
Universitat
Pompeu Fabra
*Barcelona*

# Connecting Time and Timbre:
## Computational Methods for Generative Rhythmic Loops in Symbolic and Signal Domains

**Cárthach Ó Nuanáin**

TESI DOCTORAL UPF / 2017

Thesis Director:

---

Dr. Sergi Jordà
Music Technology Group
Dept. of Information and Communication Technologies
Universitat Pompeu Fabra, Barcelona, Spain

Dissertation submitted to the Department of Information and Communication Technologies of Universitat Pompeu Fabra in partial fulfillment of the requirements for the degree of

DOCTOR PER LA UNIVERSITAT POMPEU FABRA




Music Technology Group (http://mtg.upf.edu), Department of Information and Communication Technologies (http://www.upf.edu/dtic), Universitat Pompeu Fabra (http://www.upf.edu), Barcelona, Spain.

*Do mo mháthair, Marian.*

This thesis was conducted carried out at the Music Technology Group (MTG) of Universitat Pompeu Fabra in Barcelona, Spain, from Oct. 2013 to Nov. 2017. It was supervised by Dr. Sergi Jordà and Mr. Perfecto Herrera.

Work in several parts of this thesis was carried out in collaboration with the GiantSteps team at the Music Technology Group in UPF as well as other members of the project consortium.

Our work has been gratefully supported by the Department of Information and Communication Technologies (DTIC) PhD fellowship (2013-17), Universitat Pompeu Fabra, and the European Research Council under the European Union's Seventh Framework Program, as part of the GiantSteps project ((FP7-ICT-2013-10 Grant agreement no. 610591).

# Acknowledgments

First and foremost obviously I wish to thank my advisors and mentors Sergi Jordà and Perfecto Herrera. Thanks to Sergi for meeting me in Belfast many moons ago and bringing me to Barcelona. I thank him for his constant commitment, dedication and above all, inspiration. I am equally thankful to Perfecto Herrera for his vast knowledge, infinite patience and unwavering support.

To the most gigantic steppers of them all, the musketeers: Ángel, Dani and Martin, one for all and all for one.

Thanks to Xavier, Cristina, Sonia, Lydia and the rest of UPF for making it a great environment for this lowly scholar.

Shout out to all the MTG gang: Juanjo (particularly for the translations, squawk), Marius (expert courier), Alastair, Sergio, Frederick, Dimi, Oriol, Moha, Rafa, Sebas, Carles, Georgi. Dara, Gerard, Álvaro, Pano, Giuseppe, Martí etc. I am missing others I know but know that it is not intentional.

Cheers to the Barcelona Laptop Orchestra for their collaboration.

Thanks to the Barcelona crew for keeping me (in)sane these past years, and the Bonitos for keeping the mandolin strings from rusting.

Thanks to my Cork-based friends and my Dublin-based friends.

I am indebted to Peter for reviewing some drafts.

Faoi dheireadh caithfidh me an buíochas is mó a thabhairt do mo chlann, as ocht an grá agus an tachaíocht a thug sibh dom i rith an bhliain dorcha, deachair seo a bhí againn go léir. Go raibh maith agat James & Batchimeg, Colm & Melanie 'gus fáilte go dtí an domhan Marló beagaín! Míle buíochas freisin go dtí muintir McCarthy agus Ó Nuanáin.

Go raibh maith agat a thuismitheoirí, ar dtús báire as ocht thabhairt dom an cheoil, mar ní bheadh mé abálta é seo a dhéanamh gan é. Go raibh maith agat, Joe as gach rud, ach go háirithe an cabhair, an bia, na deochanna, agus an Béarla!

Agus le sin, go raibh míle maith agat Marian, tá brón orm níl tú in ann seo a léamh a chroí.

# Abstract

The practice of music composition often stems from a small idea or motif that blossoms into a complete work through careful application of repetition, variation and a dash of inspiration. In the highly rhythmic and repetitive strain of music that is electronic dance, this motif is the fundamental building block known as the *loop*, and is accorded greater importance than any other style. While other genres can depend upon long-established rules governing things like harmonic structure and form, many dance tracks do not stray far from its fundamental central theme in the traditional sense. Rather, the craft of this unique music is revealed through complex, layering arrangements of timbre and intensity arising from the liberal application of rhythmic activity and sonic effects.

