MODELLING OF GAMAKAS FOR KARṆĀṬIC FLUTE MUSIC SYNTHESIS

Thesis

Submitted in partial fulfillment of the requirements for the degree of

DOCTOR OF PHILOSOPHY

by

RAGESH RAJAN M

DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING,
NATIONAL INSTITUTE OF TECHNOLOGY KARNATAKA,
SURATHKAL, MANGALORE -575025

OCTOBER, 2021

DECLARATION

I hereby declare that the research Thesis entitled **MODELLING OF GAMAKAS FOR KARṆĀṬIC FLUTE MUSIC SYNTHESIS** which is being submitted to the ***National Institute of Technology Karnataka, Surathkal*** in partial fulfillment of the requirement for the award of the Degree of ***Doctor of Philosophy*** in **Department of Electronics and Communication Engineering** is a bona fide report of the research work carried out by me. The material contained in this research thesis has not been submitted to any university or institution for the award of any degree.

RAGESH RAJAN M,
Reg. No. 145091 EC14F07,
Department of Electronics and Communication Engineering.

Place: NITK-Surathkal.
Date: 08/10/2021

CERTIFICATE

This is to certify that the Research Thesis entitled **MODELLING OF GAMAKAS FOR KARṆĀṬIC FLUTE MUSIC SYNTHESIS** submitted by **RAGESH RAJAN M** (Register Number: 145091 EC14F07) as the record of the research work carried out by him, is accepted as the Research Thesis submission in partial fulfillment of the requirements for the award of degree of **Doctor of Philosophy**.

Deepu

Dr. DEEPU VIJAYASENAN
(Research Guide)

18/10/21

Chairman-DRPC
(Signature with Date and Seal)

प्रोफेसर एवं विभागाध्यक्ष / PROF & HEAD
ई एवं सी विभाग / E & C Department
एन आई टी के, सुरतकल/N I T K, Surathkal
मंगलूर / MANGALORE - 575 025

Dedication

Akhaṇda maṇdalākāram
Vyāptam yēna carācaram
Tatpadam darṡitam yēna
Tasmai ṡrī guravē namah

Ajñāna timirāndhasya
Jñānānjana ṡalākayā
Cakshurunmīlanam yēna
Tasmai ṡrī guravē namah

"Salutations to the great guru, who elucidates the supreme truth which is infinite and all–pervading."
"Salutations to the great guru, who, by means of the medicated needle of knowledge, opens the eyes of people blinded by ignorance."

To my teachers, parents, in-laws, and my wife Shilpa.

Acknowledgements

At the very outset, I wish to express my profound gratitude and respect to my music teachers who initiated me to the eternal world of Karṇātic Music. Without the interest and love for music instilled by them, I would never have the courage to undertake such a laborious and challenging task as my research problem.

I am deeply indebted to my research guide Dr. Deepu Vijayasenan, who was the backbone of this entire research. From finding the problem to designing the solution, from understanding the concept to coding the algorithm, from the beginning to the very end, he stayed by me as a teacher, motivator, and the solution for all the problems. Every time I have failed or faltered in the course of this journey, he has patiently extended his support to get me back on track – every single time.

They say that a good teacher explains, a superior teacher demonstrates and a great teacher inspires. I wholeheartedly thank Dr. Arulalan M R, for being my best teacher for the lifetime. The ease with which he handled the subjects, the time he spent helping me when I almost gave up, and the positivity and sincerity he upheld even at difficult times will stay with me forever.

I am deeply thankful to Professor Michel Kocher, HEIG-VD, Switzerland for boosting my confidence in coding, especially in MATLAB. For someone like me, who was so diffident in coding and who had always found it difficult to learn, the immense help from such a humble, talented, patient, and great human being will always be regarded as a gift from God.

I use this opportunity to thank Professor Dr. Sumam David for the kind support she had extended in academic, research, and administrative matters whenever I needed help. I also thank Professor Dr. M.S. Bhat and Professor Dr. Laxminidhi T., the Head of the Department of Electronics and Communication Engineering, for their valuable suggestions and administrative support for the entire duration of this research.

I express my sincere gratitude to Dr. Basavaraj Talawar and Dr. Aparna P, who were the members of my RPAC. As the assessors and evaluators, they had always been positive and encouraging throughout the entire period of this research. I also extend my heartfelt thanks to all the teaching staff of the Department of Electronics and Communication Engineering at NITK.

I am truly blessed to have a group of loving friends at NITK across the departments. To mention a few, I am deeply grateful to Princy, Meryl, Deeksha, Shara, Mansoor, Sinan, and Akash, who were there with me when I was disheartened and dispirited. I also express my wholehearted thanks to my lab-mates Jayaram, Shareef, Vasu, and Sravani at SMILE lab, and all the other fellow researchers in the Department of Electronics and Communication Engineering.

I also thank Mrs. Prabha, Mrs. Soumya, Mrs. Amita, Mr. Sanjeeva Poojary, Mr. Guru Tilak, and Mr. Ratheesh from the Department of Electronics and Communication Engineering for the wholehearted support given to me for the entire period of my research.

I humbly, lovingly, extend my sincere thanks to my whole family who stood by me in all the difficult times, even though I was not available on several important occasions to fulfil my duties as a son, a son-in-law, a husband, and a brother. I will always be thankful to this bunch of highly positive and loving souls who always encouraged and supported me throughout this tiring but exciting journey. Words will never be enough to express my gratitude and love towards my wife, Shilpa, for being my support even under the immense pressure and mental strain she has endured during this long period.

Everything that belongs to me has been a gift from God Almighty. I thank God for the opportunity, strength and courage given to me to take up this research, the guidance, support and knowledge imparted to me to pursue it, and the blessings bestowed upon me to complete it.

Abstract

In this work, we propose a spectral model to efficiently synthesise Karṇāṭic bamboo flute music from the notes, duration, and *rāga* information of a song. Karṇāṭic flute music synthesis from basic notations is a challenging problem due to two major reasons. The first one is that the *gamaka*s are generally omitted from the musical notations in the tradition of KM. Hence, for the automatic synthesis of KM, the *gamaka*s associated with every note need to be predicted from the musical notations. The second reason is the continuously varying pitch contour of a note in the presence of *gamaka*s.

We propose a method to detect the presence and type of *gamaka*s associated with each note in a data-driven manner, from the annotated symbolic music alone. In this regard, we propose features based on the notes of the song. These features are used as inputs to a Random Forest Classifier (RFC). From our experiments, the accuracy values obtained for predicting the presence and type of *gamaka*s are ∼77% and ∼70%, respectively. These are significantly better than random classification accuracies. We also analyse the importance of neighbourhood of notes for the detection and classification of *gamaka*s. It is observed that the best accuracy is obtained for *gamaka* presence detection when a both-sided neighbourhood of size three is considered; and the best accuracy for *gamaka* type prediction is obtained with a both-sided neighbourhood of size one. The analysis performed on the training data reveals that there is information contained in these neighbourhoods for distinguishing between *gamaka* and non-*gamaka* notes.

For synthesising Karṅāṭic flute music, we model three different components of the flute sound, namely, pitch contour, harmonic weights, and time domain amplitude envelope. Cubic splines are used to parametrically represent these components. Subjective analysis of the results shows that the proposed method is better than the existing popular methods in terms of tonal quality as well as the propriety of rendering *gamaka*s. Hypothesis test results show that the observed improvements over other methods are statistically significant at 95% confidence interval.

Keywords: *Gamaka* , Karṅāṭic Music, note-based features, Random Forest Classifier, symbolic music, flute music synthesis, cubic spline interpolation.

Contents

List of Figures

List of Tables

Abbreviations

ANN Artificial Neural Network.

DT Decision Tree.

GMM Gaussian Mixture Model.

HM Hindusthāni Music.

KM Karṅāṭic Music.

KNN K Nearest Neighbours.

MOS Mean Opinion Score.

MTV Micro-tonal variations.

RFC Random Forest Classifier.

SMS Spectral Model Synthesis.

SVM Support Vector Machine.

List of Symbols

$A_i(t)$ Activation function for the noise corresponding to i^{th} note.

K Number of harmonics present in the original signal.

N Numberof discrete notes present in the middle octave..

$\phi_k(t)$ Time varying phase of the k^{th} harmonic of the original signal.

$\widehat{A}_k$ Harmonic weight of the k^{th} harmonic in the SMS method of synthesis..

$\widehat{E}(t)$ Time varying amplitude envelope of the synthesised signal.

$\widehat{K}$ Number of harmonics used for synthesising the signal.

$\widehat{\phi}_k(t)$ Time varying phase of the k^{th} harmonic of the synthesised signal.

$\widehat{a}_k(t)$ Time varying harmonic weight of the k^{th} harmonic of the synthesised signal.

$\widehat{f_0}(t)$ Time varying pitch contour of the synthesised signal.

$\widehat{n}(t)$ Time varying noise component in the synthesised signal.

$\widehat{s}(t)$ Synthesised signal.

$\widehat{s}_h(t)$ Harmonic part of the synthesised signal.

$f_0(t)$ Time varying pitch contour of the original signal.

$f_k(t)$ Frequency of the k^{th} harmonic of the synthesised signal.

$s(t)$ Original signal.

Chapter 1

INTRODUCTION

Music is one of the oldest art forms ever known to man. From the primitive to the modern era, it was an integral part of all human civilizations. Throughout the evolution of man as a social being, music has played an important role in communication, entertainment, and spiritual as well as socio-cultural conditioning. Studies suggest that the musical ability of a man is to be judged on the basis of his creative listening, rather than the ability to produce or compose any sort of music. From this perspective, human beings are inherently musical, since they possess the fundamental quality of being musical – the ability to perceive and differentiate sound classes. Music is a daily activity in man's life, regardless of the amount of musical training received by him (Blacking 1974, Rao 2012).

India has one of the oldest musical traditions in the world, dating back to the Vedic era. Indian music is one of the most complicated and intricate musical system compared to other musical traditions. In India, music evolved as a divine mode of communication for religious and spiritual observances. Music was a sacred art form for Indians since the gods and other celestial beings are regarded as exceptionally skilled musicians according to ancient Indian myths and epics(Shahinda 1914).

Music in India was originally taught and formulated by ancient saints, and their expositions were written in Sanskrit. As the Northern part of India underwent more cultural and political invasions thereafter, the musical tradition deviated from ancient treatises. The primeval Indian Music is more or less preserved in its original form in the southern part of India. Presently, the music of the north is called Hindusthāni Music (HM), while the musical tradition followed in the south is called Karṅāṭic Music (KM). Even though there are obvious differences in the singing styles as well as the melodic elements, one can observe that they have a lot of fundamental rules and key concepts in common, indicating their common lineage(Day 1891). In this thesis, we are focusing our attention on the intricacies involved in KM.

Musical instruments have a pivotal role in the evolution of music around the world, across civilizations. Instruments such as Chinese pipes led to the invention of different musical scales and even the arrangement of finger holes along the bore of a flute. Musical instruments were, to a great extent, responsible for developing the tonal perception of man. They produce the music in its true sense, surpassing the barriers of language (Sambamoorthy 1982).

The flute is one of the earliest musical instruments used by man (Conard *et al.* 2009). Based on the style of blowing, they are classified into three, namely, side-blown flutes, direct flutes, and vertical flutes. Flutes of different kinds are used in different musical traditions around the world. In the tradition of Indian music, side-blown bamboo flutes are used for solo performances and also as one of the major accompaniments for the vocal music concerts (Sambamoorthy 1982).

Karnatic flutes are made from bamboo and contain eight finger holes (tone holes) and one embouchure hole (Ramamurthy and Raghavan 2013). Sound is produced when the air jet from the player's mouth hits the edges of the embouchure hole and excites the air column inside the cylindrical body of the flute (Helmholtz 2013). The effective length of the air column can be adjusted by means of opening and closing the tone holes. This changes the resonance frequency, thereby generating different notes in the octave (Benade 1990). Higher or lower octaves of the same note can be generated by increasing or decreasing the blowing pressure, respectively (Helmholtz 2013).

1.1 Challenges

Musical instrument synthesis has been a topic of interest for several decades. A lot of models and techniques were invented to faithfully synthesize the sound of different musical instruments. These advancements have yielded excellent results in terms of the quality and naturalness of some musical instrument tones. But, still, there is a lot of scope for improvement in modelling continuously excited musical instruments such as flute (Pérez 2009).

Synthesizing Karnatic music in bamboo flutes is a challenging problem. This is mainly due to two reasons. The flute tone has a complex nature with the relative strengths of its different harmonics varying from time to time. The second reason is the continuously varying pitch contour in Karnatic music. *Gamaka*, the pitch bends used as essential ornamentations, is one of the important characteristics of KM. In the context of KM, notations contain only skeletal information such as the name of the note and its duration. No information on *gamaka*s is present in the KM notation. Hence, the presence and type of *gamaka*s need to be predicted and their continuous pitch bends need to be properly modelled for implementing an automatic music synthesis system.

1.2 Motivation

KM is extremely complex due to the Micro-tonal variations (MTV) called *gamaka*s applied to different notes. Furthermore, *gamaka*s need to be predicted from basic notations since they are generally omitted from the KM notations. Neither the prediction of *gamaka*s nor their modelling is addressed by the existing works in flute music synthesis. Moreover, certain *gamaka*s in KM cannot be reproduced on a bamboo flute due to the limitations imposed by its physical structure. Hence, a flute synthesis system can be thought of as an extension of the actual flute, overcoming its structural limitations. Such a system can be useful in applications such as concert synthesis. Also, this system enables a music enthusiast to generate music on the flute, irrespective of his proficiency in playing the actual instrument. This feature will be very useful for music composers.

1.3 Background of Karṅāṭic Music

As mentioned earlier in this chapter, Karṅāṭic Music (KM) is the musical tradition followed in the southern states of India, while Hindusthāni Music (HM) is the music tradition of the north. Both the streams rely on the fundamental concepts of *rāga* (roughly equivalent to the 'mode' in western music), *tāḷa* (rhythm), *ṡruti* (tonic), and *swara* (note). There are many similar *rāga* and *tāḷa* names shared by both of these traditions. But, the actual pitch variations in the *rāga*, the way of rendering the musical phrases, the styles of ornamentations used, and the measure of the *tāḷa* are usually different (Day 1891). In our discussions, we will be concentrating more on

KM and its characteristic features.

1.3.1 Swara (Swaram)

Sangītaratnākaram, a thirteenth-century treatise of Indian Music, defines *swara* as the one which is capable of inducing pleasure by itself. Thus, independent of all other factors, *swara* can generate the melody. Hence, it is treated as the elementary unit of melody in Karnatic Music (Sathyanarayana 2004).

Swara can be roughly translated as a note in the western music perspective. There are seven *swara*s in KM in one octave. They are: *Shadjam (Sa)*, *Ṙshabham (Ri)*, *Gāndhāram (Ga)*, *Madhyamam (Ma)*, *Panchamam (Pa)*, *Dhaivatam (Dha)* and *Nishādam (Ni)*. Except for *Sa* and *Pa*, all the other five *swara*s have multiple note positions (*swara sthānam*). The number of note positions in an octave is still a subject of discussion among musicologists. But, most of the Karnatic musicians follow the twelve–note system. In this, there are only 12 distinct note positions; but four of them are shared by eight *swara*s (same note position, but, different note label). Irrespective of the systems used to represent *swara sthānam*s, the scholars are unanimous about the variability of *swara*s from their respective note positions. In KM, each *swara* is articulated within a range centred about its note position. And, most of the time, an articulated *swara* will be sung with a less articulated or unarticulated *swara* on its either end (**?**). Hence, while the *swara sthānam*s are regarded as the technical note positions in an octave, *swara*s are treated as the melodic atoms in KM. Seven *swara*s, twelve *swara sthānam*s, and their frequency ratios with respect to *Sa* are given in table 1.1.

1.3.2 Rāga (Rāgam)

Rāga is one of the most important features of Indian Music that separates it from other musical traditions in the world. Ancient Indian musician and saint Mātanga defined *rāga* as the special series of musical sounds, ornamented with the notes, note sequences, and melodic sequences in such a way as to induce musical delight in the mind of the listener. *Rāga* is a melodic structure, formed by the characteristic arrangement of *swara*s. A *rāga* may contain four to seven *swara*s. It can also be visualized as the musical locus of a set of *swara*s (Sathyanarayana 2004).

Each *rāga* is governed by some rules, which restrict the use of certain *swara*s and

Table 1.1: Seven *Swara*s, twelve *swara sthānam*s, and their frequency ratios (James 2012)

Swara	**Representation**	**Ratio**
Shadjam	Sa	1
Shuddha Ṙshabham	Ri 1	$^{16}/_{15}$
Chathushruthi Ṙshabham / Shuddha Gāndhāram	Ri 2 / Ga 1	$^{9}/_{8}$
Shadshruthi Ṙshabham / Sādhārana Gāndhāram	Ri 3 / Ga 2	$^{6}/_{5}$
Anthara Gāndhāram	Ga 3	$^{5}/_{4}$
Shuddha Madhyamam	Ma 1	$^{4}/_{3}$
Prathi Madhyamam	Ma 2	$^{45}/_{32}$
Panchamam	Pa	$\frac{3}{2}$
Shuddha Dhaivatham	Dha 1	$^{8}/_{5}$
Chathushruthi Dhaivatham / Shuddha Nishādam	Dha 2 / Ni 1	$^{27}/_{16}$
Shadshruthi Dhaivatham / Kaishiki Nishādam	Dha 3 / Ni 2	$^{9}/_{5}$
Kākali Nishādam	Dha 3 / Ni 2	$^{15}/_{8}$

specify their patterns. These rules define the characteristic features like ascending pattern of *swara*s (*ārohana*), descending pattern of *swara*s (*avarohana*), and the *gamaka*s (Micro-tonal variations), which give that *rāga* its identity (called *rāga bhāva*) (Rao 2012, Nagavi and Bhajantri 2011, Day 1891). The similarity between the compositions based on the same *rāga* is because of the presence of these features. Musical sequences are so important that even the *rāga*s with the same set of *swara*s can have entirely different *rāga bhāva*. Adhering to the rules, a musician has all the freedom to use his skill, experience, and imagination to express his emotions (Day 1891).

1.3.3 Ornamentation In Karnatic Music

In KM, the ornamentation or embellishment is called *alamkāra*. The word *alamkāra* has two meanings, namely, adequacy and embellishment. In KM, both these meanings are realized by *alamkāra* in three different layers. The first one is called *swara alamkāra*, which adds beauty to a single note. The second layer is called *varṅa alamkāra*, which is a pattern of multiple notes. The third one is *sthāya*, which is the melodic figure of a *rāga* (Sathyanarayana 2004).

1.3.3.1 Gamaka (Gamakam)

Sangītaratnākaram describes *gamaka* as the "shaking of a note imparting pleasure to hearing and mind"(Shringy and Sharma 1978). The variable nature of *swara* is achieved by means of *gamaka*. *Gamaka*s are often translated as the ornamental inflexions given to the notes, which indirectly assigns an 'optional' nature to their use. In the context of KM, the shaking, curvature or the dynamics present in the pitch contour of a note are not ancillary elements used for mere beautification (Sathyanarayana 2004). Nor they are deliberately added to show off the musical knowledge of the performer. There are certain phrases in which the use of a particular type of *gamaka* is mandatory, while rendering some notes. In such situations, *gamaka* is not at all an optional embellishment; but the integral part of the *swara* itself. Thus, it demonstrates both the meanings of the word *alamkāra*. Hence, *gamaka* is considered as the *alamkāra* of *swara*. There are very few occasions where the musician has all the freedom to choose the type of *gamaka*, or can even decide whether to use them or not. Only in such cases, *gamaka*s can be thought of as the 'optional' ornamental elements.

The shaking or curving of oscillating a note can be performed in almost infinite number of ways. The methods to impart this variability is subjective, depending on the level of musical knowledge and the musical tradition followed by the performer and also the style of the composition. *Sangītaśāstra*, an ancient musical treatise, points out that listing all the various possibilities in *gamaka*s is practically impossible. Different treatises have tried to classify *gamaka*s into broader categories instead of listing the entire set of possible variations. Some of the *gamaka*s are specific to vocal music, and some others are specific to musical instruments such as *vīṅa* (Sathyanarayana 2004). The *Mahābhārata C̀ooḍāmāṇi* speaks about 10 types of *gamaka*s (Krishna and Ishwar 2012), namely, *ārōhaṇam*, *avarōhaṇam*, *āhatam*, *pratyāhatam*, *sphuritam*, *tripuc̀c̀ham*, *dhālu*, *kampitam*, *āndōḷam* and *moorc̀c̀hana* (James 2012). Subbarāma Dīkshita, has also classified *gamaka*s into ten broad categories in his work *Sangīta Sampradāya Pradarśini* (Dikshitar 2008).

1.3.4 Tāḷa (Tāḷam)

Tāḷa is the rhythmic framework in KM. It divides time into fixed and equal length segments called *āvarta*s. Each *āvarta* can be subdivided into smaller time units. Tempo of the entire musical piece can be changed by changing the duration of these smaller units. *Tāḷa* decides duration and onset of each and every syllable and *swara* (Sathyanarayana 2004). There are different *Tāḷa* arrangements in KM, each having a unique set of *āvarta*s. Hundreds of *tāḷa*s are mentioned in the ancient treatises. In the modern Karṅāṭic Music, 35 different *tāḷa* arrangements are used (Day 1891).

