Art and Challenges. Retrieved from https://www.tandfonline.com/doi/full/10.1080/09298215.2013.865760

Breiman, L. (2001). Random Forests. Retrieved from https://link.springer.com/content/pdf/10.1023/A:1010933404324.pdf

Çataltepe, Z., & Altinel, B. (2007). Hybrid music recommendation based on different dimensions of audio content and an entropy measure. Retrieved from https://ieeexplore.ieee.org/abstract/document/7098941

Conklin, D. (2010). Discovery of distinctive patterns in music. Retrieved from https://www.researchgate.net/publication/220571586_Discovery_of_distinctive_patterns_in_music

Conklin, D., & Anagnostopoulou, C. (2011). Comparative Pattern Analysis of Cretan Folk Songs. Retrieved from https://dl.acm.org/doi/pdf/10.1145/1878003.1878015

Crochemore, M. L. (2009). Suffix Tree. Retrieved from https://www.researchgate.net/publication/255398622_Suffix_Tree

Crochemore, M., & Lecroq, T. (2009). Trie. In *Encyclopedia of Database Systems.* Retrieved from https://link.springer.com/referenceworkentry/10.1007/978-0-387-39940-9_1143

Dannenberg, R. B., & Hu, N. (2002). Pattern Discovery Techniques for Music Audio. Retrieved from https://www.researchgate.net/publication/2839455_Pattern_Discovery_Techniques_for_Music_Audio

Dapogny, A., Bailly, K., & Dubuisson, S. (2015). Pairwise Conditional Random Forests for Facial Expression Recognition. Retrieved from https://openaccess.thecvf.com/content_iccv_2015/papers/Dapogny_Pairwise_Conditional_Random_ICCV_2015_paper.pdf

Ding, S., Zhu, H., Jia, W., & Su, C. (2012). A survey on feature extraction for pattern recognition. Retrieved from https://link.springer.com/content/pdf/10.1007/s10462-011-9225-y.pdf

Dubnov, S., Assayag, G., Lartillot, O., & Bejerano, G. (2003). Using machine learning methods for musical style modeling. Retrieved from https://www.researchgate.net/profile/Shlomo-Dubnov/publication/2955985_Using_machine-learning_methods_for_musical_style_modeling/links/0c96052b390f04ee4d000000/Using-machine-learning-methods-for-musical-style-modeling.pdf

Dutta, S., & Murthy, H. (2014). Discovering Typical motifs of a raga from one-liners of songs in Carnatic music.

Eikvil, L., & Huseby, R. B. (2002). Pattern Recognition in Music. Retrieved from http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.97.6667&rep=rep1&type=pdf

Fedorenko, L. (2020). What Is Pattern Recognition in Machine Learning? Retrieved from https://topdigital.agency/what-is-pattern-recognition-in-machine-learning/

Gimel’farb, G. (n.d.). String Matching Algorithms. 3-7. Retrieved from https://www.cs.auckland.ac.nz/courses/compsci369s1c/lectures/GG-notes/CS369-StringAlgs.pdf

Jain, A., Murty, M. N., & Flynn, P. (1999). Data Clustering: A Review. Retrieved from https://dl.acm.org/doi/pdf/10.1145/331499.331504

Jesan, J. (2004). The neural approach to pattern recognition. Retrieved from https://dl.acm.org/doi/10.1145/991108.985625

Kim, T.-h. (2010). Pattern Recognition using Artificial Neural Networks: A review. Retrieved from https://link.springer.com/chapter/10.1007/978-3-642-13365-7_14

Kokash, N. (2005). An introduction to heuristic algorithms. Retrieved from https://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.105.8050&rep=rep1&type=pdf

Kotze, H. C., & Kuhn G., J. (1999). An evaluation of the Lempel-Ziv-Welch data compression algorithm. Retrieved from https://ieeexplore.ieee.org/abstract/document/129018

Kranenburg, P. V., & Conklin, D. (2016). A Pattern Mining Approach to Study a Collection of Dutch Folk Songs . Retrieved from https://arrow.tudublin.ie/cgi/viewcontent.cgi?article=1016&context=fema

Laney, R., Samuels, R., & Capulet, E. (2015). Cross Entropy as a Measure for Musical Contrast. Retrieved from https://link.springer.com/chapter/10.1007/978-3-319-20603-5_20

Lartilllot, O., & Toiviainen, P. (2007). Motivic matching strategies for automated pattern extraction. Retrieved from https://www.semanticscholar.org/paper/Motivic-matching-strategies-for-automated-pattern-Lartillot-Toiviainen/e7dec51228f209fa86f4529b7b2e9b8ae3f4f188

Lartillot, O. (2003). Discovering Musical Patterns through Perceptive Heuristics. Retrieved from https://jscholarship.library.jhu.edu/handle/1774.2/15

Lartillot, O. (2014). In-depth Motivic Analysis based on Multiparametric Closed Pattern and Cyclic Sequence Mining. Retrieved from https://zenodo.org/record/1418345

Lartillot, O. (2015). *PatMinr: General overview of the approach.* Retrieved from https://github.com/olivierlar/miningsuite/wiki/PatMinr:-General-overview-of-the-approach

Lartillot, O., Thedens, H., & Jensenius, A. R. (2018). Computational model of pitch detection, perceptive foundations, and application to Norwegian Folk Music. Retrieved from https://www.duo.uio.no/bitstream/handle/10852/71867/lartillot.pdf