This dissertation explores computational methods for generating and varying that rhythmic activity that is so pivotal in composing effective loops. We begin the journey in the symbolic domain, and draw upon a wealth of historical methods for algorithmic composition coupled with the state of the art in rhythmic similarity perception to build *GenDrum*, an intelligent drum machine using genetic algorithms. Our listener survey reveals the validity of the approach, but we question the general adroitness of purely symbolic means in capturing the acute essence of timbre.

Modern approaches to composing electronic music are distinguished by the liberal use of sampling and appropriation of existing sounds as well as the design of purely synthesised new sounds. Concatenative synthesis applies high-level rules and criteria that seek to combine phrases of sounds together in a more intelligent and systematic manner. It is a content-based approach that uses music information retrieval research to work directly with audio and its latent encoding of multidimensional spectral and timbral character.

In the second contribution of the thesis we examine the relevance and application of concatenative synthesis in the specific context of electronic dance music production. We present a comprehensive review of key works in the area of concatenative synthesis and summarise the algorithmic underpinnings that tend to drive these systems. *PyConcat*, a research oriented synthesis framework, is offered to encapsulate many of the key approaches along with some novel improvements to the field and state of the art in the area of timbral feature choice as well as unit selection with Hidden Markov Models.

But, above all, the concern remains with the implications of designing concatenative systems that consider the needs of the *user* and the place of such systems in their existing composition and production workflows. The final contribution of the thesis delivers *RhythmCAT*, a visually appealing virtual instrument plugin with a unique in-

teraction metaphor for exploring concatenative synthesis in a practical, user-friendly manner. This focus on the user implications of concatenative synthesis leads naturally to the question of evaluation: not only in our system but also the wider ecosystem of generative and creative computational agents. Another evaluation methodology is proposed that expands on the one conducted in the symbolic domain and the results are presented and discussed in detail. The thesis concludes with our critical impressions of the work presented and the possible directions for future study.

# Resum

La pràctica de la composició musical part sovint d'una petita idea o motiu que floreix en una obra completa mitjançant una acurada aplicació de la repetició, variació i una mica d'inspiració. En música electrònica de ball (que és altament repetitiva i basada en el ritme), aquest motiu conegut com *loop* és el seu component fonamental, i adquireix una importància més gran que en qualsevol altre estil. Mentre altres gèneres poden dependre de regles establertes per regir l'estructura harmònica i la forma, els temes de música electrònica de ball no s'allunyen molt del seu motiu central fonamental en un sentit tradicional. Aquesta particular música s'elabora a través de complexos arranjaments de capes tímbriques i d'intensitat que sorgeixen a partir d'una aplicació lliure de l'activitat rítmica i dels efectes de so.

Aquesta tesi explora mètodes computacionals per a la generació i variació de l'activitat rítmica, que és un peça clau en la composició de *loops* efectius. Comencem en el domini simbòlic, basant-nos en múltiples mètodes clàssics de composició algorítmica, i en el coneixement més avançat sobre percepció de similitud rítmica per a construir *GenDrum*, una caixa de ritmes intel·ligent basada en algoritmes genètics. Una avaluació amb oients demostra que és un enfocament vàlid, però ens qüestionem l'habilitat de mètodes purament simbòlics per a capturar l'essència del timbre.

Els mètodes més moderns de composició de música electrònica es distingeixen per un ús liberal del samplejat i l'apropiació de sons preexistents, així com del disseny de nous sons purament sintetitzats. La síntesi concatenada aplica regles d'alt nivell i criteris que intenten combinar frases de sons d'una manera més intel·ligent i sistemàtica. Aquest enfocament basat en contingut utilitza la recerca en recuperació d'informació musical per treballar directament amb àudio i amb una codificació latent i multidimensional de l'espectre i de les característiques tímbriques.

Com a segona contribució d'aquesta tesi, examinem la rellevància i aplicació de la síntesi concatenada per a producció de música electrònica de ball. Presentem una revisió exhaustiva de treballs clau en l'àrea de la síntesi concatenada i resumim els fonaments algorítmics en què se solen basar aquests sistemes. *PyConcat*, un entorn per a síntesi orientat a la recerca, encapsula molts dels mètodes més importants juntament amb algunes millores per a la selecció de característiques tímbriques, i la selecció d'unitats amb Models Ocults de Markov.

Però, especialment, segueixen sent clau les implicacions en considerar les necessitats dels *usuaris* en el disseny de sistemes concatenats, i el lloc que aquests ocupen en els seus processos compositius i de producció. La contribució final de la tesi és *RhythmCAT*, un instrument virtual visualment atractiu i amb una metàfora d'interacció única per a l'exploració de la síntesi concatenada d'una manera pràctica i fàcil d'utilitzar.