1.4 Objective

- To predict the presence of *gamaka* associated with a note from the textual information containing the note labels, note durations, and the *rāga* information of a song.
- To predict the type of *gamaka* from the textual information of a song.
- To synthesise bamboo flute tone of a plain note by modelling the frequency contours, amplitude envelope, and wind noise by means of a modified sinusoids plus noise model.
- To synthesise bamboo flute tone for *gamaka*s and non-*gamaka* transitions by means of modelling the continuous–time behaviour of the frequency contour, amplitude envelope, and wind noise.

1.5 Major Contributions of the Thesis

The main contributions of the thesis are summarized in this section.

1. A system for predicting the presence and type of *gamaka*s from the skeletal notations is proposed. The input to this system is a text file containing information regarding notes, their duration and the *rāga* of the song. Random Forest Classifier is used for prediction and classification. The proposed system proves to be effective in detecting and classifying *gamaka*s from the textual notations alone.

2. A system is proposed to synthesize bamboo flute tones by extending the sinusoidal model. Pitch, its other harmonics, spectral envelopes, and time domain envelopes are modelled for generating the flute tone.

3. A system for synthesizing the continuous pitch contours in the context of Karṅāṭic Music (KM) is proposed. Towards this, a continuous–time model for the frequency contours, spectral envelopes for the harmonics, and time domain amplitude envelope is developed based on cubic spline interpolation. This system is able to generate Karṇāṭic flute music involving different *gamaka*s and non-*gamaka* transitions.

1.6 Organization of the Thesis

In Chapter 1 we try to brief the background of *gamaka*s in Karṅāṭic Music and the need for a system to predict and model the *gamaka*s. Chapter 2 discusses the existing methods for the synthesis of flute music and related literature. In Chapter 3, we try to give a detailed description of the prediction and classification of different types of *gamaka*s. In Chapter 4 a new model is proposed for synthesizing the plain notes, *gamaka*s and non-*gamaka* transitions for Karṇāṭic flute music. The thesis is concluded in Chapter 5.

Chapter 2

LITERATURE REVIEW

The objective of this thesis can be broken down into two sub-problems. The first one is predicting the *gamaka*s from the input textual information, and the second one is to synthesise flute music corresponding to the input notes and the *gamaka*s. In this chapter, we present an overview of the existing literature related to the sub-problems mentioned above.

2.1 Flute Music Synthesis

Two major approaches towards the synthesis of flute tones are discussed in the literature – physical models and spectral models. While the physical models are based on the sound production mechanism of the flute, spectral models rely on the perception of sound by the human ear (Smith III 1991).

Physical models for flute dates back to the early 1980s, where the model comprised of an energy source, an energetically active non-linear element, and an energetically passive linear element. The non-linear element was used to model the air jet and the linear, element represented the bore of the flute. A delay is incorporated in the non-linear element to account for the dependence of the frequency of sound on the blowing pressure (McIntyre *et al.* 1983). Later on, this model was improved by adding the dispersive effect of finger holes on the air column inside the flute bore.

A real-time implementation of the flute model was proposed in which filters were used to model the reflection, dissipation, and losses inside the flute body. Effect of over-blowing and vibrato were also modelled (Valimaki *et al.* 1992). A transmission line model for the transverse flute was developed which modelled six finger holes. But, the partial closing of the holes was not modelled (Keefe 1990). Later on, a digital waveguide model was proposed, which modelled only the first two or three open finger holes (Välimäki *et al.* 1993). Later, the system was extended by modelling 15 finger holes (Välimäki *et al.* 1996).

A distributed tone hole model using the digital waveguides was also proposed which modelled the open, partially open and closed tone holes in real time (Scavone and Cook 1998). The model was again improved by modelling dispersion and dissipation inside the bore, keypad noise, vibrato, and tremolo (Ystad and Voinier 2001).

A filter design based approach was proposed to synthesize Indian bamboo flute tones (Ramamurthy and Raghavan 2013). For each group of the notes, spectra of individual notes were combined to generate a composite spectrum, and the coefficients were found out. Attack and decay portions were not modelled.

Spectral models date back to the late 1960s. A 700 millisecond-long flute tone (along with other wind instrument tones) was generated by means of spectral analysis method (Strong and Clark 1967) using the weighted sum of 30 sinusoids. Later on, another additive synthesis method called Spectral Model Synthesis (SMS) based on overlap–add method was developed, which modelled the spectrum as the sum of deterministic and stochastic components (Serra *et al.* 1997, Serra and Smith 1990). Time domain envelope was not explicitly modelled.

SMS model was, later, improved by adding a provision to model transients in addition to sinusoids and noise (Verma and Meng 2000). The basic sinusoidal model was improved by modifying the pitch and harmonic magnitude (Suyun and Yibiao 2016). Instead of using the peaks from the spectrum directly, the amplitude values in the gaps between the peaks were estimated by fitting a cubic spline between all the peaks. Another improved version of SMS incorporated the noise part into the sinusoidal part itself (Kreutzer *et al.* 2008). The harmonic amplitude envelope for the entire note was modelled using a sixth–degree polynomial.

A contiguous group synthesis approach for synthesising Chinese flutes was developed based on the grouping of harmonics (Horner and Ayers 1998). The amplitude envelope for group 1 was designed using line segment approximation, and the powers of this envelope were used for the other groups. Later on, this method was modified for synthesizing trills (Ayers 2003) and tremolos (Ayers 2004). the frequency of the trill was modelled as a line segment approximation of the average frequency contour shape of trills. An amplitude envelope using line segments was also proposed. This was again improved by creating a timbre database and adding provisions for changing the attack and decay rates (Rocamora *et al.* 2009).

Different Chinese flutes were modelled using additive synthesis making use of around thirty harmonic components (Ayers 2005). In additions to trills and tremolos, vibrato also was modelled. A harmonic band wavelet transform based method was developed to model the breath noise of a flute sound as pseudo periodic 1/f-like noise (Polotti and Evangelista 2001; 2007). Synthesis of Andean quena tones was also performed based on this method (Dïaz and Mendes 2015).

A method for modelling and synthesizing *gamaka* was proposed for Karnatic music (Subramanian 2013). Automatic addition of *gamaka*s for some popular songs was implemented. For other songs, the user needed to manually specify the constituent notes and their time durations involved in each *gamaka*. Synthesis was done using contiguous group synthesis. Amplitude envelope and the attacks and decays were not modelled.

Another work synthesized *gamaka*s in Karnatic flute music using a modified harmonics plus noise model (Ashtamoorthy *et al.* 2018). *Gamakas* were approximated using combinations of decaying and increasing exponential functions. Only three types of *gamaka*s were synthesized. A common amplitude envelope having a predefined attack, sustain and decay characteristics was designed for all the notes.

Most of the methods discussed in the literature were developed for synthesizing isolated notes. Synthesizing complete songs with automatic addition of ornamentations was not achieved in most of the methods. Moreover, very few works focused on the special features of Karnatic music. Most of the methods approximated the pitch contour as a discrete set of constant pitch segments. This violates the essential characteristic of Karṅāṭic Music, which is the continuously varying pitch contour. Automatic modelling and synthesis of the *gamaka*s are discussed in only one paper (Ashtamoorthy *et al.* 2018). This work addresses only three *gamaka*s, that too, as a coarse approximation of the actual shape. This approximation will not be sufficient enough to model other *gamaka*s used in KM.

2.2 Ornament Predicion

For any system to synthesize Karṇāṭic flute music, *gamaka*s need to be properly addressed. Our goal is to synthesize flute music from the skeletal notations alone, which contain only the note names, note duration, and the *rāga* information. Such an end-to-end system needs a provision to predict the presence and type of *gamaka*s associated with each note.

There are very few works done on predicting the *gamaka*s in Karṅāṭic Music (KM). To the best of our knowledge, all the existing works on *gamaka* analysis focus on acoustic data for this task (Vyas *et al.* 2015, Narayan and Singh 2014, Gupta and Rao 2011, Pratyush 2010, Miryala *et al.* 2013). Other works on Indian Classical Music (ICM) such as automatic music composition and singing voice synthesis do not address the problem of predicting *gamaka*s from the symbolic music (Varadharajan *et al.* 2014, Mohapatra *et al.* 2010, Sinha 2008, Das and Choudhury 2005, Arora *et al.* 2009, Belle *et al.* 2009, Viraraghavan *et al.* 2017, **?**, Ranjani *et al.* 2017). In Western music, the studies performed on detection and analysis of ornamentations as well as expressive music performance use audio data for these tasks (Gómez *et al.* 2011, Giraldo and Ramírez 2016, Giraldo and Ramirez 2016).

The ornamentation for Jazz guitar performances was predicted as a part of expressive music synthesis (Giraldo and Ramírez 2016). The dataset contained 16 commercially available songs performed by an artist and their score files. Pitch and duration of the current note and neighbouring notes along with the perceptual parameters, key, and chord were extracted as features. RIPPER algorithm (Cohen 1995) used with a reduced feature set gave the best prediction accuracy of around 70%, which was only 3.5% better than a random classifier. Later, the dataset was increased to 27 songs, and the machine learning techniques such as Support Vector Machine (SVM), Artificial Neural Network (ANN), Decision Tree (DT) and K Nearest Neighbours (KNN) were used for classification (Giraldo 2016). Decision Trees yielded the best accuracy of around 79%, which was an improvement of only 5% over random classifier.

2.3 Research Gap Analysis

- To the best of our knowledge, predicting the presence and type of *gamaka*s from the textual notations alone has not been attempted so far. All the existing *gamaka* prediction systems rely on the audio data for performing this task. It is not practical to input the audio information in problems such as music synthesis or automatic music composition, where the objective of the system is to generate audio files.

- Except one, all the existing works in flute synthesis do not consider the *gamaka*s and the continuous nature of pitch contour in Karṅāṭic Music (KM). Discrete approximation of the pitch contour will not be adequate for any system to properly synthesize a song in the KM perspective. A continuous time model is needed to address this problem.

Chapter 3

GAMAKA PREDICTION AND CLASSIFICATION

One of the major differences between Karṅāṭic Music (KM) and western classical music is the use of Micro-tonal variations (MTV). MTV fall in the category of ornamentations and embellishments in the western music perspective. In KM, such tonal variations are called *gamaka*s. They tend to alter the pitch of a note from its actual position. This variability of note adds naturalness to the rendition in the perspective of KM. *Gamaka*s are considered as the integral parts of melody in KM (Krishnaswamy 2003). KM performances are predominantly improvisational and the usage of *gamaka*s varies considerably across musicians. For a particular melodic structure, there can be multiple correct usages of *gamaka*s for the same series of notes. It is not practical to list out all the musically correct combinations of *gamaka*s for any song. Hence, it is a common practice to omit the *gamaka* information in the musical notation (Viswanathan 1977).

This poses a serious problem in applications such as automatic music composition and synthesis of KM, as a note in KM is properly defined only when the associated *gamaka*s are taken into consideration (Viswanathan 1977), (Sambamoorthy 1964). A considerable amount of musical expertise is required to add the most accurate and aesthetically pleasing *gamaka* to the plain notes. For the applications such as music synthesis, audio will not be available as input. Hence, for designing a stand–alone system for the synthesis/composition of KM, the *gamaka*s need to be properly predicted from the musical notations alone, without any audio input. We try to predict the presence and type of *gamaka*s from the musical notations in a data–driven manner, from the plain note information.

3.1 Challenge

The rules to assign a particular type of *gamaka* to a note in KM are neither too stringent, nor too flexible (Subramanian 2013). For example, in certain melodic frameworks, a performer is allowed to add a *gamaka* of his choice to a note. In some other cases, the use of specific *gamaka*s is mandatory. The choice of a *gamaka* varies with the fundamental melodic structure of the song and also with the performer's musical expertise. The ability to decide the most aesthetically pleasing *gamaka* from the available alternatives develops with the experience and the knowledge about the melodic structure of the song (Swift 1990). Since this is challenging for even the human singers without much musical expertise, this is a difficult problem in the machine learning viewpoint (Gayathri 1987).

3.2 Dataset

Over the years, musicologists have described and classified *gamaka*s in different ways (Swift 1990). One of the widely accepted textbooks for *gamaka*s in the twentieth century is Sangīta Sampradāya Pradarshini (SSP). This book classifies *gamaka*s into ten, based on the playing techniques of *Vīna* (an Indian musical instrument) (Dikshitar 2008). We use this as the reference book for creating dataset for our experiments. We choose two popular melodic structures (*rāga*s), namely, *Kalyāni* and *Shankarābharanam*, and their *derived rāga*s[1] for this dataset.
We digitize the musical information of all the songs belonging to these *rāga*s from SSP. The dataset covers 80 songs across nine composers from 22 *rāga*s. Table 3.1 lists all the 22 *rāga*s in the dataset and the number of songs in each of them.

[1]A *derived rāga* contains only a subset of notes that are present in its parent *rāga*(Sambamoorthy 1964).

Table 3.1: Number of songs in each *rāga* in the dataset

Parent *Rāga*	*Rāga*	Training Set	Seen *rāga* Test Set	Unseen *rāga* Test Set	Total Songs
Kalyāṇi	Kalyāṇi	–	–	9	9
	Mōhanam	–	–	5	5
	Sāranga	–	–	4	4
	Yamunā kalyāṇi	–	–	5	5
	Hamvīru	–	2	–	2
S̀ankarābharaṇam	S̀ankarābharaṇam	10	4	–	14
	Ārabhi	2	1	–	3
	Sāma	2	1	–	3
	Pūrvagauḷa	1	–	–	1
	Nāgadhwani	1	–	–	1
	Kurinji	2	1	–	3
	Hamsadhwani	1	1	–	2
	Bilahari	3	1	–	4
	Nārāyaṇi	1	1	–	2
	Bēgaḍa	2	1	–	3
	Dēvagāndhāri	2	2	–	4
	Kēdāram	1	1	–	2
	Navarōj	1	1	–	2
	Nīlāmbari	3	2	–	5
	Pūrṇachandrika	1	2	–	3
	Saraswathimanōhari	–	2	–	2
	S̀uddhavasantham	–	1	–	1
	Total	33	24	23	80

A total of around 30000 notes are present in the dataset, out of which roughly 20000 do not contain any *gamaka* while the remaining 10000 notes contain *gamaka*s. Even though ten distinct classes of *gamaka*s are found in the dataset, we do not include two of them due to the lack of samples. The eight different *gamaka*s considered in our experiments are: *Podi*, *Kampitham*, *Sphuritham*, *Nokku*, *Ētra Jāru*, *Irakka Jāru*, *Othukkal* and *Orikkai*. Out of the 55 songs belonging to *rāga S̀ankarābharaṇam* and

Table 3.2: Notes in the dataset

Set	*Gamaka* Notes	Plain Notes	Total Notes
Training	3478	8103	11581
Seen	2385	5948	8333
Unseen	3853	5608	9461

Table 3.3: Number of *gamaka*s used in the experiments

Gamaka	Training Set	Seen *rāga* Test Set	Unseen *rāga* Test Set
Podi	315	186	158
Kampitham	195	100	377
Sphuritham	588	596	527
Nokku	802	440	691
Ētra Jāru	951	574	959
Irakka Jāru	423	273	355
Othukkal	525	293	549
Orikkai	475	302	975

its derived *rāga*s (containing only a subset of notes present in *Ṡankarābharaṇam*), we choose 33 songs for training the classifier, while the remaining 22 songs are included in the test set 1. We name this set as 'seen *rāga* test set' since the characteristics of the songs in this set are similar to those used for training the classifier. Twenty three songs belonging to *Kalyāṇi* and its derived *rāga*s are used to form the test set 2. Since the melodic structure of these songs is different from that used for training, we term this set as 'unseen *rāga* test set'. The remaining two songs belong to the *rāga Hamvīru*. Even though this *rāga* is a *derived rāga* of *Kalyāṇi*, it shares some common musical features with *Ṡankarābharaṇam*. Hence, we include these two songs also in the seen *rāga* test set. The number of plain notes and *gamaka* notes for the training and test sets are listed in Table 3.2. The different *gamaka*s used for the experiments, and their number of occurrences are listed in Table 3.3. An excerpt from our dataset is shown in Table 3.4. It contains the note labels, note duration in terms of beats and the information about presence and type of *gamaka*.

Table 3.4: An excerpt from a song belonging to *rāga Hamsadhwani* contained in the training dataset

Note	Duration (beats)	*Gamaka* Presence	*Gamaka* Type	*Rāga*
		⋮		
G2	1	NO	–	*Hamsadhwani*
P2	1	YES	*Ētra Jāru*	*Hamsadhwani*
R2	2	YES	*Irakka Jāru*	*Hamsadhwani*
R2	2	NO	–	*Hamsadhwani*
N2	1	NO	–	*Hamsadhwani*
G2	1	NO	–	*Hamsadhwani*
		⋮		

3.3 Methodology

The input to our system is the notations of a song containing the note labels (first column in Table 3.4), durations (second column in Table 3.4) and the *rāga* of the song (fifth column in Table 3.4). The goal is to predict the presence and type of *gamaka* associated with each note (third and fourth columns in Table 3.4). We extract features based on the notes and employ a Random Forest Classifier (RFC) for these tasks.

3.4 Features

The characteristics of any song are defined by the notes used. Hence, we extract the features based on the notes of the songs for our experiments. Certain *gamaka*s are observed to be more prominent in specific *rāga*s (Viswanathan 1977). This necessitates the inclusion of *rāga* dependent features into the feature set. Karṅāṭic musicians observe that some characteristics phrases are associated with some *rāga*s and certain *gamaka*s occur as part of these note sequences (Viswanathan 1977). From this, we hypothesise that the occurrence of some *gamaka* on a particular note has a dependence on the adjacent notes. Based on all these facts, we consider a set of notes – called neighbourhood notes – adjacent to every note for creating its feature. The extracted features represent the pitch and duration of the notes in the neighbourhood. The

Table 3.5: Note table for feature creation

g1	*G1*	*m1*	*M1*	*P1*	*d1*	*D1*	*n1*	*N1*	*S2*
1	2	3	4	5	6	7	8	9	10
r2	*R2*	*g2*	*G2*	*m2*	*M2*	*P2*	*d2*	*D2*	*n2*
11	12	13	14	15	16	17	18	19	20
N2	*S3*	*r3*	*R3*	*g3*	*G3*	*m3*	*M3*	*P3*	*d3*
21	22	23	24	25	26	27	28	29	30

pitch of a note is represented by the absolute and relative frequency positions of the note. Duration is represented by the number of beats.

The absolute and relative frequency positions of notes are computed based on a note table. This table contains all the notes from the lower, middle, and upper octaves, appearing in the songs contained in the dataset. These notes are arbitrarily labelled using numbers 1,2,3, etc. This numbering scheme is only for representative purpose, and does not indicate the actual frequencies (in Hz) of the notes. The note table used in our experiments is given in Table 3.5.

A neighbourhood of notes adjacent to every note is considered for finding out its features. We name this note (whose features are being found out) as pivotal note. The neighbourhood may be selected either 1) only to the left of the pivotal note, or 2) only to the right of the pivotal note, or 3) on both sides (to the right and the left) of the pivotal note. The corresponding neighbourhoods are called 'left-sided', 'right-sided' and 'both-sided', respectively.

The corresponding note label for the pivotal note is found out from the note table. This represents its absolute frequency position and is included in the feature set with the name *current note*. For analysing the impact of the neighbourhood notes, we also add their relative frequency position in the feature set with respect to the pivotal note. This is obtained by subtracting the *current note* from the absolute frequency position of each of the neighbourhood notes.

As an example, let us consider the note sequence *G2*, *P2*, ***R2***, *R2*, *G2* from the snippet of the song given in Table 3.4. Let us consider *R2* (third note in the sequence) to be the pivotal note. For simplicity, here, we consider a neighbourhood of only two notes on both sides of the pivotal note. From table 3.5, the label for the note *R2* can be found out to be 12, which is used as a feature called *current note*. The label for the note which immediately precedes *R2* in the given set of notes (i.e., *P2*) is 17. The

difference between this and the *Current Note* is 5. This is included in the feature set with the name *Previous Note 1*. *Previous Note 2* is found out by subtracting *Current Note* from the absolute frequency position of *G2*, which appears two notes before the pivotal note. Its value is 2. Similarly, the values for the features *Post Note 1* (the note which immediately follows the pivotal note) and *Post Note 2* (second note after the pivotal note) are 0 and 2, respectively.
In addition to these, another set of features are included for representing the *rāga*-dependent relative frequency position of the neighbourhood notes. Another note table is constructed based on the *rāga* of the song for computing these features. This *rāga*-dependent note table contains only those notes which are present in the *rāga* of the song. Relative frequency positions of the neighbourhood notes are found out based on this table. For example, the *rāga*-dependent note table for the *rāga Hamsadhwani* is shown in Table 3.6. It contains only five notes per octave. Assuming the same sequence of notes and neighbourhood used in the previous example, the *rāga*-dependent features *Previous Note Rāga 1*, *Previous Note Rāga 2*, *Post Note Rāga 1* and *Post Note Rāga 2* for the note *R2* can be found out to be 2, 1, 0 and 1, respectively.