Lippmann, R. P. (1989). Pattern Classification Using Neural Networks. Retrieved from https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=41401

Makris, D., Karydis, I., & Sioutas, S. (2015). The Greek Music Dataset. Retrieved from https://users.ionio.gr/~karydis/my_papers/MKarydisS2015%20-%20The%20Greek%20Music%20Dataset.pdf

Meredith, D. (2006). Point-set algorithms for pattern discovery and pattern matching in music. Retrieved from https://d-nb.info/991485998/34

Neubarth, K., Shanahan, D., & Conklin, D. (2017). Supervised descriptive pattern discovery in Native American music. Retrieved from https://www.tandfonline.com/doi/full/10.1080/09298215.2017.1353637

Nieto, O., & Farbood, M. (2014). Identifying Polyphonic Patterns From Audio Recordings Using Music Segmentation Techniques. Retrieved from https://scholar.google.com/citations?view_op=view_citation&hl=en&user=7CyUUcMAAAAJ&citation_for_view=7CyUUcMAAAAJ:t6usbXjVLHcC

Nuttall, T., Casado, M. G., Ferraro, A., Conklin, D., & Repetto, R. C. (2021). A computational exploration of melodic patterns in Arab-Andalusian music. Retrieved from https://www.tandfonline.com/doi/full/10.1080/17459737.2021.1917010

Orio, N., & Roda, A. (2009). A measure of melodic similarity based on a graph representation of music structure. Retrieved from https://www.researchgate.net/publication/220723409_A_Measure_of_Melodic_Similarity_based_on_a_Graph_Representation_of_the_Music_Structure

Pavlidis, T. (1977). *Structural Pattern Recognition .* Retrieved from https://books.google.no/books?hl=el&lr=&id=i-vrCAAAQBAJ&oi=fnd&pg=PA1&dq=structural+pattern+recognition&ots=BdVIBNh0-y&sig=0M3ADOtImcyAooUx01ZtXt_8F40&redir_esc=y#v=onepage&q=structural%20pattern%20recognition&f=false

Pikrakis, A., Theodoridis, S., & Kamarotos, D. (2002). Recognition of Isolated Musical Patterns Using Hidden Markov Models. In C. Anagnostopoulou, M. Ferrand, & A. Smaill, *Music and Artificial Intelligence: Second International Conference, ICMAI 2002 Edinburgh, Scotland, UK, September 12–14, 2002 Proceedings* (pp. 55-58). Retrieved from https://www.researchgate.net/publication/225129107_Recognition_of_Isolated_Musical_Patterns_Using_Hidden_Markov_Models

Schalkoff, R. J. (2007). Pattern Recognition. Retrieved from https://onlinelibrary.wiley.com/doi/abs/10.1002/9780470050118.ecse302?fbclid=IwAR38RRHxQ2rDO4Q-XrUM6cXJZ7JW1byxLkC4ceXduFrc_cAtAOECbRJT_hk

Schwanauer, S. M., & Levitt, D. A. (1993). *Machine Models of Music.* Retrieved from https://books.google.no/books?hl=en&lr=&id=CIFqbWcjfHEC&oi=fnd&pg=PA82&dq=music+pattern&ots=Mq2yEGHcAM&sig=v5-Qezc075b2ROuKLxJD_h_c6xg&redir_esc=y#v=onepage&q=music%20pattern&f=false

Sinith, M., & Rajeev, K. (2004). Hidden Markov Model based Recognition of Musical Pattern in South Indian Classical Music. Retrieved from https://www.dsprelated.com/Documents/sinith_ICSIP.pdf

Soni, D. (2018). Supervised vs. Unsupervised Learning. Retrieved from https://towardsdatascience.com/supervised-vs-unsupervised-learning-14f68e32ea8d

Toiviainen, P. &. (2016). *MIDI Toolbox 1.1.* Retrieved from https://github.com/miditoolbox/

Toiviainen, T., & Eerola, T. (2001). A method for Comparative Analysis of Folk Music based on musical Feature Extraction and Neural Networks. Retrieved from https://d1wqtxts1xzle7.cloudfront.net/66730481/2_2001-with-cover-page-v2.pdf?Expires=1655124358&Signature=FBcLZBKkrw2fGi1Q3-VGu0qBOslgSY61ARTCFiYLAkKygqp4556TynZHAj465Ip-o2wjeKtInyZoN~RR7XU2g-xDDTg1KJ4CN7Yr9eCEzmWwNieoYpLmRqgjXGealjWyCEaSYEPL2bbMLuc6EzZB9

Velankar, M., & Dr. Kulkami, P. (2017). Pattern recognition for computational music. Retrieved from https://www.researchgate.net/publication/322592611_Pattern_recognition_for_computational_music

Wang, C.-i., & Dubnov, S. (2015). Pattern Discovery from Audio Recordings by Variable Markov Oracle: A Music Information Dynamics Approach. Retrieved from https://acsweb.ucsd.edu/~chw160/pdf/ICASSP2015_vmo_theme_discovery_Wang_Dubnov.pdf

Xiaoqing, Y., Xueqian, P., Wei, Y., & Wanggen, W. (2010). Audio similarity measure based on Renyi's quadratic entropy. Retrieved from https://www.researchgate.net/publication/251979884_Audio_similarity_measure_based_on_Renyi's_quadratic_entropy

Xu, D., & Tian, Y. (2015). A Comprehensive Study of Clustering Algorithms. Retrieved from https://link.springer.com/article/10.1007/s40745-015-0040-1