Aquest èmfasi en les implicacions de l'usuari de la síntesi concatenada porta naturalment a la qüestió de l'avaluació, tant en el nostre sistema com en l'ampli ecosistema d'agents computacionals generatius i creatius. Proposem una altra metodologia d'avaluació que amplia la realitzada en el domini simbòlic, i discutim els resultats en detall. Finalment, presentem les conclusions sobre la nostra feina i possibles direccions per a treballs futurs.

(*Translated from English by Juanjo Bosch*)

# Resumen

La práctica de la composición musical parte a menudo de una pequeña idea o motivo que florece en una obra completa mediante una cuidadosa aplicación de la repetición, variación y una pizca de inspiración. En música electrónica de baile (que es altamente repetitiva y basada en el ritmo), dicho motivo, conocido como *loop* es su componente fundamental, y adquiere una importancia mayor que en cualquier otro estilo. Mientras otros géneros pueden depender de reglas establecidas para regir la estructura armónica y la forma, las piezas de música electrónica de baile no se alejan mucho de su motivo central fundamental en un sentido tradicional. Esta característica música se elabora a través de complejos arreglos de capas de timbres e intensidad que surgen a partir de una aplicación libre de la actividad rítmica y de efectos de sonido.

Esta tesis explora métodos computacionales para la generación y variación de la actividad rítmica, que es un pieza clave en la composición de *loops* efectivos. Comenzamos en el dominio simbólico, basándonos en múltiples métodos clásicos de composición algorítmica y en el conocimiento más avanzado sobre percepción de similitud rítmica, para construir *GenDrum*, una caja de ritmos inteligente basada en algoritmos genéticos. Una evaluación con oyentes demuestra que es un enfoque válido, pero nos cuestionamos la habilidad de métodos puramente simbólicos para capturar la esencia del timbre.

Los métodos más modernos de composición de música electrónica se distinguen por un uso liberal del sampleado y de la apropiación de sonidos preexistentes, así como del diseño de nuevos sonidos puramente sintetizados. La síntesis concatenativa aplica reglas de alto nivel y criterios que intentan combinar frases de sonidos de una manera más inteligente y sistemática. Este enfoque basado en contenido utiliza la investigación en recuperación de información musical para trabajar directamente con audio y con una codificación latente y multidimensional del espectro y de las características tímbricos.

Como segunda contribución de esta tesis, examinamos la relevancia y aplicación de la síntesis concatenativa para producción de música electrónica de baile. Presentamos una revisión exhaustiva de trabajos clave en el área de la síntesis concatenativa y resumimos los fundamentos algorítmicos en los que se suelen basar estos sistemas. *PyConcat*, un entorno para síntesis orientado a la investigación, encapsula muchos de los métodos más importantes junto con algunas mejoras para la selección de características tímbricas, y para la selección de unidades con Modelos Ocultos de Markov.

Pero, especialmente, siguen siendo clave las implicaciones al considerar las necesidades de los *usuarios* en el diseño de sistemas concatenativos y el lugar que estos ocupan en sus procesos compositivos y de producción. La contribución final de la

tesis es *RhythmCAT*, un instrumento virtual visualmente atractivo y con una metáfora de interacción única para la exploración de la síntesis concatenativa de una manera práctica y fácil de usar. Este énfasis en las implicaciones del usuario de la síntesis concatenativa lleva naturalmente a la cuestión de la evaluación, tanto en nuestro sistema como en el amplio ecosistema de agentes computacionales generativos y creativos. Proponemos otra metodología de evaluación que amplía la realizada en el dominio simbólico, y discutimos los resultados en detalle. Finalmente, presentamos las conclusiones sobre nuestro trabajo y posibles direcciones para trabajos futuros.

(*Translated from English by Juanjo Bosch*)

# Contents

# List of Symbols

The following is a list of different symbols used in the dissertation along with a short description of each symbol.

| Symbol | Description |
|---|---|
| $i, j, k, n$ | Indices |
| $N, K$ | Length of discrete signal or size of set |
| $A, B$ | Sets representing two binary string representations of rhythm patterns for comparison, or a transition matrix and emission matrix respectively in a HMM |
| $a_i, b_j$ | Element (or onset) $i$ and $j$ |
| $f$ | General fitness of a genetic algorithm |
| $\lambda$ | A specification for a HMM |
| $\pi$ | Initial probability distribution of a HMM |
| $O$ | A set of observation symbols in a HMM |
| $S$ | A set of hidden symbols in a HMM |