The duration of the notes in the neighbourhood, as well as the pivotal note, are also included in the feature set. These are represented in terms of the number of beats. Assuming the same sequence in the previous examples, it can be seen from Table 3.4 that the duration of the pivotal note is two. Similarly, the durations of the preceding two notes succeeding two are 1, 1, 2 and 1, respectively. Thus, if a both-sided neighbourhood of five notes is considered, there are 11 features for representing the durations, 11 for note frequency positions and 10 for *rāga*-dependent relative frequency positions. Thus, a total of 32 features are extracted for every note.

Table 3.6: *Rāga*-dependent note table for *rāga Hamsadhwani*

G1	*P1*	*N1*	*S2*	*R2*	*G2*	*P2*	*N2*	*S3*	*R3*	*G3*	*P3*
1	2	3	4	5	6	7	8	9	10	11	12

3.5 Random Forest Classifier

The model–based classifiers such as Gaussian Mixture Model (GMM) as well as the Artificial Neural Networks (ANN) assume the feature vectors to be real–valued (Larose

2015). Since our features are nominal and not real–valued, we use a Random Forest Classifier for the detection and classification of *gamaka*s. RFC is an ensemble of Decision Trees (Breiman 2001). Decision Trees try to classify the data based on a sequence of binary decisions. The process starts with all the data samples at the root node. The data is split into two different groups at each node depending on a yes/no question based on a feature. The best feature enabling maximum separation between the two classes is chosen by the algorithm. The splitting process is continued to subsequent nodes until the entire data is classified (Duda *et al.* 2012). Parameters of RFC include the number of trees, maximum number of leaf nodes, depth of the trees, minimum number of samples needed to split a node, etc. We try to optimise some of these parameters on the basis of cross validation. The optimal features are then used on the test sets for finding the performance of the classifier.

3.6 Experiments

We try to predict the presence and type of *gamaka*s from the skeletal notations of a song. This is performed in two phases. In the first phase, we try to check whether there is any *gamaka* associated with a note. This phase is called *gamaka detection*. In the next phase, the type of *gamaka* associated with a note is predicted. This phase is called *gamaka classification*. For *gamaka* classification task, we consider only those notes having any *gamaka* associated with them. The notes which do not contain *gamaka*s are manually removed from the data before the classification task. Initially, the experiments are conducted using the default parameters for the classifier. Subsequently, the parameters and the type and size of the neighbourhood are optimised based on cross–validation results. There are 33 songs in the training set. A 33-fold cross–validation is performed, each time leaving out a song completely and training on the other 32 songs. The left-out song is used as the validation set and the prediction accuracy is found out. This process is repeated for the entire 33 songs in a round-robin manner. The cross–validation accuracy is found out by averaging the individual accuracy values obtained for each of these 33 songs.

3.6.1 Gamaka Detection

For detecting the *gamaka*s, we use the features based on different types of neighbourhoods. The neighbourhoods considered here are of three types – 'left–sided', 'right–sided' and 'both–sided'. A left–sided neighbourhood of 'N' notes contains the pivotal note along with N notes to the left of it (a total of N+1 notes). Similarly, a right–sided neighbourhood of N notes contains the pivotal note and N notes to the right of it. A both–sided neighbourhood of N notes contains the pivotal note, N notes to the left of it and N notes to the right of it (a total of 2N+1 notes). We consider different values for N, ranging from zero to ten. A neighbourhood of size zero contains only the pivotal note. We try to optimise the classifier parameters such as the number of trees and the minimum number of samples required to split a node. A grid search of these parameters is performed jointly, along with the size and type of neighbourhood. Fig. 3.1 shows the variation of cross validation accuracy with respect to different neighbourhoods. From the results, it is concluded that a both–sided neighbourhood of size 3 (a total of 7 notes) yields the best cross validation accuracy.

Fig. 3.2 shows the variation of cross–validation accuracy with respect to the number of trees. From the plot, the optimal value for the number of trees can be found to be 1536. Minimum samples needed to split a node is varied from one to ten, and the optimal value for this parameter is found to be four. The cross–validation ac-

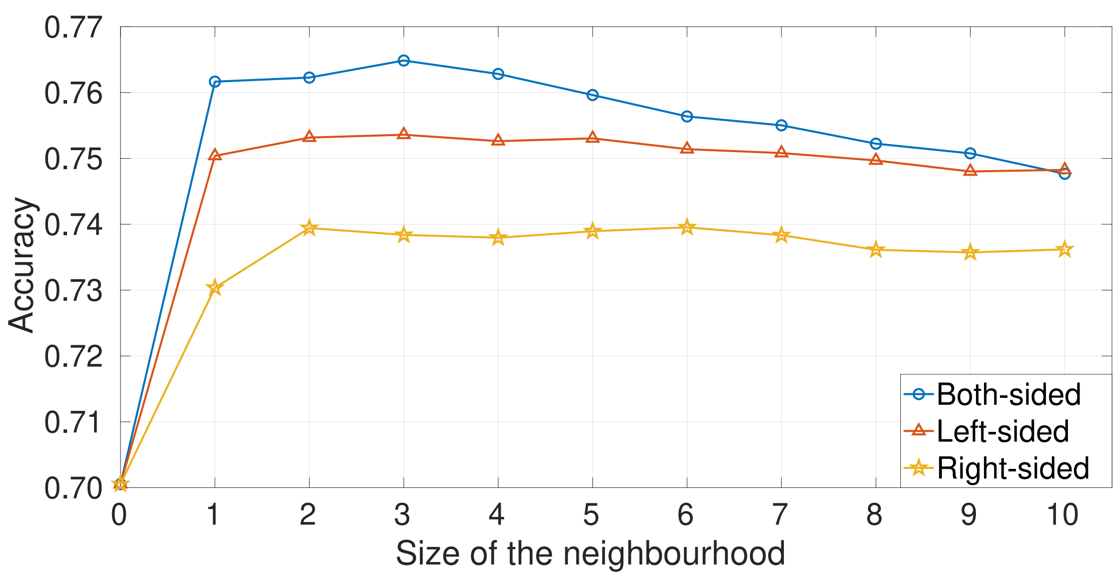


Figure 3.1: Accuracy values corresponding to different neighbourhoods.

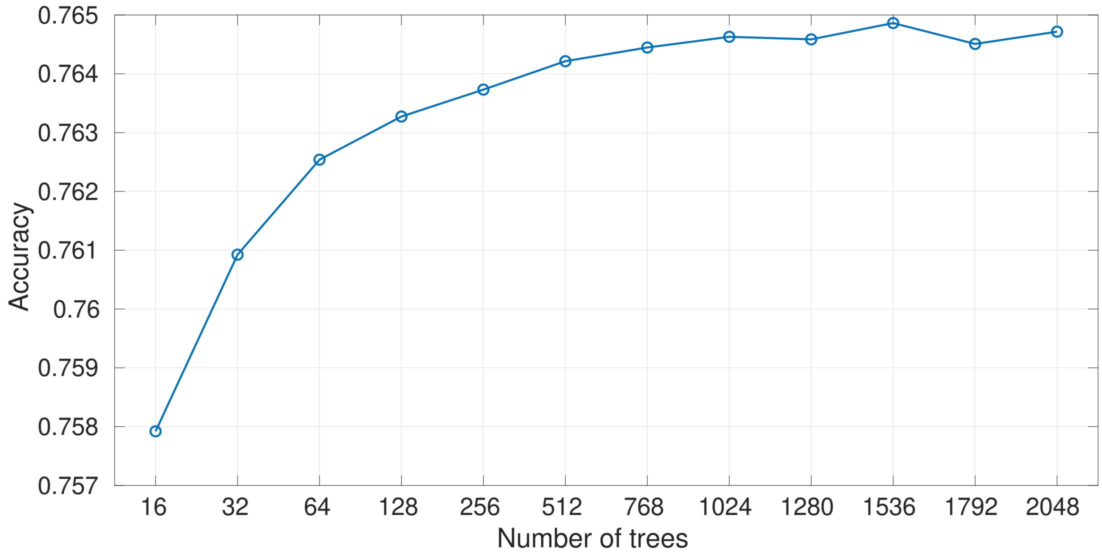


Figure 3.2: Variation of accuracy with respect to number of trees (both–sided neighbourhood of size 3).

Table 3.7: Cross validation accuracies (for the both-sided neighbourhood of 3 notes).

Default Parameters	Tuned Parameters
75.1%	76.5%

curacies are listed in Table 3.7. Tuning of hyper parameters seems to have minimal impact on the classifier's performance. The cross–validation accuracy is improved by approximately 1.4%.

We try to evaluate the performance of these features and tuned parameters on the test sets. Experimental results are shown in Table 3.8. We compare the accuracy of our tests with random classification accuracy. Random classification is performed by choosing the class which occurs the most in the dataset. In our training data, non—*gamaka* occurs the most. Hence, deciding any sample as non-*gamaka* yields an accuracy of 71.3%. From the results, it can be seen that the accuracy values obtained by the use of RFC are better than the random classification accuracy. In the seen *rāga* test set, the accuracy is improved by approximately 5.2% over the random classification accuracy. In the case of unseen *rāga* test set, the improvement is around 8.3%. Since the non-*gamaka* class is more probable than the *gamaka* class in our training data, accuracy cannot be considered as a meaningful measure. For example, the random classifier described above gives an accuracy of 71.3% even though it does

Table 3.8: Test accuracies for *gamaka* detection

Test Set	Random Classification		RFC	
	Accuracy	F score	Accuracy	F score
Seen *Rāga*s	71.3%	0	76.5%	0.45
Unseen *Rāga*s	59.9%	0	68.2%	0.42

not detect even a single *gamaka* from the seen *rāga* test set. Hence, to assess the performance of the classifier, we also compute the F score for both the seen and unseen *rāga* test sets. F score is a measure of the classifier's performance in selecting the positive class. It is computed as the harmonic mean of precision and recall (Powers 2011). Since the random classifier does not detect any sample as the positive class (*gamaka* in this case), its F score is zero. RFC outperforms the random classifier by a huge margin in terms of F score also. F score of RFC is found to be 0.45 and 0.42 for the seen and unseen test sets, respectively. The melodic structure of the songs in Seen *rāga* test set and training set is the same, and hence they have similar characteristics. This accounts for the increased accuracy and F score for the seen *rāga* test set.

3.6.2 Analysis of the Results

Our experiments show that the *gamaka*s depend on the neighbourhood notes. An elementary analysis of the training data is conducted to study this dependence. The number of distinct note sequences that are common to the *gamaka* and non-*gamaka* classes is computed for a fixed neighbourhood size. We compare these with the number of distinct common sequences if the labelling were to be random. If there is no *gamaka* information contained in the neighbourhood notes, the number of distinct common sequences obtained from the actual and random labelling will be similar.

We consider both–sided neighbourhood of size ranging from zero to ten for finding out the distinct common sequences in the training data. Random labelling is performed in such a way that the probabilities of *gamaka* and non-*gamaka* notes are the same as those in the actual labels (only 30% of notes in the training data contain *gamaka*s). Mean and standard deviation of the number of common sequences for all neighbourhood sizes are computed for ten random labellings. Fig. 3.3 shows the number of sequences for different neighbourhood sizes for the training data using

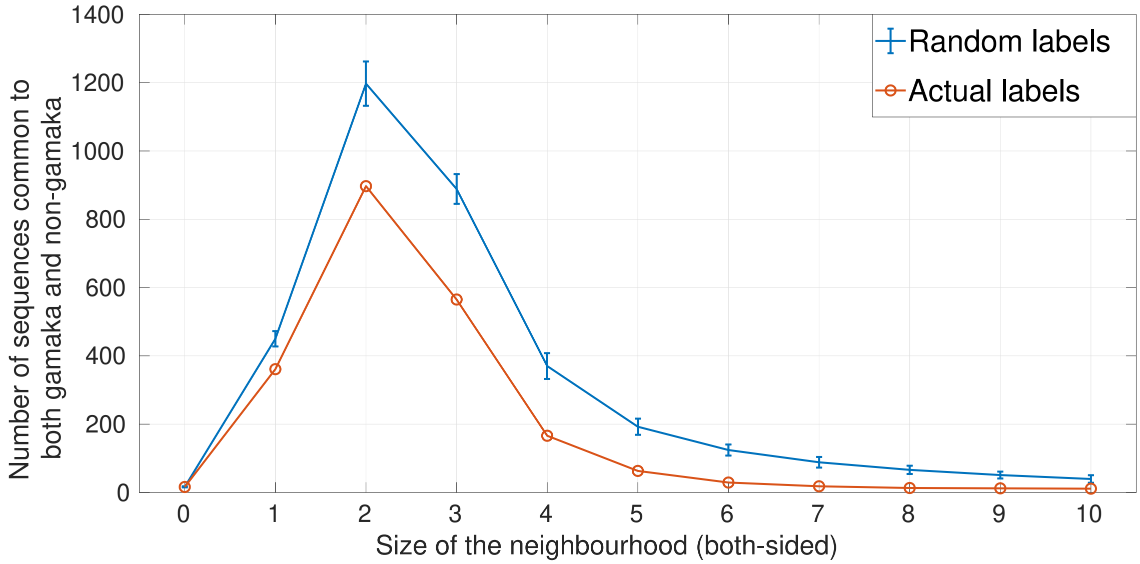


Figure 3.3: Number of sequences common to both *gamaka* and non-*gamaka* notes for different neighbourhood sizes. The plot for the random labelling shows mean and the error bar shows $\pm 3\sigma$ for 10 random labelling.

actual labels and random labels for *gamaka*. In the plot, the number of common sequences for random labelling is represented using the mean. The error bars represent three times the standard deviation ($\pm 3\sigma$). It can be seen that the number of common sequences is always less for the actual labelling as compared to the random labelling. This shows that there is more distinction between the sequences belonging to the *gamaka* and non-*gamaka* classes in the actual labelling as compared to the random labelling. Hence, there is a certain amount of information contained in the neighbourhood notes.

For example, from Fig. 3.3, it can be seen that the average number of common sequences found for ten random labellings is 1197. But, the number of common sequences using the actual labels is only 897. There is a reduction of around 25% in the number of sequences common to both the *gamaka* and non-*gamaka* classes compared to the randomly labelled data. This reduction is significant since it is larger than the 3σ (± 64) range. This trend is always followed in all the neighbourhood sizes considered in our experiments.

The number of overlapping sequences for the actual labelling as well as the random labelling decreases towards both the ends of the plot shown in Fig. 3.3. The reduced number of overlapping sequences on either side of the plots can be explained

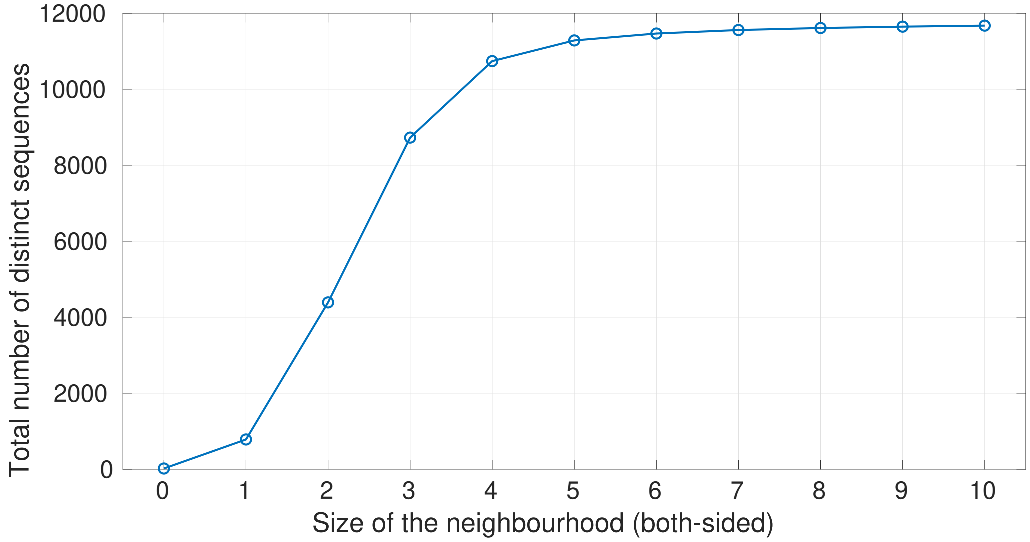


Figure 3.4: Total number of distinct sequences (for both *gamaka* and non-*gamaka* notes) for different neighbourhood sizes

by two different phenomena. For smaller neighbourhood sizes, the number of distinct sequences is less. Hence, naturally, the number of overlapping sequences are also less. Fig. 3.4 shows the total number of distinct sequences for both *gamaka* and non-*gamaka* notes in the training data for different neighbourhood sizes. It is seen that the number of distinct sequences saturates as the neighbourhood size increases. Even though the number of combinations increases with the length of the sequence, all these combinations are not musically viable. This is the reason for the saturation of the curve for larger neighbourhood sizes. As the total number of sequences for *gamaka* and non-*gamaka* remains fixed, and the length of the sequences increasing, the number of matching sequences reduces. This is the reason for the reduction in the number of common sequences towards the right end of the curve shown in Fig. 3.3.

The optimal neighbourhood obtained after cross–validation corresponds to the mid–region of the curve shown in Fig. 3.3. The reduction in the number of common sequences is prominent in this neighbourhood. The analysis performed here takes only the note frequencies into consideration. Information contained in the *rāga* and the durations of the note are omitted from this analysis. The inclusion of this information may provide a better distinction between *gamaka* and non-*gamaka* classes.

3.6.3 Gamaka Classification

In this phase, we try to predict the type of *gamaka* from the notes. We consider only eight different types of *gamaka*s for this task. Features are the same as those used in the previous set of experiments. Training and test sets also contain the same songs as in the detection experiment. But, we manually remove all the plain notes from these songs and use only the *gamaka* notes for the classification experiment.

We consider the same types and sizes of the neighbourhoods as in the detection experiment and perform the neighbourhood optimisation. Left, right and both–sided neighbourhoods are considered, while the number of neighbourhood notes are varied from zero to ten. We also perform optimisation of the hyper–parameters as in the previous experiment. We perform a round–robin cross–validation as in the detection experiment to find the optimum neighbourhood, and the results are illustrated in Fig. 3.5. It can be seen that the best accuracy is obtained when a both–sided neighbourhood of size one is considered (total the notes). Optimum values for the minimum number of samples needed for RFC to split a node is found to be two, and the optimum value for the number of trees is found to be 128. Variation of accuracy with respect to the number of trees is shown in Fig. 3.6.

The cross–validation accuracies under different experimental conditions are listed in Table 3.9. Only a very small change (∼1.3%) in the accuracy values are observed after parameter optimisation. We use the optimum parameter values to perform experiments on the test sets. Test results are tabulated in Table 3.10. Here also we try to compare the results with random classification. In the seen *rāga* test set, the most repeated type of *gamaka* is *Sphuritham*. Classifying every *gamaka* as *Sphuritham* yields an accuracy of 25.5%. In the unseen *rāga* test set, the most repeated type of *gamaka* happens to be *Orikkai*, and the blind assignment of *Orikkai* to all samples yields an accuracy of 25.7%. From the results, it is clear that the RFC with tuned parameters outperforms the random classifier by ∼45% in the *seen rāga* test set, and by ∼33% in the *unseen rāga* test set.

The F scores for distinguishing a specific *gamaka* from the other seven classes in the test sets are displayed in Fig. 3.7. From the figure, it can be seen that the F score for the *gamaka Kampitham* is low in both the seen and unseen *rāga* test sets. This can be attributed to the lesser number of samples for this *gamaka* (less than 200 samples) compared to the other *gamaka*s in the data set. In this experiment also, seen *rāga* test set has the increased accuracy and F score compared to the unseen *rāga* test

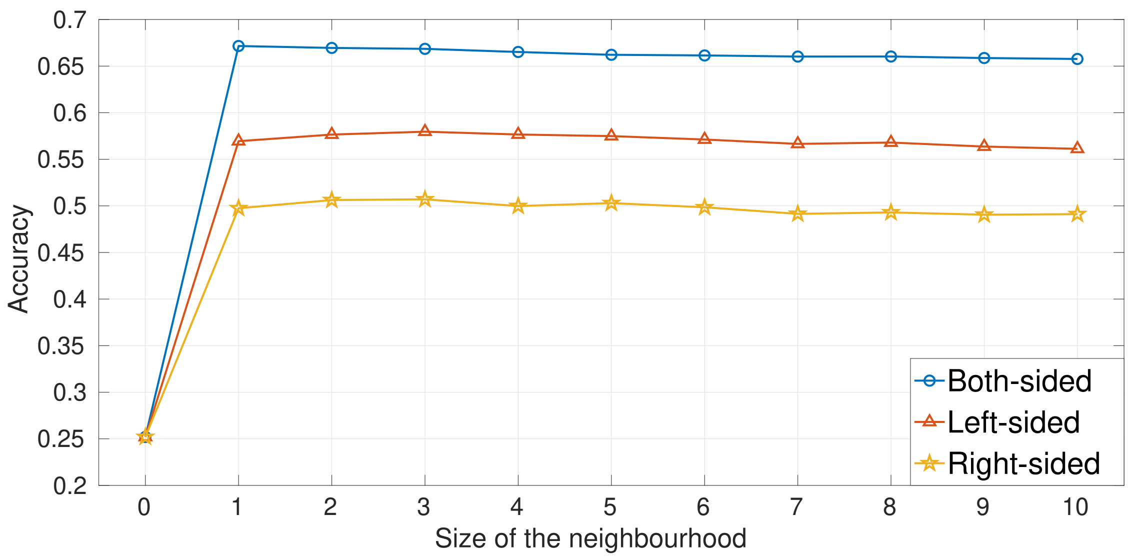


Figure 3.5: Accuracy values corresponding to different types and sizes of neighbourhoods (*gamaka* classification).

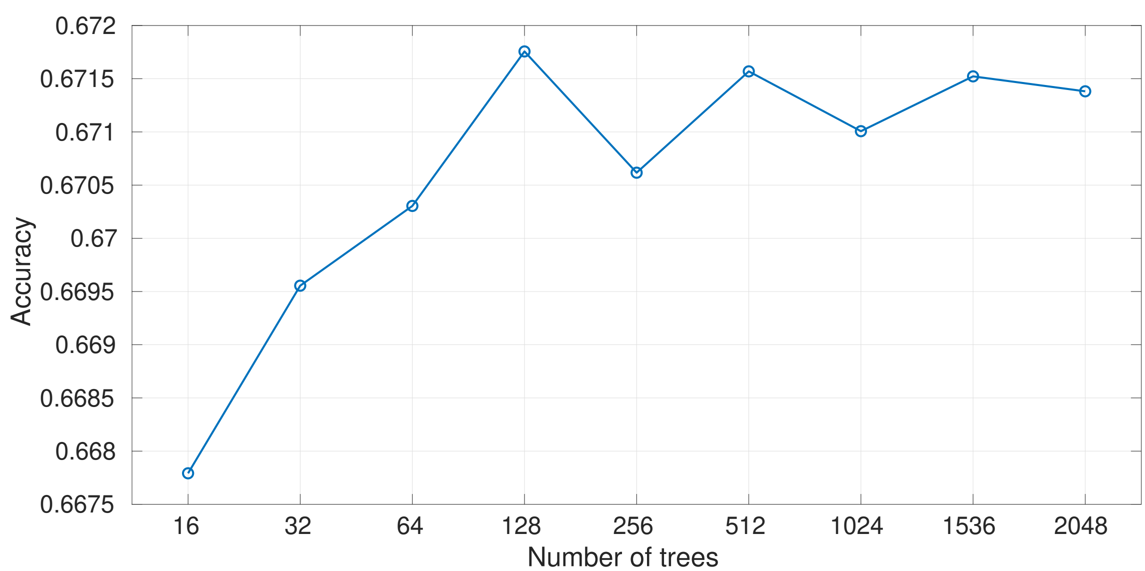


Figure 3.6: Variation of classification accuracy with respect to number of trees (for the both–sided neighbourhood of size 1).

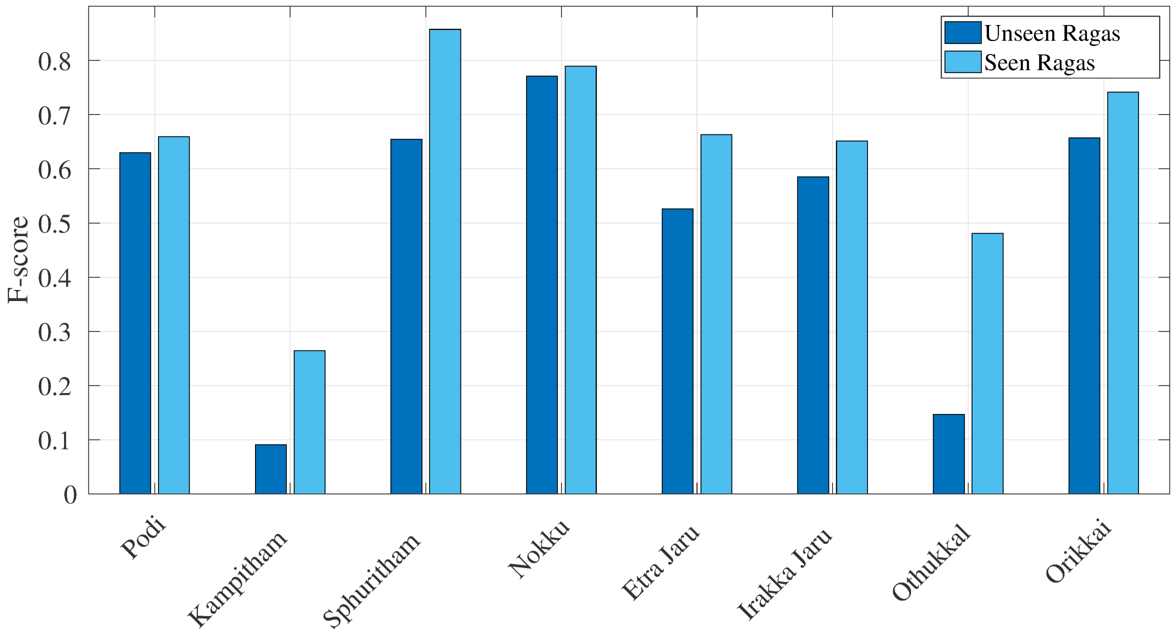


Figure 3.7: F score of different *gamaka*s for the seen and unseen *rāga* test sets.

Table 3.9: Cross validation accuracies for gamaka classification(for the both-sided neighbourhood of size 1)

Default Parameters	Tuned Parameters
65.9%	67.2%

set. Again, it can be due to the similarity in the melodic structure between the songs in the training set and the seen *rāga* test set.

The confusion matrices obtained for the seen and unseen *rāga* test sets are shown in Table 3.11. Both the matrices have similar kinds of confusion between the *gamaka*s. It can be seen that the *gamaka* named *Kampitham* is confused with several other *gamaka*s. This has the least number of correctly classified instances in both the seen and the unseen test sets. Prominent confusion is observed in the case of *Podi*, which is often misclassified as *Othukkal*. Another notable confusion is seen between *Othukkal*

Table 3.10: Test accuracies for *gamaka* classification

Test Set	Random Classification Accuracy	Accuracy
Seen *Rāgas*	25.5%	70%
Unseen *Rāgas*	25.7%	59%

Table 3.11: Confusion matrix for seen *rāga* test set. Correctly classified instances are listed along the diagonal. Shaded cells represent prominent confusions

(a) Seen *rāga*s

Predicted Class \ Actual Class	**Podi**	**Kampitham**	**Sphuritham**	**Nokku**	**Ētra Jāru**	**Irakka Jāru**	**Othukkal**	**Orikkai**
Podi	92	0	0	0	0	1	0	0
Kampitham	0	16	2	1	1	2	0	0
Sphuritham	0	14	518	20	4	11	9	37
Nokku	11	11	25	362	39	16	9	4
Ētra Jāru	0	18	6	22	361	5	89	18
Irakka Jāru	0	37	27	24	5	185	3	14
Othukkal	83	2	9	7	160	0	177	4
Orikkai	0	2	9	4	4	53	6	225

Actual Class

and *Ētra Jāru*. Similarly, *Irakka Jāru* class is wrongly classified as *Orikkai*. For the unseen *rāga* test set, two more prominent confusions are observed, which are not present in the seen *rāga* case. One is between *Irakka Jāru* and *Orikkai* and the other one is between *Orikkai* and *Sphuritham*. In both the test sets, *Nokku*, *Orikkai* and *Sphuritham* are the most correctly classified *gamaka*s.

3.6.4 Analysis on the Training Data

An elementary analysis on training data is performed to get an insight into the performance of *gamaka* classification system. For this analysis, we focus on the characteristic phrases in the training set containing *gamaka*s. Towards this, we extract the most frequently appearing sequences where a *gamaka* is present in the pivotal note. We try to analyse the existence of any characteristic note patterns specific to certain *gamaka*s from these most frequently occurring sequences. We consider two different neighbourhoods for this analysis. The first one is both–sided neighbourhood of size one (a total of three notes including the pivotal note), and the second one is the both–sided neighbourhood of size two (a total of five notes including the pivotal note). We extract all sequences of length three and five with a *gamaka* on the pivotal note. From these,

Table 3.12: Confusion matrix for unseen *rāga* test set. Correctly classified instances are listed along the diagonal. Shaded cells represent prominent confusions

(a) Unseen *rāga*s

Predicted Class \ Actual Class	**Podi**	**Kampitham**	**Sphuritham**	**Nokku**	**Ētra Jāru**	**Irakka Jāru**	**Othukkal**	**Orikkai**
Podi	74	0	0	0	1	0	2	0
Kampitham	0	19	4	0	11	4	0	1
Sphuritham	3	108	421	33	9	6	21	159
Nokku	21	42	34	576	49	41	10	27
Ētra Jāru	0	88	9	21	497	6	145	150
Irakka Jāru	1	93	36	8	6	222	2	34
Othukkal	59	2	8	49	383	7	361	67
Orikkai	0	25	14	3	2	69	8	536

Actual Class

five most repeated sequences are determined for each type of *gamaka* for each of the neighbourhoods.

We find out the frequencies of notes for all these sequences and normalise them such that the pivotal note has unit frequency. This is performed to find out the change in relative frequencies of notes on either side with respect to the frequency of the pivotal note. The plot of these normalised note frequencies for each of the *gamaka*s is shown in Fig. 3.8 and Fig. 3.9. Thicker lines represent the more frequent sequences containing each *gamaka*. Legend in the plot shows the number of times each sequence is repeated. From plots, it can be seen that the unique characteristic patterns for five–note sequences are less than those for the three–note case. This can be a reason for increased cross validation accuracy in the case of both–sided neighbourhood of size one.

From the plots in Fig. 3.8, it can be seen that the most frequent patterns of *Ētra Jāru* and *Othukkal* are very similar to each other. This can be a reason for the increased confusion between these *gamaka*s. Similarly, *Irakka Jāru* and *Orikkai* share common patterns in the most repeating sequences, which may be the reason for the prominent confusion between these two. *Nokku* has characteristic patterns which are unique, and shares very few note patterns with other *gamaka*s. This can be a reason for reduced confusion and increased F-score of this *gamaka*. Another interesting observation is that *Kampitham* shares note patterns with all other *gamaka*s, thereby lacking a unique shape for the note pattern. This can be a reason for it being misclassified as several other *gamaka*s most of the times. However, this analysis takes only the relative frequency of the note patterns into consideration. Other features related to the note duration and *rāga*s are omitted from this analysis.

3.7 Conclusion

In this chapter, we try to predict the presence and type of *gamaka*s associated with a note from the skeletal notations of a song. Towards this, we propose features based on the frequency and duration of notes present in the song as well as the *rāga* of the song. These features are extracted for a neighbourhood of notes around the pivotal note. A Random Forest Classifier is used for the detection and classification of *gamaka*s.

Optimisation of the type and size of the neighbourhood along with the parameters of the classifier is performed based on the cross–validation results. Using the optimised values for the parameters, accuracy of around 76.5% is observed for the seen *rāga* test set. For the unseen *rāga* test set, the accuracy obtained is 68.2%. Best accuracy is yielded by a both-sided neighbourhood of size three (a total of seven notes). Evidence from the training data also suggests that there is information contained in this neighbourhood which helps distinguish between *gamaka* and non-*gamaka* notes.

We follow the same approach for *gamaka* classification experiment. For seen *rāga* test set, we observe classification accuracy of 70%. For unseen *rāga* test set, the accuracy is found to be 59%. the optimal neighbourhood is found to be both–sided with size one (three notes in total). We observe that the *gamaka*s named *Sphuritham*, *Nokku*, *Orikkai* and *Irakka Jāru* exhibit characteristic patterns in the note sequences, and are having a better F score. The accuracy obtained for the unseen *rāga* test set is less compared to the seen *rāga* test set in both the detection and the classification experiments. Even though we consider around 20000 notes covering 80 songs belonging to 22 *rāga*s, it is still very less compared to the wide range of songs belonging to a very large number of *rāga*s (around 200 *rāga*s are listed in SSP (Dikshitar 2008) in KM. We expect that a larger, diverse dataset containing more *rāga*s will help the system detect the presence and type of *gamaka*s more accurately.

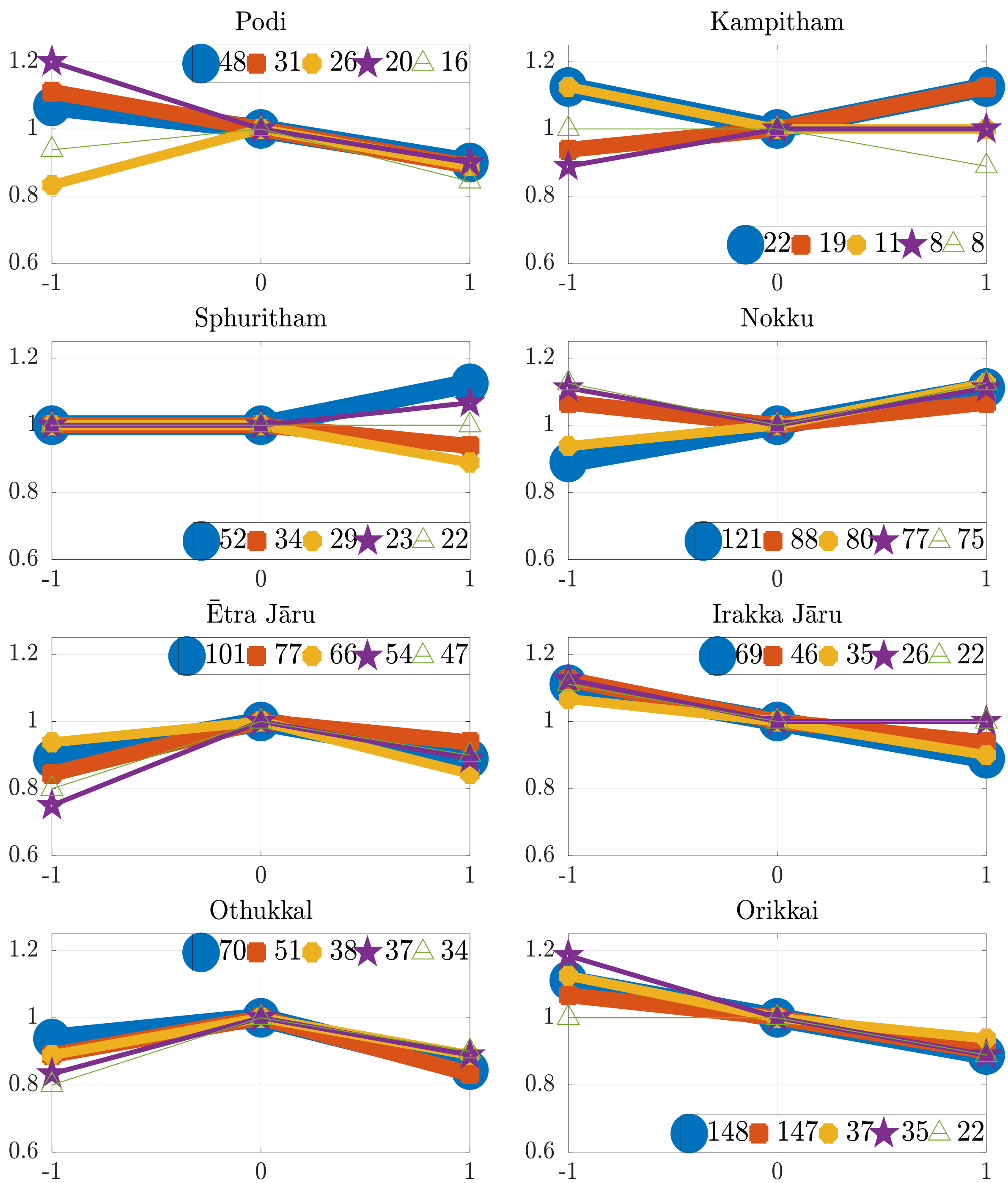


Figure 3.8: Note frequency plots (normalised to the frequency of the pivotal note) of 5 most repeated sequences for each *gamaka* (Considering the both-sided neighbourhood of size 1). The legend shows the number of occurrences for each sequence.

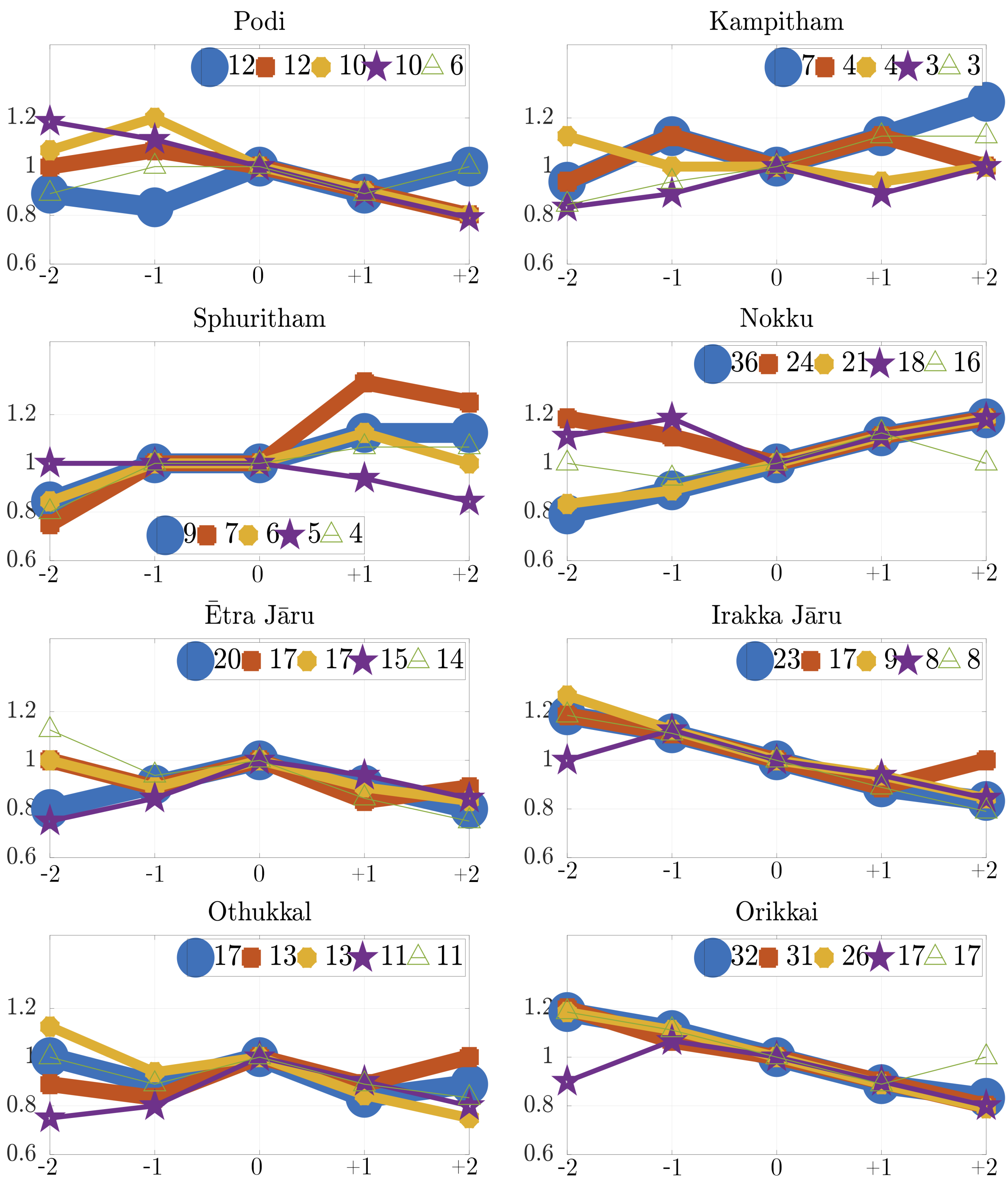


Figure 3.9: Note frequency plots (normalised to the frequency of the pivotal note) of 5 most repeated sequences for each *gamaka* (Considering the both-sided neighbourhood of size 2). The legend shows the number of occurrences for each sequence.

Chapter 4

SYNTHESIS OF KARṆĀṬIC FLUTE MUSIC

Karṅāṭic Music (KM) is different from western music in many ways. One of the major differences lies in the use of *gamaka*, which can be thought of as a bend or inflexion in the pitch contour of a note. These can occur in the transition region between two notes or during a single note itself. Due to the extensive use of *gamaka*s, the pitch contour of a note in Karṇāṭic flute music may fluctuate most of the times. Such a continuous pitch contour is approximated using discrete segments in the frame–based synthesis methods. Even though the perceived difference can be made very low by the use of smaller windows, this method still deviates from the fundamental concept of continuous pitch contour. In this chapter, we propose a continuous time spectral model to synthesise bamboo flute music for Karṅāṭic songs without using a frame–based approach. Towards this, we model all the important spectral parameters of bamboo flute tone to synthesise plain notes, *gamaka*s and non-*gamaka* transitions.

4.1 Karṅāṭic Music on South Indian bamboo Flute

A bamboo flute tone is rich in harmonic content. Most of the energy present in a bamboo flute tone is contributed by the fundamental frequency and its harmonics. Fig. 4.1 shows the spectrogram of a single note played in South Indian bamboo flute. The dominance of the harmonic components is evident from the spectrogram. There is also a noise–like energy present in the spectrogram. Thus, a bamboo flute tone can be decomposed into several harmonically related sinusoids plus coloured noise.

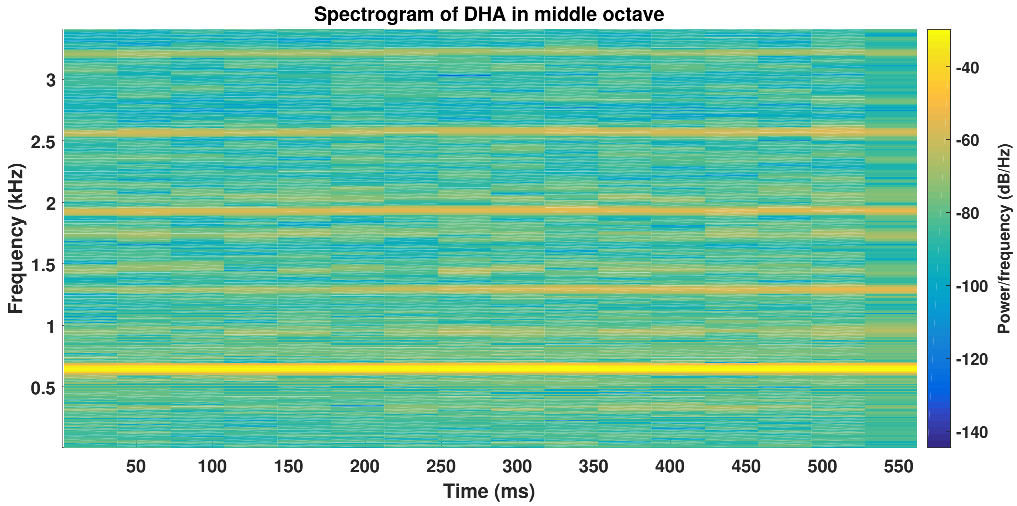
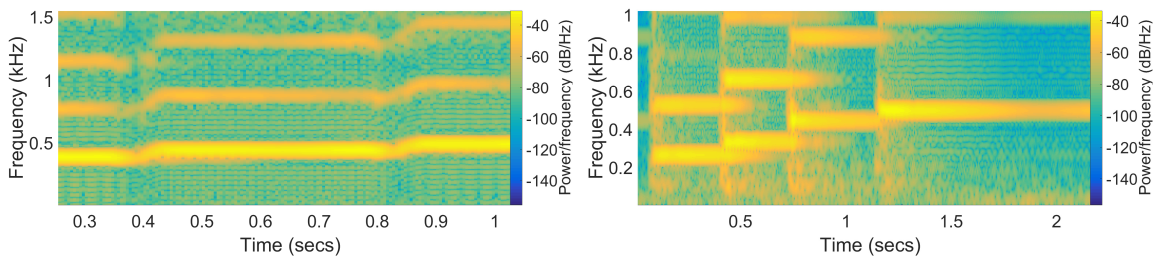


Figure 4.1: Spectrogram of a single note played on south Indian bamboo flute (window size=70 ms, overlap=35ms, FFT size=4096). The image is magnified to show the individual harmonics in the spectrogram.

4.1.1 Pitch Contour

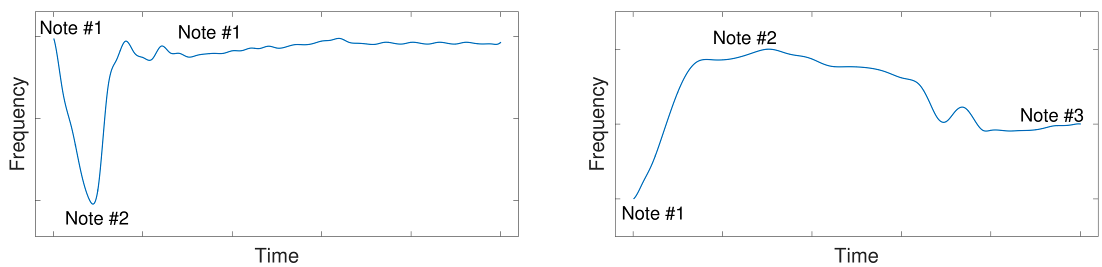
When multiple notes are played on the bamboo flute, the resulting tones will have a continuous pitch contour if the notes are produced in a single blow. But, in the case of a piano, pitch contours take only a discrete set of values. In the transition region between two notes, pitch values of both the notes will be present. To demonstrate this, we compare the spectrograms of the note sequences produced using a bamboo flute and a piano. Figure 4.2 shows the spectrograms for KM played on a south Indian bamboo flute, and western musical notes played on a piano. From the spectrogram of the piano sample, it can be clearly seen that the frequency contours of individual notes overlap in the transition region. The sinusoidal components corresponding to the previous note start to fade out slowly after the beginning of the next note. In the case of the flute spectrogram, the note transitions are continuous. This smooth transition is evident in the case of other harmonics too.

Fig. 4.3 shows the pitch contours of two different *gamaka*s. Regions corresponding to different notes are labelled. Each of these pitch contours corresponds to only a single note, even though the pitch transitions clearly show the presence of more than one note in each of them. Traditionally, these additional but essential notes are omitted while writing the notation of the song. For example, in Fig. 4.3a, pitch contour shape of the *gamaka* called *Sphuritham* is shown. Pitch contour starts from one note, goes



(a) Flute **(b)** Piano

Figure 4.2: Spectrograms for Karṅāṭic Music played on bamboo flute and western music played on piano



(a) *Sphuritham* **(b)** *vali*

Figure 4.3: Instantaneous pitch contours for two different *gamaka*s played on flute

down to reach the lower note, and then jumps up to reach the same note and settles there. The entire pitch bend is traditionally assumed to be one single note. Another *gamaka* named *Vali* is shown in 4.3b. In this, while going from one note to the other ('Note #1' to 'Note #3'), pitch contour touches another note ('Note #2'). This intermediate note is omitted from the musical notation.

4.1.2 Spectral Weights of Harmonics

The spectral weights for different harmonics are not the same for all notes played on the flute. The relative weights of harmonics with respect to the first harmonic also differ from note to note. For example, Fig. 4.4 shows the variation of relative weights for the second and third harmonics for five different notes in a Karṅāṭic *rāga Mōhanam*. When moving from one note to another, not only the pitch and harmonic frequencies change, but also their respective weights. For a signal consisting of continuous frequency changes, the weights for the note transition regions and the *gamaka* regions need to be interpolated for a perfect representation.

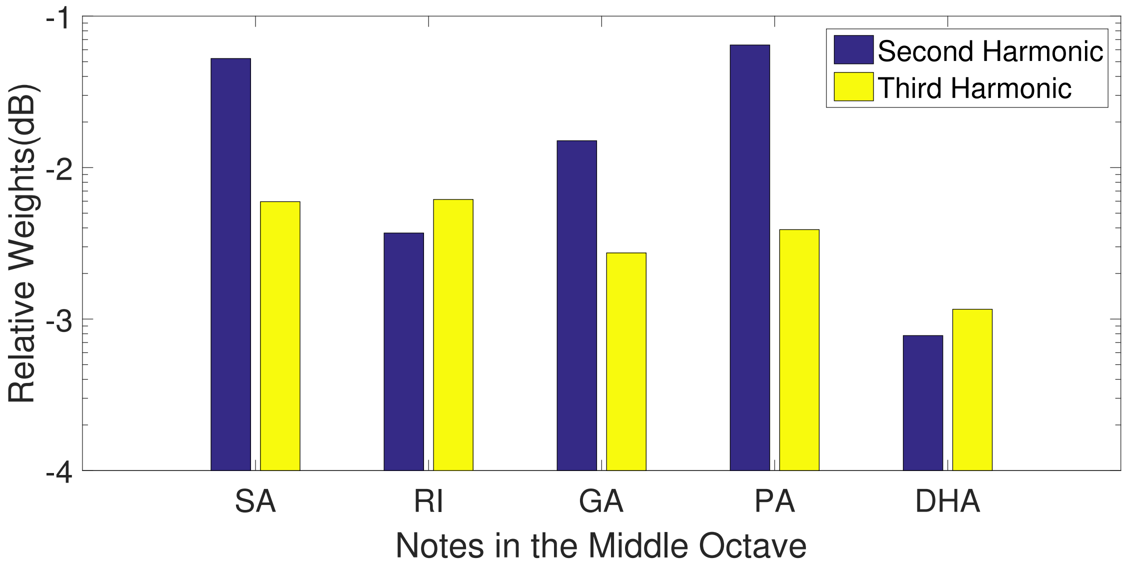


Figure 4.4: Variation of relative weights for the first two harmonics over five different notes. Weight of first harmonic is normalized at 0dB.

4.1.3 Time Domain Envelope

Another important characteristic feature of a flute tone is its time domain amplitude envelope. This can be split into three different regions; namely, attack, sustain and decay. How the waveform evolves into its actual shape is different for different pitch contours. Fig. 4.5 shows the amplitude envelopes for different types of flute tones. It can be observed that the amplitude envelope of a single plain note changes with the presence/absence of *gamaka* in the note.

4.1.4 Wind Noise

The wind noise is another important component present in flute sound. In a flute, while the harmonic part of the signal is generated as a result of the sustained oscillations produced inside the bore, the wind noise is produced by the turbulent streaming of the air when it passes through a narrow opening (Serra *et al.* 1997). In addition to the harmonic components, a noise–like energy can also be seen in the spectrogram shown in Fig. 4.1. This noise–like energy also needs to be modelled for adding naturalness to the flute tone. For analysis, wind noise for a note is recorded by blowing into the flute without creating resonance while maintaining the same finger positions for generating the actual tone for the note. Fig. 4.6 shows the spectrograms of the wind

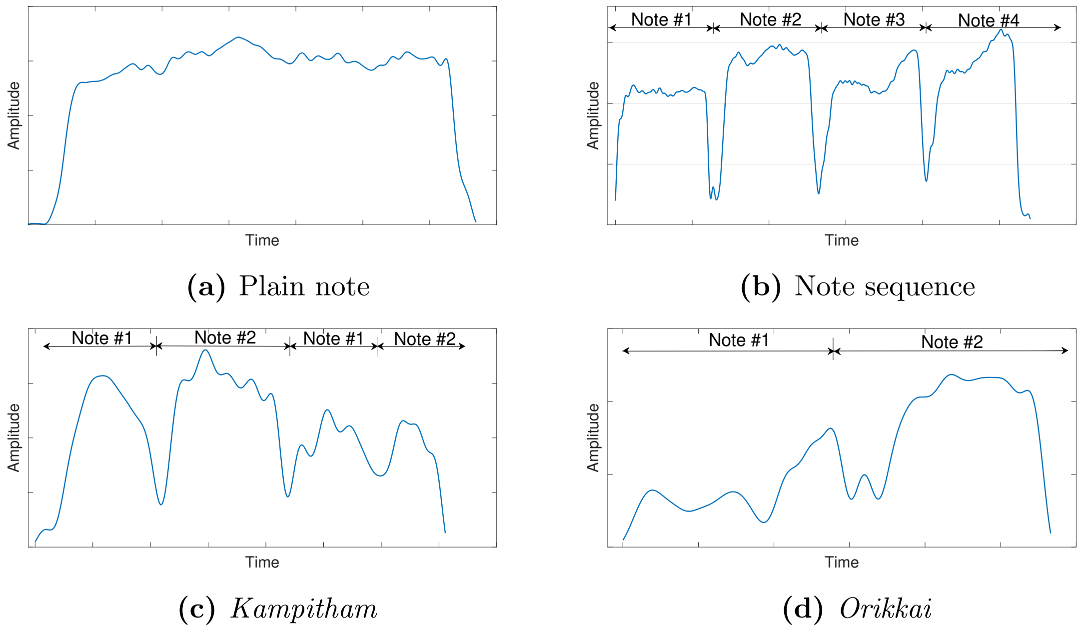
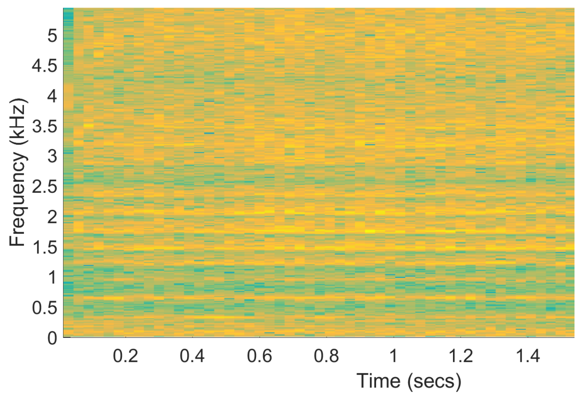


Figure 4.5: Amplitude envelopes for four different note sequences

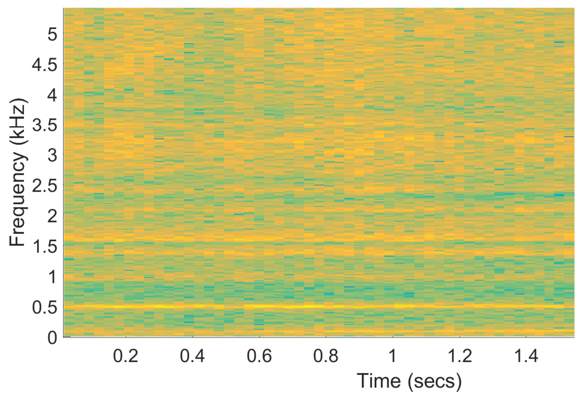
noise recorded for the notes *dha* and *ga* for middle octave. From the spectrograms, it can be seen that there are dominant peaks present in the noise spectra which are located very close to the fundamental frequency of the actual note. A similar trend is observed in other notes too. Thus, it can be seen that the noise is different for different notes. This demands the noise to be modelled differently for each note for a perfect reconstruction of the note tone. At the same time, the spectrograms for the noise signals from different octaves are almost the same, where the spectral peaks appear almost at the same locations. From this, it can be deduced that the noise signal for each note is different, but they are independent of the octave positions (for Chinese flutes, these properties of the breath noise have been reported in the literature (Ayers 2005)). Hence, we feel that modelling the noise for every note from any one octave will be sufficient in representing the wind noise.

4.2 Flute Music Synthesis

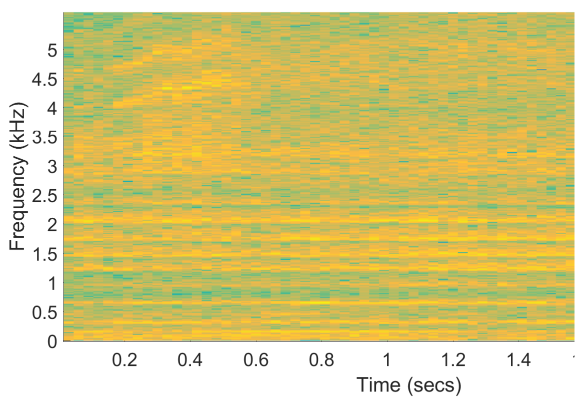
Our goal is to generate flute music from the song notations. The inputs to our system are note labels, durations and *gamaka*s associated with each note present in the song. We model the pitch contour, harmonic amplitudes, time domain envelope and the



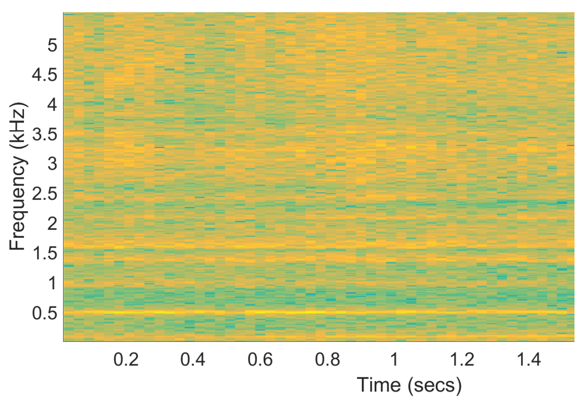
(a) *dha* noise (middle octave)



(b) *ga* noise (middle octave)



(c) *dha* noise (lower octave)



(d) *ga* noise (higher octave)

Figure 4.6: Spectrograms of the wind noise for two notes in the same octave.

wind noise for each of the notes for this task. Synthesis is based on the modified Spectral Model Synthesis (SMS) (Serra *et al.* 1997).

4.3 Baseline: Spectral Model Synthesis (SMS)

This is a frame by frame analysis-by-synthesis method for modelling the sound produced by any physical system. The spectrum of the sound is approximated as the sum of sinusoids plus filtered white noise. In the analysis phase, from the spectrum of the original signal, the parameters such as the number of sinusoids and the time–varying phase and spectral weight of each of the sinusoid are estimated for every frame. The weighted sum of these sinusoids is subtracted from the original signal to obtain the residual signal. By spectral fitting of this residual, the impulse response of the noise filter is obtained. White noise is filtered using this filter to obtain the noise part of the signal. Adding the sinusoidal part and noise part together gives the final synthesized signal for the corresponding frame. For every frame, the synthesized signal is expressed as

$$\widehat{s}_i(t) = \sum_{k=0}^{\widehat{K}_i} \widehat{A}_{k,i}(t) \cdot \cos(2\pi \widehat{f}_{k,i} t) + \widehat{n}_i(t), \tag{4.1}$$

where $\widehat{K}_i$ is the estimated number of sinusoids, $\widehat{A}_{k,i}(t)$ and $\widehat{f}_{k,i}$ are the amplitude and frequency for k^{th} sinusoid for the i^{th} frame, and $\widehat{n}_i(t)$ is the noise part for i^{th} frame. Repeating this process for all the frames and performing overlap–addition on them, using the Hanning window, the final synthesized signal is obtained.

4.4 Proposed System

Karṅāṭic Music is characterised by the continuous nature of pitch contour. If we directly implement the SMS in a frame–based approach, this continuity cannot be achieved perfectly. Frame–based synthesis and overlap–addition would provide only a discrete approximation of the actual pitch contour. Hence, we propose to synthesise the entire frequency contour of all the notes present at the input without using windows. We parameterise the pitch contour, $\widehat{f}_0(t)$, using cubic splines, which makes the time and frequency scaling much easier. The frequency contours for the other sinusoidal components are generated as the integer multiples of $\widehat{f}_0(t)$, as given by

$$\widehat{f}_k(t) = k \cdot \widehat{f}_0(t). \tag{4.2}$$

We factorise the parameter $\widehat{A}_k(t)$ into two independent components. One component accounts for the different spectral weights of harmonic corresponding to different notes. The second component accounts for the time domain amplitude of the signal $\widehat{s}(t)$. Hence, $\widehat{A}_k(t)$ depends not only on the frequency domain weights of different notes but also on the attack, sustain and decay of the time domain waveform. As depicted in Fig. 4.5, time domain amplitude envelopes for the same notes vary differently for different conditions. The waveform of the same note evolves differently in time domain depending upon whether that note is played as a plain note or as part of a transition / *gamaka*. We model these two components separately. We represent the first component as $\widehat{a}_k(t)$, which is used to express only the spectral weights of the different sinusoidal components without considering the time domain amplitude envelope. The second component, $\widehat{E}(t)$, is the time domain amplitude envelope of the signal, which takes the attack, sustain and decay into consideration.

This component plays an important role in defining the timbre of different notes. Here also, we use parametric representation using cubic splines for making the time scaling of these contours easier. The synthesized signal for each note is given by

$$\widehat{s}(t) = \widehat{E}(t)\left(\sum_{k=0}^{\widehat{K}} \widehat{a}_k(t) \cdot \cos(\widehat{\phi}_k(t)) + \widehat{n}(t)\right), \tag{4.3}$$

where, $\widehat{\phi}_k(t)$ is the time varying phase contour for the k^{th} sinusoid, which is obtained by integrating $\widehat{f}_k(t)$. A simplified block diagram of the whole process is given in Fig. 4.7

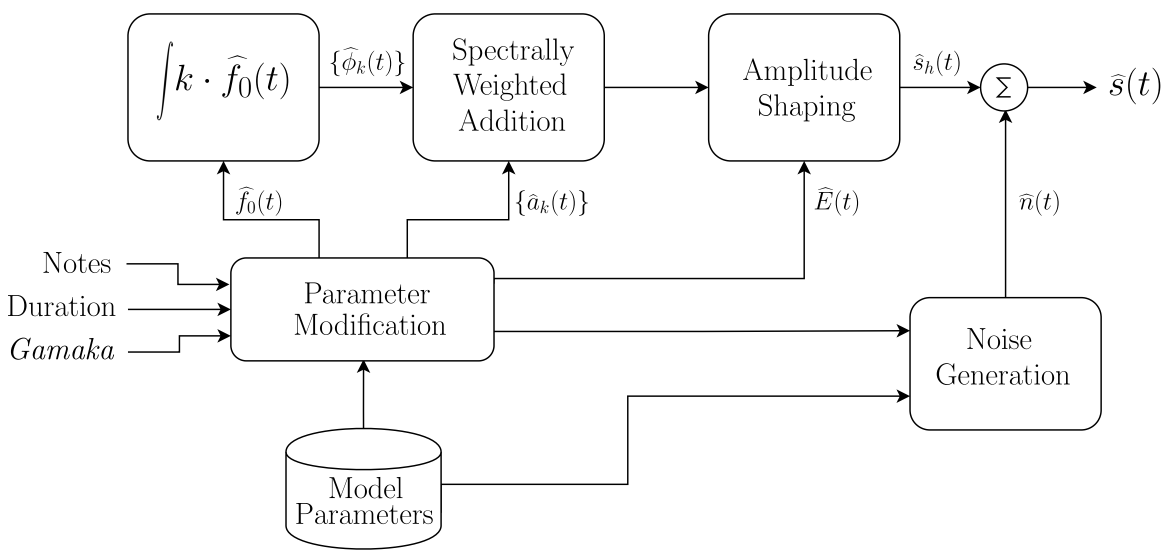


Figure 4.7: Block diagram of the proposed system

4.4.1 Dataset

We choose a popular song from the Karṅāṭic *rāga Mōhanam* played by a professional flautist. We use an F-scale Karṅāṭic bamboo flute for playing the song. The recording is manually segmented into three pitch classes, namely, plain notes, *gamaka*s and non-*gamaka* transitions. Eight different types of *gamaka*s, two non-*gamaka* transitions,

Table 4.1: Different Notes and their corresponding pitch values.

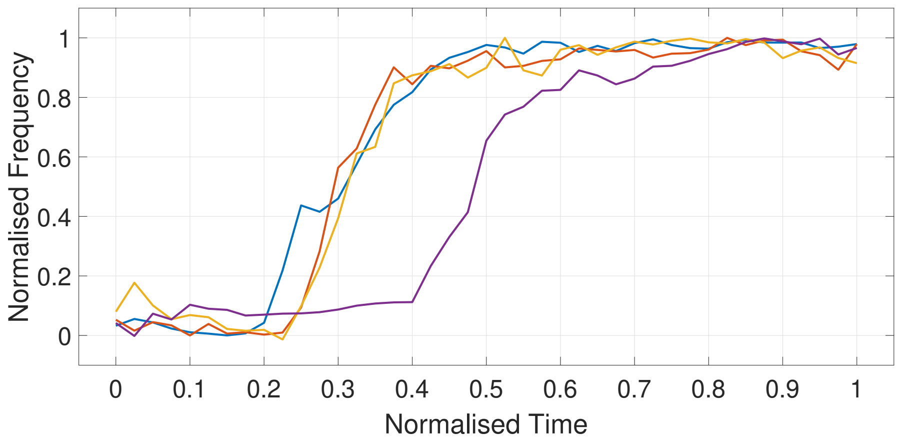
Octave	Note Label	$\widehat{f_0}$ (Hz)	Octave	Note Label	$\widehat{f_0}$ (Hz)
Lower	P1	286.5	Middle	P2	580.0
	D1	322.2		D2	643.0
Middle	S2	382.3	Upper	S3	762.0
	R2	438.7		R3	870.0
	G2	488.0		G3	970.0

and ten plain notes covering three octaves are present in the recorded samples. The dataset contains around 300 notes with 220 plain notes, 80 *gamaka*s and about 100 non-*gamaka* transitions. Samples for individual *gamaka*s vary from 6 to 18.

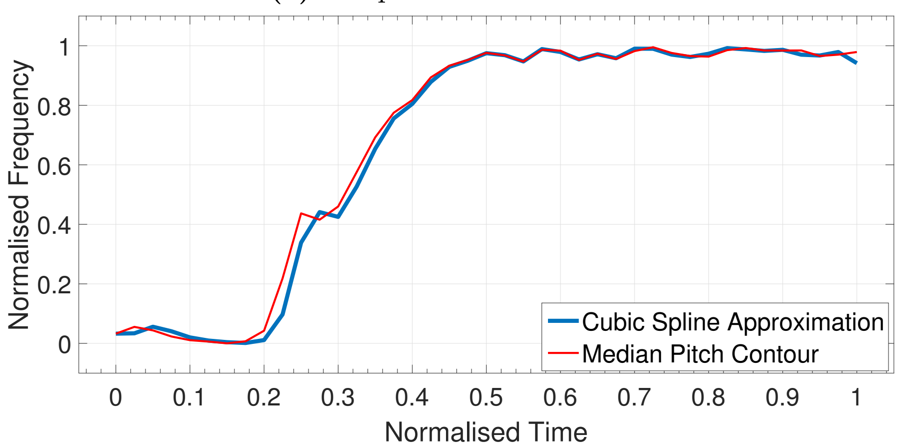
4.4.2 Estimation of Model Parameters

Model parameters are the spectral weights of different notes, pitch contour and time domain amplitude envelope shapes for different types of *gamaka*s / transitions, and the noise waveforms for different notes. We consider ten different notes, eight different *gamaka*s and two types of non-*gamaka* transitions in this work. This data is divided into 11 sub-classes for computing the pitch and amplitude envelope. All the plain notes fall into one subclass, and the eight *gamaka*s and two transitions form the other ten sub-classes.

Pitch contour of all the sound files belonging to the class 'plain notes' are extracted manually using PRAAT (Boersma and Weenink 2001). The median pitch value is found out for each note and is stored as the pitch value, $\widehat{f_0}$, of that note. Pitch values for all the notes used in our experiments are listed in Table 4.1. The alphabet denotes the note's name, and the number indicates the octave it belongs to. For example, 'P1' stands for the note *Pa* in the lower octave, 'P2' stands for the same note in the middle octave, and 'S3' stands for the note *Sa* in the upper octave. Pitch contours of all the sound samples belonging to each *gamaka* / transition are also extracted using PRAAT. For each subclass, the pitch contours of each sound are re-sampled to a standard size of 51 points to compensate for the difference in their duration. We compute the euclidean distance between all the sample pitch contours belonging to each subclass. We select the median pitch contour which best represents the subclass, by choosing the one which has minimum distance from all the other pitch contours



(a) Sample Pitch Contours



(b) Median pitch contour and its Spline Approximation

Figure 4.8: Time as well as frequency normalized pitch contours, their median pitch contour and its spline approximation for the *gamaka* named *Ētra Jāru*

belonging to that subclass.

The representative pitch contours for all the subclasses are normalised in time and frequency such that the time and frequency variations are limited between zero and one. Such a generalised shape can be then scaled in time and frequency to match the desired duration and pitch. These normalized pitch contours are parameterised using cubic spline modelling. A detailed explanation of the cubic spline modeling is given in Appendix A. We use 50 cubic splines to parameterise each pitch contour. These coefficients are stored as the representative pitch contour shape for each *gamaka* / transition. Fig. 4.8a shows the frequency and duration normalized pitch contours belonging to the *gamaka* named *Ētra Jāru*. The median pitch contour and its spline approximation are shown in Fig. 4.8b. We have 500 coefficients stored in the parameter set, corresponding to ten different subclasses. Normalized pitch contour shapes for different *gamaka* and non-*gamaka* transitions used in this experiment are shown in Figure.4.9 and Figure.4.10, respectively.

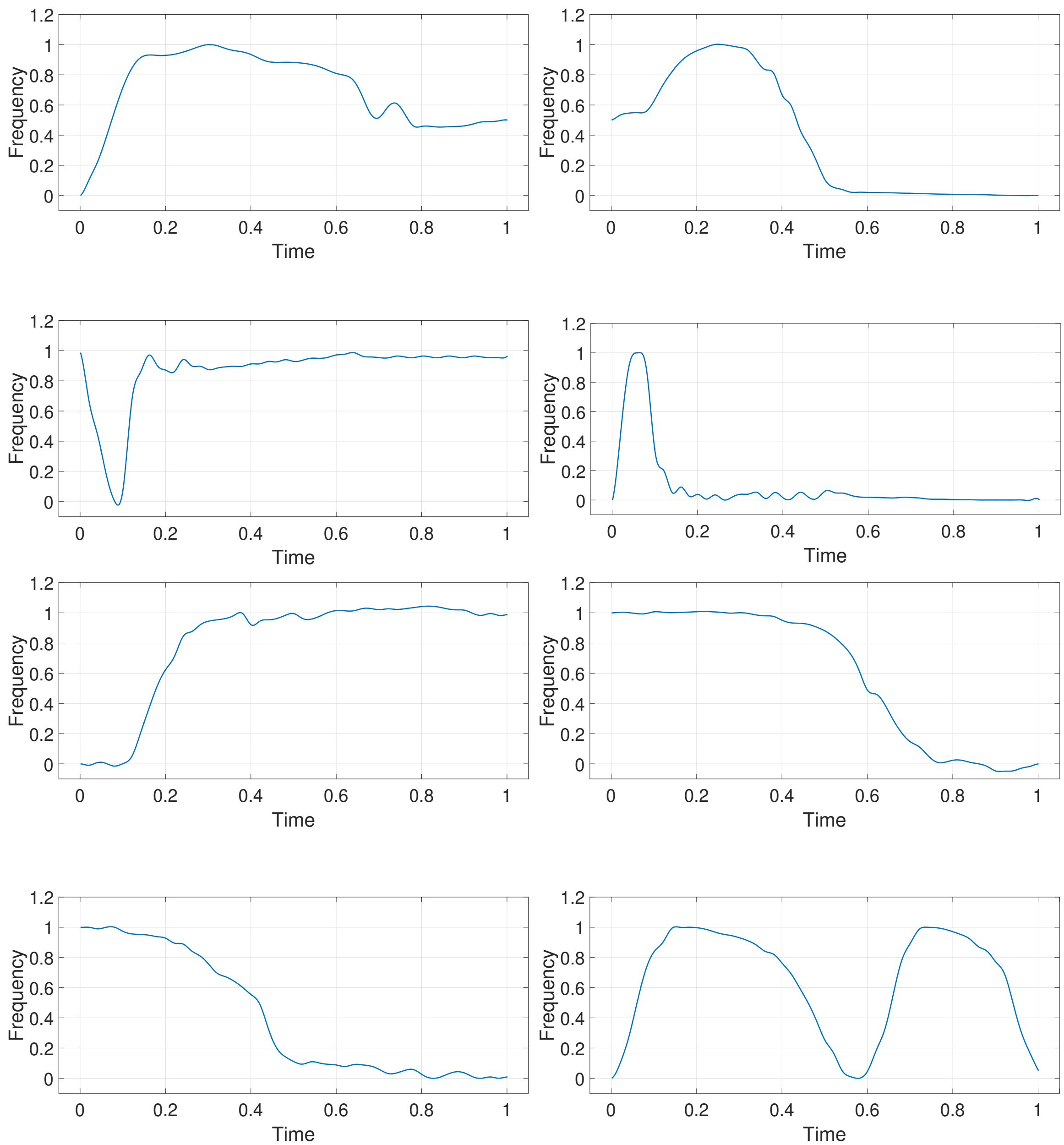


Figure 4.9: Normalized pitch contours of different *gamaka*s used in this work.

We repeat the same procedure for creating a parameter set of representative shapes for the time domain amplitude envelopes, $\widehat{E}(t)$. For this, the time domain amplitude

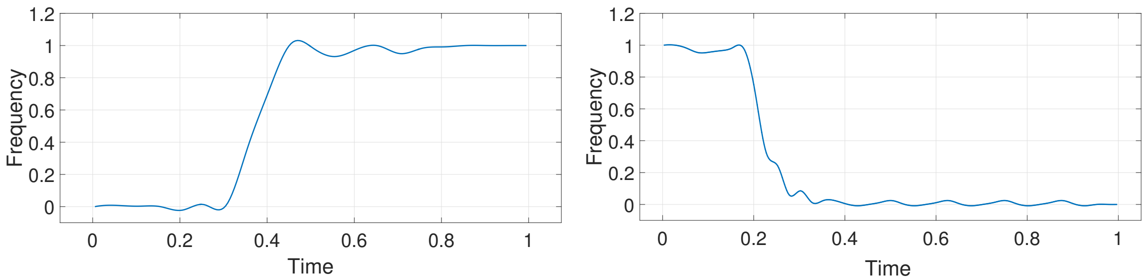


Figure 4.10: Normalized pitch contours of different non-*gamaka* transitions used in this work.

envelopes of all the sound files are extracted. Representative time domain amplitude envelope is found out by re-sampling the envelopes for each subclass and selecting the one with minimum distance from all others, as in the case of pitch contour. The time and amplitude normalised representative amplitude envelopes are parameterised using cubic splines, and the coefficients are stored in the parameter set. A total of 550 coefficients are stored for representing the amplitude envelopes for all the ten subclasses and the plain notes.

For calculating the spectral weights of different notes, we consider only the plain notes. In the first step, the effect of the amplitude envelope is nullified by dividing the note waveforms by the corresponding amplitude envelopes. Spectral analysis is performed on the resultant signal for finding out the weights, a_k, of each harmonic. We consider only the first ten harmonics for our experiments since the magnitude of higher harmonics is very small for the samples in our database. Thus, ten spectral weights are extracted for each of the ten plain notes, and they are stored in the database.

A separate dataset of wind noise is recorded using the same F-scale flute by blowing into the flute without creating resonance. As explained in Section 4.1.4, the octave difference does not significantly affect the wind noise characteristics. Hence, we use recorded wind noise corresponding to the notes in the middle octave only. Since there are five notes in one octave for the *rāga* used in our experiments, we use the noise waveforms corresponding to these five notes. Each of the noise waveforms has a duration of eight seconds.

4.4.3 Synthesis of Plain Notes

Input to our system is a text file containing information such as the note label, note duration in terms of the number of beat cycles and the *gamaka* / transition information. If there is no *gamaka* associated with a note, that note is assumed to be plain. If there is no *gamaka* between two notes, a non-*gamaka* transition is inserted between them at the time of synthesis. An example of the input text file is shown in Table 4.2.

Table 4.2: An excerpt from input text file.

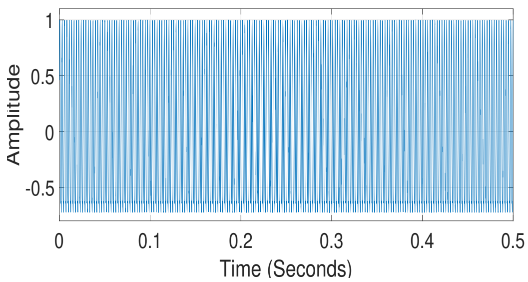
Note Label	**Duration**	***Gamaka***
	⋮	
G2	1	—
G2	1	*Sphuritham*
R2	2	*Irakka Jāru*
R2	1	*Prathyāhatham*
G2	2	—
	⋮	

For a plain note, the pitch contour and the spectral weights of the harmonics are constant for the entire time duration. Based on the note label, the corresponding pitch value, $\widehat{f}_0$ and the weights of different harmonics, $\widehat{a}_k$s, are found out from the parameter set. A constant pitch contour of this frequency is generated for the desired duration specified in the input. By the integer multiplication of this pitch contour, frequency contours for different harmonics are generated. Since all the frequency contours for a plain note are constant in time, the corresponding phase contours can be obtained by multiplication with time. The phase for the k^{th} harmonic between the time instants t_1 and t_2 is given by

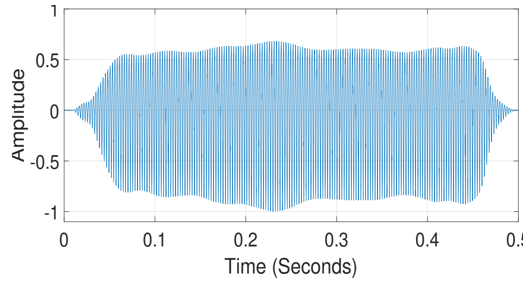
$$\widehat{\phi}_k(t) = \int_{t_1}^{t} \widehat{\omega}_k dt = 2\pi k \widehat{f}_0 t + \widehat{\phi}_k(t_1), \tag{4.4}$$

where $\widehat{\phi}_k(t_1)$ is the initial phase of the k^{th} harmonic. The initial phase at time t_1 is added to make sure that the phase is continuous at the note boundaries.

For generating the time domain amplitude envelope $\widehat{E}(t)$, parameterised representative shape for the plain note is selected from the parameter set, and it is time–scaled to match the desired note duration. Abrupt jumps at the note boundaries are avoided by making the envelope and its first derivative continuous at the endpoints. We discretize these parameters by evaluating the cubic splines at each sample points. In our experiments we use a sampling frequency of 32kHz. The sinusoidal part is synthesized by the weighted addition of sinusoidal components and multiplying the sum with an



(a) Without envelope



(b) With envelope

Figure 4.11: Waveform of synthesized note *Sa* with and without time domain amplitude envelope.

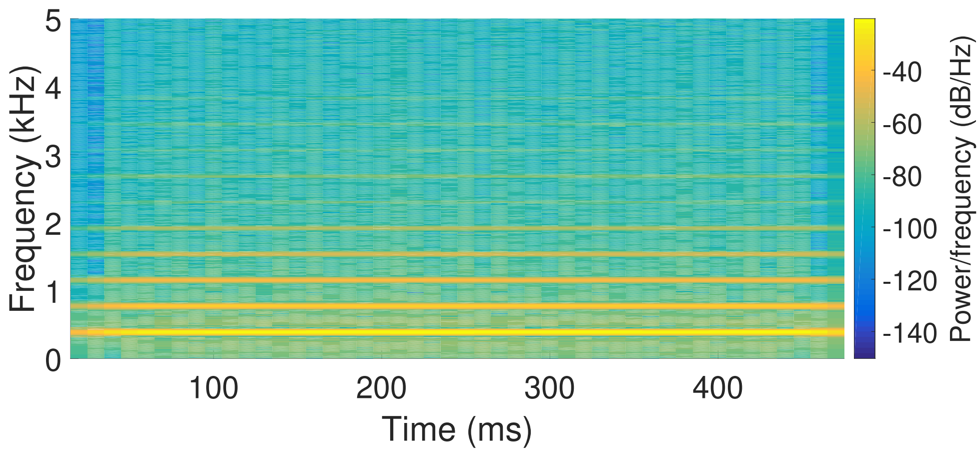


Figure 4.12: Spectrogram of synthesized plain note *Sa*

amplitude envelope as given by

$$\widehat{s}_h(t) = \widehat{E}(t)\left(\sum_{k=0}^{\widehat{K}} \widehat{a}_k(t) \cdot cos(\widehat{\phi}_k(t))\right) \tag{4.5}$$

For generating the noise part, we use pre–recorded noise signals corresponding to the input notes. The noise signals are lengthened or shortened to match the desired input duration. Lengthening is performed by looping the same waveform multiple times and shortening performed by truncation. Duration modified noise signals are modulated by the amplitude envelope $\widehat{E}(t)$, and final synthesis is performed using Equation (4.3). Waveforms of the synthesised signal with and without amplitude envelope are shown in Fig. 4.11 and the spectrogram for the final synthesised signal is shown in Fig. 4.12.

4.4.4 Synthesis of Transitions

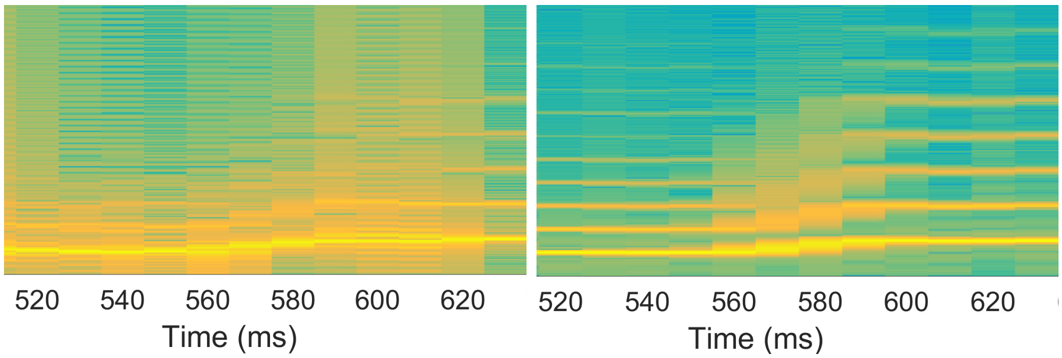
In the case of *gamaka* and non-*gamaka* transitions, pitch contour is not assumed to be constant. If two adjacent notes are different and there is no *gamaka* associated with the second note, a non-*gamaka* transition is inserted between the two notes. After parsing the input, every pair of notes are checked for the presence of *gamaka* between them. If no *gamaka* is present in the second note, information such as the starting note, ending note, and duration are extracted. Based on the starting and end notes, the corresponding note frequencies are found out from the parameter set. If the second note is higher in frequency than the first one, an upward transition is to be added, and if the second note's frequency is lower, a downward transition is to be added.

Once the type of transition is finalized, the parametric form of its pitch contour is selected from the database. The parametric form is stored in the database such that its time and frequency vary between zero and one. It is scaled in time to match the duration specified at the input and also scaled in frequency to generate a transition pitch contour, $\widehat{f_0}(t)$, between the first note and the second note. For example, if the starting and ending notes' frequencies are f_1 and f_2, and the duration extends from time instants t_1 to t_2, then the representative pitch contour is scaled such that its frequency varies from f_1 to f_2 and the time duration spans from t_1 to t_2. The endpoint slopes of the parametric form are set to zero to enable smooth concatenation with adjacent pitch contours. Frequency contours of other harmonics are obtained by integer multiplication of this scaled pitch contour.

Frequency contours of individual harmonics are integrated with respect to time to obtain the phase contours. Since the cubic splines are used for the parametric representation of the pitch contour, the result of integration can be obtained in closed form. Phase contour of k^{th} harmonic between the time instants t_1 to t_2 is expressed as

$$\widehat{\phi}_k(t) = 2\pi k \int_{t_1}^{t} \widehat{f_0}(t)\ dt + \widehat{\phi}_k(t_1), \tag{4.6}$$

where $\widehat{f_0}(t)$ is pitch contour. The constant of integration, $\widehat{\phi}_k(t_1)$, is the initial phase of the k^{th} harmonic at the starting point of the current segment being synthesized. By incorporating this phase correction term, phase smoothly varies across the note boundaries. Spectrograms for the note boundary with and without phase correction are shown in Fig. 4.13. Phase contours of the transition are appended smoothly to the phase contours of the previous note to generate the continuous phase contour for



(a) without phase continuity **(b)** with phase continuity

Figure 4.13: Spectrograms of synthesized transition with and without imposing phase continuity.

the entire input sequence.

As opposed to the plain note case, spectral weights of different harmonics are not constant in the case of note transitions. For generating the continuously varying spectral weights, we first extract weights of different harmonics for all the constituent notes from the parameter set. The regions where each note appears are identified with the help of pitch contour. For the regions where each note is active, corresponding spectral weights are assigned. For the region where the note transition occurs, the weights are found by interpolation. We use cubic spline interpolation for obtaining the smooth and continuously varying spectral weights, $\widehat{a}_k(t)$. An example depicting the pitch contour for the transition from the note Sa to the note Pa and the corresponding spectral weights for the first three harmonics are shown in Fig. 4.14. As in the case of plain note synthesis, the parametric form of the time domain amplitude envelope for the desired transition is selected from the database. It is scaled in time to match the duration of the actual transition. Smooth concatenation with the previous note's amplitude envelope is also performed before using the envelope to modulate the weighted sum of the sinusoids. The harmonic part of the synthesized signal is obtained by multiplying the weighted sum of the harmonically related sinusoids with the amplitude envelope as given by Equation (4.5).

Since there are two notes present in the pitch contour of a transition, noise signals corresponding to both the notes are to be added at the respective time instants. We perform a windowed addition of the duration modified noise waveforms to generate the noise part for the transition. Towards this, we introduce a function called activation function for each of the notes. The activation function of a note takes the value unity

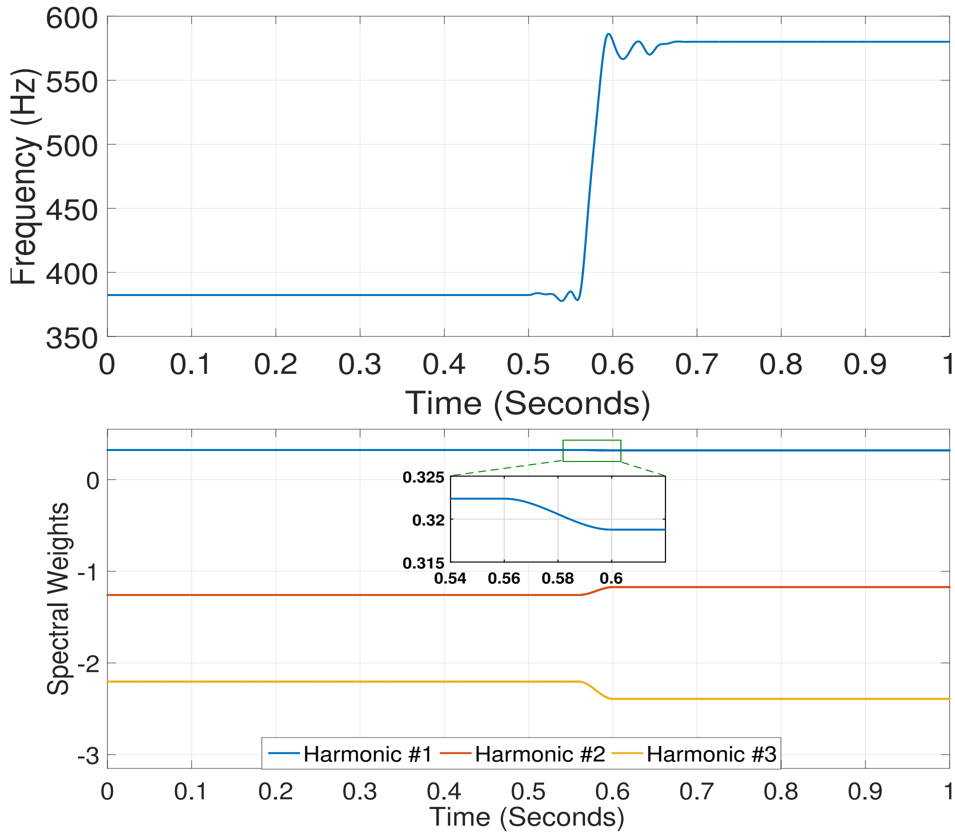


Figure 4.14: Pitch contour and corresponding spectral weights of first three harmonics for an upward transition from the note *Sa* to the note *Pa*.

for the entire duration where that note is active. Whenever the note is absent, its activation function is at zero. The time instances for which a note is present or absent is located with the help of pitch contour.

For example, activation functions for different notes corresponding to the transition from note *Sa* to the note *Pa* are shown in Fig. 4.15. As can be seen from the pitch contour shown in Fig. 4.14, when only the *Sa* is present in the pitch contour, only the activation function corresponding to *Sa* is at one for that entire duration. Same is true for the case of the note *Pa*. But, in the transition region, corresponding activation functions for *Sa* and *Pa* are found out by spline interpolation. Activation function for all the other eight notes are at zero for the entire duration. The noise waveforms for all the notes are selected from the database and are time–scaled to match the input duration. They are multiplied with the corresponding activation functions to generate the active noise waveforms for the particular duration. All the active noise waveforms

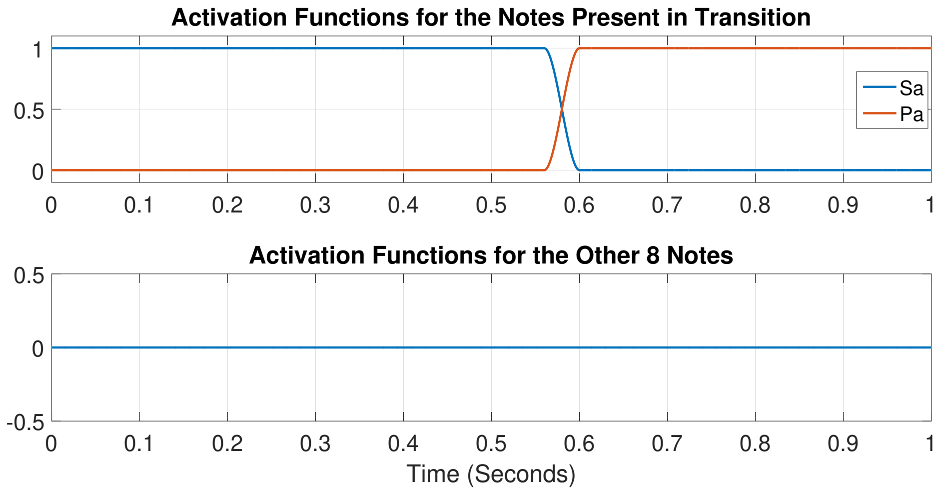


Figure 4.15: Activation functions for the notes corresponding to the upward transition from *Sa* to *Pa*.

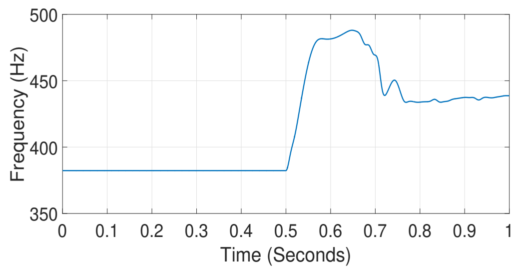
are added together to generate the noise part of the synthesised signal, as given by

$$\widehat{n}(t) = \sum_{i=1}^{N} A_i(t) \cdot n_i(t), \tag{4.7}$$

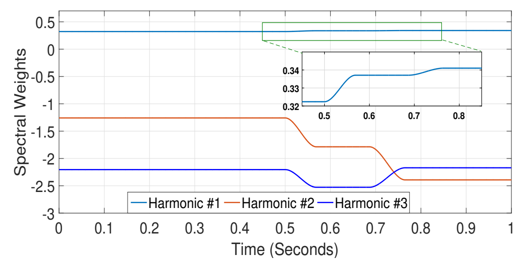
where N is the number of notes in one octave, $A_i(t)$ is the activation function, and $n_i(t)$ is the duration modified noise waveform corresponding to the i^{th} note in the database. Adding the harmonic part and the envelope modulated noise part together gives the final synthesized signal as given in Equation 4.3.

4.4.5 Synthesis of *Gamaka*

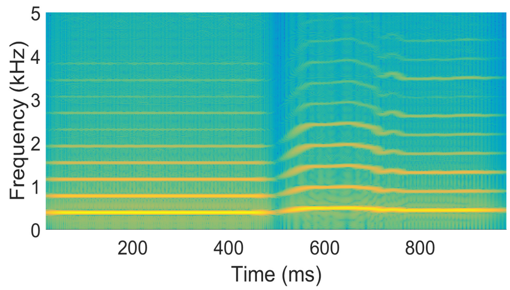
*Gamaka*s are also synthesized in the same manner as that of the transitions. Some *gamaka*s may contain more than two constituent notes in their pitch contour. In such cases, the spectral weights of all those constituent notes need to be interpolated to find the spectral weights of the *gamaka* region. Also, the noise activation functions for more than two notes will have non-zero values during the course of such *gamaka*s. The pitch contour, interpolated spectral weights, noise activation functions, synthesized signal and spectrum for such a *gamaka* are shown in Fig. 4.16.



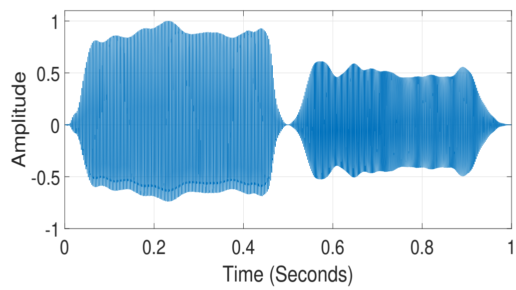
(a) Pitch contour



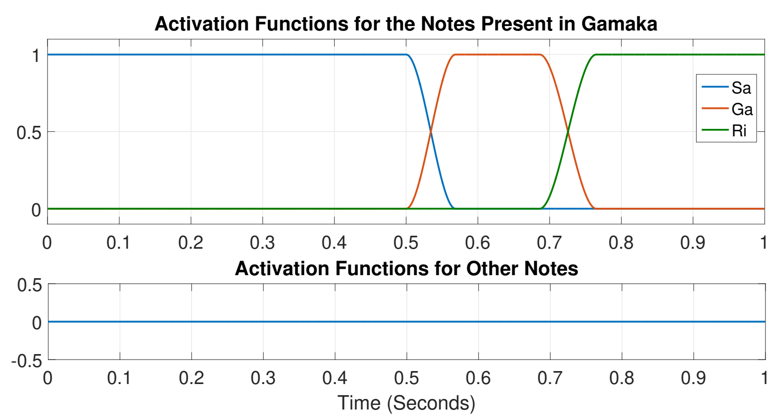
(b) Spectral weights of first three harmonics



(c) Spectrogram of the synthesised signal



(d) Synthesised signal



(e) Noise activation functions

Figure 4.16: Pitch contour, interpolated harmonic weights, spectrogram of the synthesized signal, waveform of synthesized signal and noise activation functions for a three note *gamaka*.

4.5 Subjective Quality Evaluation

We conduct listening tests for evaluating the quality of the synthesized music. Two different subjective quality assessments are conducted. The first one is to analyse the impact of different steps involved in the proposed method in deciding the tonal quality of the synthesized flute tone. In the second experiment, we compare our synthesis method with the existing spectral synthesis methods. We try to analyse two parameters of the synthesized music in this assessment – the tonal quality and the correctness of *gamaka* rendition.

For the first listening experiment, we choose five different excerpts from different songs and generate five different stimuli[1] from each of these excerpts. The first one is the synthesized tone using the proposed method without adding the amplitude envelope and the wind noise. The second one is the synthesised flute music with the amplitude envelope and without adding the wind noise. The third one is the synthesised flute music using the complete model, including the harmonics, amplitude envelope and noise. Fourth one is the original recording from a real flute, and the fifth one is generated by passing the original recording through a high pass filter with cut off frequency 3.5kHz.

We choose trained Karṅāṭic flautists and vocalists as the listeners for subjective evaluation. Each listener is asked to listen to the five different versions of all the five audio files. He/she is asked to rate the tonal quality of each audio file. Tonal quality is a measure of how well the tone resembles an actual flute tone. A five–point scale is used to rate the quality, with one corresponding to the worst quality and five corresponding to the best. Only those responses which rated the original flute recording as the best one are considered for the evaluation. Acceptable responses for all the audio files are compiled, and the mean ratings are found out. The results are summarised in Table 4.3.

The results show that modelling the amplitude envelope and adding the noise help improve the quality of the synthesised tone. To test the statistical significance of these improvements, we perform students paired T–test, and the results show that the improvements are significant at a 95% confidence interval.

For the second listening experiment, we use excerpts from five different songs, each excerpt consisting of 15 seconds. Each of these five files are synthesised using

[1] Audio samples of all these versions are available at https://sites.google.com/view/muraleeravam/
.

Table 4.3: Mean Opinion Score for Assessment I

Method	**Tonal Quality**
Harmonic Part Only	2.7
Harmonics + Amp. Envelope	2.9
Harmonics + Noise + Amp. Envelope	3.2

Gāyaka (Subramanian 2013), basic SMS (Serra *et al.* 1997) and the proposed method. In addition to this, we record the original flute music for each of these five audio files. We also include a high pass filtered version of the original recording as the fifth stimulus. We manually add the description for each *gamaka* to synthesise the song in Gāyaka software. For every *gamaka* present in the song, we specify the constituent notes and the duration for each of them in the entire course of that *gamaka*.

To synthesise the SMS version of the song, we generate harmonic frequency contours and harmonic weight progressions in continuous time using cubic spline interpolation. These continuous time parameters are then subdivided into frames of length 20 ms. During each frame, the values of frequency contours and the harmonic weights are made constant by replacing them with the median values of these parameters in that frame. The noise part is synthesised as explained in the previous section. The amplitude envelope is not modelled explicitly. After every frame is synthesised, overlap–addition is performed using Hanning window.

Each listener is asked to listen to the five different versions [2] of all the five audio files and rate two parameters of each audio file – the tonal quality and the propriety of *gamaka* rendition– on a five–point scale. Tonal quality is a measure of how well the tone resembles an actual flute tone, and the propriety of *gamaka* rendition is a measure of how well the *gamaka*s are synthesised in comparison with the traditional, standard renditions of *gamaka*s in KM. Acceptable responses for all the audio files are compiled, and the mean ratings are found out. The results are summarised in Table 4.4. There is an improvement of both the tonal quality and the correctness of rendering *gamaka*s using the proposed method when compared to the other two methods. The improvement observed over the conventional frame–based SMS is commendable. In this experiment also, the statistical test results show that the improvements are

[2] Audio samples of all these versions are available at https://sites.google.com/view/muraleeravam/ .

Table 4.4: Mean Opinion Score for Assessment II

Method	**Tonal Quality**	**Correctness of *Gamaka***
Gāyaka (Subramanian 2013)	3	3
SMS (Serra *et al.* 1997)	2.4	2.3
Proposed	3.5	3.6

significant at a 95% confidence interval.

4.6 Implementation of the Complete System

We combine the *gamaka* classification system and the flute synthesis system to obtain the flute music from the textual notations alone. We choose two popular songs from the *rāga Mōhanam* to test the performance of the combined system. The skeletal notations for these songs are prepared in the form of a ".csv" file containing the note label, duration of each note and the *gamaka*s associated with each note.

The notation file is fed to the feature extraction stage, and the note based features are extracted. We try to use two different training sets for training the Random Forest Classifier (RFC). The first one is called *Seen Rāga* training set, which contains five songs from the *rāga* used for testing (*Mōhanam*). The other training set is called *Unseen Rḡa* training set since it does not contain any song from the *rāga* used in the test set. We test the performance of the system using both these training sets.

After training the RFC with each of these training sets, we feed the features extracted from the skeletal notations of the test songs for predicting *gamaka* presence. Obtained results are saved as a ".csv" file, where an additional column is added to indicate the status of the prediction. If a *gamaka* is predicted, the status is *True*, and if no *gamaka* is predicted, the status is *False*. Based on this status, only the *gamaka* notes are separated from the feature file, and is fed to the *gamaka* classification system for predicting the type of *gamaka*s. The same set of songs are used for training both these systems. RFC predicts the type of *gamaka*s, and the final result is stored in a ".csv" file. This file contains the entire skeletal notation of the test songs plus a column named *Gamaka*. This column contains the type of *gamaka* if any *gamaka* is predicted associated with a note and is left blank if no *gamaka* is predicted.

This file is fed to the flute music synthesis system for generating flute music corre-

Table 4.5: Mean Opinion Score for performance assessment of the complete system

Method	Propriety of rendition
Non-*gamaka* synthesis	3.3
Gamaka synthesised using Seen *Rāga* Training	3.4
Gamaka synthesised using Unseen *Rāga* training	3.1

sponding to the test songs. We try to generate two different outputs with the synthesis systems. The first one is synthesised using the predicted *gamaka*s as the input, and the second one is synthesized without considering any *gamaka*s. This is done for testing the effectiveness of the *gamaka* prediction system. Even though *gamaka*s are not considered, non-*gamaka* transitions are synthesized in both these versions. The same procedure is repeated with the second training set, and *gamaka* and non-*gamaka* versions are synthesized. Thus, four different versions for each test song are generated - *gamaka* and non-*gamaka* versions corresponding to *Seen Rāga* and *Unseen Rāga* training.

4.6.1 Subjective Quality Evaluation of the complete System

As in the previous listening experiments, we choose the same set of listeners for evaluating the performance of the system. We also choose the same set of songs used in the previous experiments. We prepare four stimuli corresponding to each excerpt of the test songs. The first one is the original song played on a bamboo flute, the second one is the non-*gamaka* version of synthesized song, the third one is the synthesized song containing *gamaka*s predicted using the help of *Seen Rāga* training set, and the fourth one is the high pass filtered version of the original song. Listeners are asked to rate the propriety of rendition on a scale of one to five as in the previous experiments. We select only those responses which rated the original flute rendition with a maximum rating and the low pass filtered version with a minimum rating. The mean opinion scores obtained are listed in Table 4.5

We perform a students' paired T–test to check the statistical significance of these results. Analysis shows that the results are not statistically significant at a 95% confidence interval. This can be attributed to the inaccuracy of *gamaka* prediction system. There are 50 *gamaka*s in the test songs out of 105 notes. Only six *gamaka*s are predicted by the system. Thus, the majority of the original *gamaka*s are undetected by the system. Out of the six *gamaka*s detected by the system with the *Seen Rāga* training, five *gamaka*s are actually present in the original song, and one is false positive. Hence, even though the precision of our system is good, poor recall resulted in not detecting most of the actual *gamaka*s.

We believe that the degradation in performance of *gamaka* prediction and classification system is due to the class imbalance in the training data. Only 30% of the notes in the training set contain any *gamaka* and the rest of the notes are plain notes, where roughly 50% of the test song contain *gamaka*s. We hope that adding more samples of *gamaka* notes into the training data will improve the performance of the entire system. In addition, the songs used for testing the system belong to a *rāga* which has only five songs in the training set.

Even though the *gamaka*s are not synthesised for the non-*gamaka* synthesis, the non-*gamaka* transitions between the notes are synthesised. The upward transition is somewhat similar to the *gamaka* called *Ētra Jāru* in shape. Similarly, the downward transition is similar to *Irakka Jāru*. Due to this similarity, the above mentioned *gamaka*s can be considered "partially synthesised" even in the case of non-*gamaka* synthesis. This can be a reason for the lack of significant improvement in MOS of *gamaka* and non-*gamaka* synthesis.

4.7 Conclusion

We propose an efficient method to synthesise plain notes and *gamaka*s for Karṇāṭic flute music by extending the sinusoidal model. Towards this, we try to model three important aspects of flute sounds – the frequency contours, weights of different harmonics and the time domain amplitude envelope. All these are modelled as continuous functions of time without using the overlap–add method.

We analyse different recordings to find out a representative shape for the pitch contour and time domain amplitude envelopes of each plain note, transition and *gamaka*. We represent them in a parametric form by means of cubic splines to facilitate the time and frequency scaling to match the input pitch and durations. The progression of harmonic weights with respect to time is also modelled using cubic splines. Synthesis is performed by weighted addition of harmonically related sinusoids, which is then modulated by a time domain envelope. We use the pre-recorded wind noise corresponding to each of the notes and modulate them using a noise activation function before adding to the weighted sum of harmonics.

Mean Opinion Score obtained from the subjective evaluation suggests that modelling the time domain amplitude and adding the wind noise improve tonal quality. Another evaluation is conducted to compare the performance of the proposed approach with the existing Karṅāṭic flute synthesis methods. Results suggest that the proposed method is better in terms of tonal quality and the correctness of rendering *gamaka*s. Hypothesis tests performed on the subjective evaluation results show that the observed improvements are statistically significant over a 95% confidence interval.

We combine two subsystems to build a complete system for synthesising flute music from the skeletal notations of a song. The first stage predicts the presence and type of *gamaka*s from the notation file, and based on these predicted *gamaka*s the flute music is synthesised. The results for *gamaka* prediction show a low recall value, and most of the actual *gamaka*s are undetected. As a result, the synthesised songs are similar to those without *gamaka*s. Statistical analysis of the Mean Opinion Scores also suggests that there is no significant improvement in the quality of the song even after adding the predicted *gamaka*s to the notes.

Chapter 5

CONCLUDING REMARKS

In this work, we implemented a system for synthesising Karṅāṭic Music (KM) on a bamboo flute with only minimal textual notations as the input. Input to the system contained only textual information regarding label of the note, duration of the note and the *rāga* of the song. We divided this complete system into two separate sub–systems. The first one is a *gamaka* prediction and classification system to predict the presence and type of *gamaka* from the skeletal notations. The second sub–system was designed to take the additional input of *gamaka* information associated with each note and to synthesise the bamboo flute music corresponding to the input notations.

We created a textual dataset containing around 30000 notes belonging to 22 *rāga*s. We proposed features based on note frequency, duration and *rāga*, and employed a Random Forest Classifier (RFC) for predicting the presence and type of *gamaka*s. The classifier was trained using around 11000 notes, and two different test sets were used to evaluate the performance of the classifier. Accuracy values obtained for *gamaka* presence prediction was around 76.5% and 68.2% for these two test sets. Similarly, the accuracy values for predicting the type of *gamaka* was found to be 70% and 59%. Analyses performed on training data revealed that the *gamaka*s exhibit certain characteristic patterns of note sequences, and the uniqueness of such patterns are prominent when the length of the sequence is three.

To design the second sub–system, we modelled the parameters of bamboo flute tone in continuous time domain. The parameters such as frequency contours, spectral weights of the harmonics, and the time domain amplitude envelope were parameterised using cubic splines to achieve this. In contrast to the conventional methods for synthesising the flute tones, we did not make use of any window–based methods for the synthesis, which helped in reproducing the truly continuous nature of the parameters mentioned above. We conducted subjective quality assessment to compare the efficacy of the proposed method with the popular methods for synthesising Karṅāṭic flute music. On a five–point scale, Mean Opinion Score (MOS) of 3.6 was reported for the proposed method, while the other frame-based methods scored 2.3 and 3. We combined the two sub–systems to build the complete flute music synthesis system. In this case too, we observed a MOS of 3.4, which was close to that obtained for the second sub–system alone.

Our dataset for *gamaka* prediction system contained only 80 songs, while the dataset for the second sub–system included only one *rāga*. Lack of samples in the training dataset could be the reason for the reduced recall rate associated with the *gamaka* prediction and classification tasks. We do hope that a larger dataset for both the *gamaka* prediction and synthesis tasks will yield better results, and the proposed system can be used for synthesising Karṅāṭic flute music for more *rāga*s.

Bibliography

Arora, V., **L. Behera**, and **P. Sircar**, Singing Voice Synthesis for Indian Classical Raga System. *In IET Irish Signals and Systems Conference (ISSC 2009)*. 2009.

Ashtamoorthy, A., **P. Prasad**, **S. Dhar**, and **D. Vijayasenan**, Frequency contour modeling to synthesize natural flute renditions for carnatic music. *In 2018 International Conference on Signal Processing and Communications (SPCOM)*. IEEE, 2018.

Ayers, L., Synthesizing trills for the chinese dizi. *In Proceedings of the International Computer Music Conference, ICMC, Singapore.*. 2003.

Ayers, L., Synthesizing timbre tremolos and flutter tonguing on wind instruments. *In Proceedings of the International Computer Music Conference, ICMC, Miami, Florida, USA*. 2004.

Ayers, L. (2005). Synthesising chinese flutes using csound. *Organised Sound*, **10**(1), 37–49.

Belle, S., **R. Joshi**, and **P. Rao** (2009). Raga Identification by Using Swara Intonation. *Journal of ITC Sangeet Research Academy*, **23**.

Benade, A., *Fundamentals of Musical Acoustics*. Dover Books on Music Series. Dover Publications, 1990. ISBN 9780486264844.

Blacking, J., *How musical is man?*. University of Washington Press, 1974.

Boersma, P. and **D. Weenink** (2001). Praat, a system for doing phonetics by computer. *Glot international*, **5**, 341–345.

Breiman, L. (2001). Random Forests. *Mach. Learn.*, **45**(1), 5–32. ISSN 0885-6125. URL `https://doi.org/10.1023/A:1010933404324`.

Cohen, W. W., Fast Effective Rule Induction. *In Proceedings of the twelfth international conference on machine learning*. 1995.

Conard, N., **M. Malina**, and **S. Münzel** (2009). New flutes document the earliest musical tradition in southwestern germany. *Nature*, **460**, 737–40.

Das, D. and **M. Choudhury**, Finite State Models for Generation of Hindustani Classical Music. *In Proceedings of International Symposium on Frontiers of Research in Speech and Music*. 2005.

Day, C. R. (1891). The music and musical instruments of southern india and the deccan.

Dïaz, A. and **R. Mendes**, Analysis synthesis of the andean quena via harmonic band wavelet transform. *In Proc. Digital Audio Effects (DAFx-15), Trondheim, Norway*. 2015.

Dikshitar, S., *Sangīta Sampradāya Pradarsini*. English Web Version, 2008. URL http://www.ibiblio.org/guruguha/ssp.htm.

Duda, R. O., **P. E. Hart**, and **D. G. Stork**, *Pattern Classification*. John Wiley & Sons, 2012.

Gayathri, R. K. (1987). Improvisation in Alapana Performance: A Comparative View of Raga Shankarabharana. *Yearbook for Traditional Music*, **19**, 45–64. ISSN 07401558. URL http://www.jstor.org/stable/767877.

Giraldo, S. (2016). *Computational Modelling of Expressive Music Performance in Jazz Guitar: A Machine Learning Approach*. Ph.D. thesis, Pompeu Fabra University, Barcelona.

Giraldo, S. I. and **R. Ramirez** (2016). A Machine Learning Approach to Discover Rules for Expressive Performance Actions in Jazz Guitar Music. *Frontiers in psychology*, **7**.

Giraldo, S. I. and **R. Ramírez** (2016). A Machine Learning Approach to Ornamentation Modeling and Synthesis in Jazz Guitar. *Journal of Mathematics and Music*, **10**(2), 107–126.

Gómez, F., **A. Pikrakis**, **J. Mora**, **J. M. Dıaz-Bánez**, **E. Gómez**, and **F. Escobar**, Automatic Detection of Ornamentation in Flamenco. *In Fourth International Workshop on Machine Learning and Music MML*. 2011.

Gupta, C. and **P. Rao**, Objective Assessment of Ornamentation in Indian Classical Singing. *In Speech, Sound and Music Processing: Embracing Research in India*. Springer, 2011, 1–25.

Helmholtz, H., *On the sensations of tone*. Courier Corporation, 2013.

Horner, A. and **L. Ayers** (1998). Modeling acoustic wind instruments with contiguous group synthesis. *Journal of the Audio Engineering Society*, **46**(10), 868–879.

James, K. (2012). Real time raga detection and analysis using computer.

Keefe, D. H. (1990). Woodwind air column models. *The Journal of the Acoustical Society of America*, **88**(1), 35–51.

Kreutzer, C., **J. Walker**, and **M. O'Neill**, A parametric model for spectral sound synthesis of musical sounds. *In 2008 International Conference on Audio, Language and Image Processing*. IEEE, 2008.

Krishna, T. M. and **V. Ishwar**, Carnatic music: Svara, gamaka, motif and rāga identity. *In Proceedings of the 2nd CompMusic Workshop; 2012 Jul 12-13; Istanbul, Turkey. Barcelona: Universitat Pompeu Fabra; 2012.*. Universitat Pompeu Fabra, 2012.

Krishnaswamy, A., Application of pitch tracking to south indian classical music. *In Multimedia and Expo, 2003. ICME'03. Proceedings. 2003 International Conference on*, volume 3. IEEE, 2003.

Larose, D. T., *Data Mining and Predictive Analytics*. John Wiley & Sons, 2015.

McIntyre, M. E., **R. T. Schumacher**, and **J. Woodhouse** (1983). On the oscillations of musical instruments. *The Journal of the Acoustical Society of America*, **74**(5), 1325–1345.

Miryala, S. S., **K. Bali**, **R. Bhagwan**, and **M. Choudhury**, Automatically identifying vocal expressions for music transcription. *In ISMIR*. 2013.

Mohapatra, S., **A. Awasthi**, and **A. Mukherjee** (2010). *Automatic Music Generation for Indian Classical Music*. Project report, Indian Institute of Technology Kanpur, Kanpur, India.

Nagavi, T. C. and **N. U. Bhajantri**, Overview of automatic indian music information recognition, classification and retrieval systems. *In 2011 International Conference on Recent Trends in Information Systems*. IEEE, 2011.

Narayan, A. and **N. Singh**, Detection of micro-tonal ornamentations in dhrupad using a dynamic programming approach. *In Proceedings of the 9th Conference on Interdisciplinary Musicology–CIM14. Berlin, Germany*. 2014.

Pérez, A. (2009). Enhancing spectral synthesis techniques with performance gestures using the violin as a case study. *Department of Information and Communication Technologies*.

Polotti, P. and **G. Evangelista** (2001). Analysis and synthesis of pseudo-periodic 1/f-like noise by means of wavelets with applications to digital audio. *EURASIP Journal on Advances in Signal Processing*, **2001**(1), 584201.

Polotti, P. and **G. Evangelista** (2007). Fractal additive synthesis. *IEEE Signal Processing Magazine*, **24**(2), 105–115.

Powers, D. M. W. (2011). Evaluation: From Precision, Recall and F-measure to ROC, Informedness, Markedness & Correlation. *Journal of Machine Learning Technologies*, **2**(1), 37.

Pratyush (2010). *Analysis and Classification of Ornaments in North Indian (Hindustani) Classical Music*. Ph.D. thesis, Universitat Pompeu Fabra, Barcelona.

Ramamurthy, S. and **M. Raghavan**, Filter design for synthesis of musical notes: A multidimensional feature-based approach. *In 2013 IEEE International Conference on Signal and Image Processing Applications*. 2013.

Ranjani, H., **D. Paramashivan**, and **T. V. Sreenivas**, Quantized melodic contours in indian art music perception: Application to transcription. *In ISMIR*. 2017.

Rao, S., Culture specific music information processing: A perspective from hindustani music. *In 2nd Comp-Music Workshop*. 2012.

Rocamora, M., **E. Lopez**, and **L. Jure**, Wind instruments synthesis toolbox for generation of music audio signals with labeled partials. *In Proceedings of 2009 Brazilian Symposium on Computer Music SBCM09*, volume 2. 2009.

Sambamoorthy, P., *South Indian Music*, volume 3. Indian Music Pub. House, 1964.

Sambamoorthy, P., *The Flute*. Indian music Publishing House, 1982. URL `https://books.google.co.in/books?id=IwbOOwAACAAJ`.

Sathyanarayana, R., *Karnataka music as aesthetic form*, volume 14. Project of History of Indian Science, Philosophy, and Culture, Centre for . . . , 2004.

Scavone, G. P. and **P. R. Cook**, Real-time computer modeling of woodwind instruments. *In Proc. Int. Symp. on Musical Acoustics (ISMA-98), Leavenworth, WA*. Citeseer, 1998.

Serra, X. and **J. Smith** (1990). Spectral modeling synthesis: A sound analysis/synthesis system based on a deterministic plus stochastic decomposition. *Computer Music Journal*, **14**(4), 12–24. ISSN 01489267, 15315169.

Serra, X. *et al.* (1997). Musical sound modeling with sinusoids plus noise. *Musical signal processing*, 91–122.

Shahinda, *Indian Music: By Shahinda Begum Fyzee-Rahamin., with Preface by F. Gilbert Webb*. William Marchant & Company, 1914.

Shringy, R. K. and **P. L. Sharma**, *Saṅgīta-Ratnākara of Śārṅgadeva*, volume I. Motilal Banarsidass, 1978.

Sinha, P. (2008). Artificial Composition: An Experiment on Indian Music. *Journal of New Music Research*, **37**(3), 221–232.

Smith III, J. O., Viewpoints on the history of digital synthesis. *In Proceedings of the International Computer Music Conference, ICMC*. 1991.

Strong, W. and **M. Clark** (1967). Synthesis of wind-instrument tones. *The Journal of the Acoustical Society of America*, **41**(1), 39–52.

Subramanian, S. K. (2013). *Modelling gamakas of Carnatic music as a synthesizer for sparse prescriptive notation*. Ph.D. thesis, Ph. D. thesis, National University of Singapore, Singapore.

Suyun, F. and **Y. Yibiao**, Improve music synthesis quality by particular harmonic spectrum interpolation based on sinusoidal model. *In 2016 IEEE 13th International Conference on Signal Processing (ICSP)*. IEEE, 2016.

Swift, G. N. (1990). South Indian Gamaka and the Violin. *Asian music*, **21**(2), 71–89.

Välimäki, V., **R. Hänninen**, and **M. Karjalainen**, An improved digital waveguide model of flute-implementation issues. *In Proceedings of the International Computer Music Conference*. Citeseer, 1996.

Valimaki, V., **M. Karjalainen**, **Z. Jánosy**, and **U. K. Laine**, A real-time dsp implementation of a flute model. *In Acoustics, Speech, and Signal Processing, IEEE International Conference on*, volume 2. IEEE Computer Society, 1992.

Välimäki, V., **M. Karjalainen**, and **T. I. Laakso**, Modeling of woodwind bores with finger holes. *In International Computer Music Conference, ICMC 1993*. 1993.

Varadharajan, J., **G. Sridharan**, **V. Natarajan**, and **R. Sridhar**, Automatic Synthesis of Notes Based on Carnatic Music Raga Characteristics. *In Advanced Computing, Networking and Informatics-Volume 1*. Springer, 2014, 145–152.

Verma, T. S. and **T. H. Meng** (2000). Extending spectral modeling synthesis with transient modeling synthesis. *Computer Music Journal*, **24**(2), 47–59.

Viraraghavan, V. S., **R. Aravind**, and **H. A. Murthy**, A Statistical Analysis of Gamakas in Carnatic Music. *In Proceedings of the 19th ISMIR Conference*. 2017.

Viswanathan, T. (1977). The Analysis of Rāga ālāpana in South Indian Music. *Asian Music*, 13–71.

Vyas, H. M., **S. M. Suma**, **S. G. Koolagudi**, and **K. R. Guruprasad**, Identifying gamakas in carnatic music. *In Contemporary Computing (IC3), 2015 Eighth International Conference on*. IEEE, 2015.

Ystad, S. and **T. Voinier** (2001). A virtually real flute. *Computer Music Journal*, **25**(2), 13–24.

Appendix A
Cubic Spline Interpolation

Cubic spline interpolation makes use of a set of third degree polynomials to interpolate a set of data points such that the entire fit, its first derivative and second derivative are continuous at every point. If there are $n+1$ data points represented by $\{(t_1, x_1), (t_2, x_2), ..., (t_{n+1}, x_{n+1})\}$, we can fit n piecewise cubic polynomials between them. Each polynomial $p_i(t)$ fit between the points (t_i, x_i) and (t_{i+1}, x_{i+1}) can be expressed as

$$p_i(t) = a_i t^3 + b_i t^2 + c_i t + d_i, \tag{A.1}$$

where, $t_i < t \leq t_{i+1}$, and $i = 1, 2, ...\ n$. Imposing the conditions of continuity, we obtain the following set of equations for each cubic spline fit between the points (t_i, x_i) and (t_{i+1}, x_{i+1}).

$$p_i(t_i) = x_i \qquad i = 1, 2, ...\ n. \tag{A.2}$$

$$p_i(t_{i+1}) = x_{i+1} \qquad i = 1, 2, ...\ n. \tag{A.3}$$

$$p'_i(t_{i+1}) = p'_{i+1}(t_{i+1}) \qquad i = 1, 2, ...\ n-1. \tag{A.4}$$

$$p''_i(t_{i+1}) = p''_{i+1}(t_{i+1}) \qquad i = 1, 2, ...\ n-1. \tag{A.5}$$

Thus, we obtain a total of $4n-2$ equations. Since we have $4n$ unknown variables, we impose two more conditions to find their values. We set the slopes of the function at the beginning and end points to zero. The new equations are given by

$$p'_1(t_1) = 0 \tag{A.6}$$

$$p'_n(t_{n+1}) = 0. \tag{A.7}$$

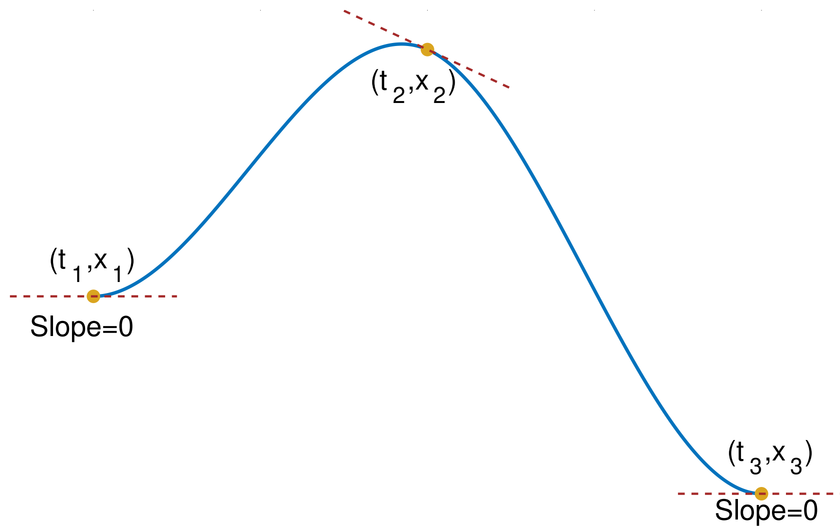


Figure A.1: Example of two cubic splines fit through three points

By solving these $4n$ equations, we get all the coefficients for all the individual polynomials. For example, Fig. A.1 shows two cubic splines fit through three points, namely, $(t_1, x_1), (t_2, x_2)$ and (t_3, x_3). In this case, the set of equations can be written as

$$p_1(t_1) = x_1 \tag{A.8}$$
$$p_1(t_2) = x_2 \tag{A.9}$$
$$p_2(t_2) = x_2 \tag{A.10}$$
$$p_2(t_3) = x_3 \tag{A.11}$$
$$p_1'(t_2) = p_2'(t_2) \tag{A.12}$$
$$p_1''(t_2) = p_2''(t_2) \tag{A.13}$$
$$p_1'(t_1) = 0 \tag{A.14}$$
$$p_1'(t_3) = 0 \tag{A.15}$$

Once the coefficients for the cubic splines for all the representative shapes are found out, time modification can be easily achieved by changing the dependent variable t. For example, if the support of a particular function is to be doubled, the time t is replaced by $\frac{t}{2}$, and if the support need to be halved, the time t is replaced by $2t$. In our application, since the frequency contours are continuous in time, phase contours need to be found out by integration. Since the integral of a cubic polynomial can be found out in closed form, finding the phase from the cubic spline interpolated frequency contour is easy.

Publications based on the thesis

Refereed International Journals/Conferences

1. **M. Ragesh Rajan**, D. Vijayasenan and A. Vijayakumar, "Predicting Gamakas – The Essential Embellishments in Karnatic Music," in IEEE Access, vol. 7, pp. 175386-175395, 2019, doi: 10.1109/ACCESS.2019.2957236. **(Published)**

2. **M. Ragesh Rajan**, A. Vijayakumar, D. Vijayasenan, "Prediction of Aesthetic Elements in Karnatic Music: A Machine Learning Approach", in Proc. Interspeech 2018, pp.2042-2046, DOI:10.21437/Interspeech.2018-991. **(Published)**

3. S. B. Kalluri, D. Vijayasenan, S. Ganapathy, **M. Ragesh Rajan** and P. Krishnan, "NISP: A Multi-lingual Multi-accent Dataset for Speaker Profiling," IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), 2021, pp. 6953-6957, doi: 10.1109/ICASSP39728.2021.9414349. **(Published)**

4. **M. Ragesh Rajan**, "Singing Voice Synthesis System for Carnatic Music." In 2018 5th International Conference on Signal Processing and Integrated Networks (SPIN), pp. 831-835. IEEE, 2018. **(Published)**

5. **M. Ragesh Rajan** and D. Vijayasenan, "A Continuous Time Model for Karnatic Flute Music Synthesis," in Circuits, Systems, and Signal Processing. **(Under Review)**

Bio-data (FOR PHD)

Name	: RAGESH RAJAN M
Address	: ANAT HOUSE WEST OF ARAKULAM, P.O KODUNGALLUR, THRISSUR, KERALA.
Email	: mrageshrajan@gmail.com
Qualification	: M.Tech in Communication Engineering and Signal Processing (2012), from Govt. Engineering College, Thrissur, Kerala, with CGPA **7.84**.