# List of Figures

# List of Tables

# Chapter 1
# Introduction

> "*What is music but organised noises?*"
>
> (Varèse & Wen-chung, 1966)

If music is indeed, as Varèse suggested, to be distilled summarily to "organised noises" in time, then surely the art of music composition has amongst its most effective tenets the dialectical balance between repetition and variation. For a musical motif, idea or theme to be memorable, the composer must use repetition to exploit the human's application of memory in establishing patterns and context between sequences of notes, or so maintains Levitin:

> "*Repetition, when done skillfully by a master composer, is emotionally satisfying to our brains, and makes the listening experience as pleasurable as it is.*" (Levitin, 2007)

But to dispel boredom and make it really interesting and engaging it unquestionably needs to be varied; the composer must carefully choose which musical parameters are coaxed from their centres to confound expectation and break new ground within the oeuvre. Even as the unabashed antagoniser of tradition, Pierre Boulez, concedes:

> "*...a high level of interest in repetition and variation (analogy and difference, recognition and the unknown) is characteristic of all musicians.*"
>
> (as cited by Campbell (1997))

For common practice period Western Art music, repetition is used to turn melodic and rhythmic fragments into concrete themes, but repetition of course is also dominantly utilised in macrostructural arrangement of form, most notably in the ternary-based

sonata - where thematic material is introduced, developed and recapitulated in a systematic manner that aids listeners navigate dense and complex works (Benward & Saker, 2008).

Later, African-American roots music such as blues and R&B would centre its aesthetic on bite-sized, guitar-driven ostinatos known as "riffs", subsequently sowing the seeds for the explosion of rock and pop culture (Middleton, 2009; Hatch & Millward, 1987). Minimalist composers such as Steve Reich, Philip Glass, LaMonte Young and Terry Reilly (Reich & Hillier, 2011; Nyman, 1999) would push the mantra of "doing a lot with a little" through exaggerated repetition to its extremes[1], mirroring prior movements in literature (e.g. Beckett) and the visual arts. Minimalism, in its essence, tips the weight of repetition and variation; repetition is so deeply ingrained and rife in the process[2] that any slight change or variation introduced is overwhelmingly noticeable and immensely consequential.

The stylistic intentions of this thesis however, are largely concerned with Electronic Dance Music - a subset of electronic music first appearing roughly in the 1980s intended initially for dancing at events like raves or nightclubs (McLeod, 2001; Butler, 2006). In dance music,[3] small, repeated patterns known as "loops" provide the core building blocks for compositions (Neill, 2002; Patricio & Dittmar, 2016). A single idea, perhaps consisting of a drum pattern, a keyboard vamp or often a recontextualised, “borrowed” audio sample - as a result of careful listening to myriad musical or sonic sources - can become the seed for an entire track[4], as Neill observes:

> “*Just as composers in earlier historical periods often worked within a given set of large-scale formal parameters (sonata form, dance forms, tone poems etc.), innovative pop electronic composers use steady pulse, loop-based structures and 4/4 time as a vehicle for a wide range of compositional ideas and innovations*” (Neill, 2002)

Electronic dance music is ultimately a product of the technology on which it is created during a nascent time span, a music born out of the limitations of the tools available. Initially these tools comprised synthesisers, samplers and drum machines, but as the digital computer became viable for consumers, they increasingly involved sophisticated software that virtualises modern recording studio capability.

So with a music so seemingly mechanical and so intrinsically coupled with the machine, naturally it is pertinent to ask: are the machines themselves capable of fulfilling

[1] Reich, like his predecessor Varèse gained notoriety for instigating "the last great musical scandal of the 20th century" (Ross, 2007). During the concert premiere of his seminal Four Organs an audience member was purported to have ran down the aisles, screaming “All right, I confess!”.

[2] Nyman and Reich both use the term “process music” in distinguishing their work as well as that of Cage from works by post-war serialists and perhaps more maximalists such as Xenakis and Stockhausen

[3] Nomenclature will be dealt with in the following chapter

[4] We shall see in Chapter 2 - in the case of “drum & bass” - how a particular motif can also spawn an entire subgenre within dance music.

the role of the composer? Or to put it in other words, rather than the composer choosing and organising the sounds that the systems generate, can we go one step further and call on the machine to choose and organise sounds also? Of course what we are describing here is algorithmic (Roads, 1996), or generative music (Collins, 2008) - where computer programs are used to compose music - a topic that has fascinated researchers, composers and musicologists alike for over half a century (Burns & Helen, 1994; Fernández & Vico, 2013), undoubtedly preceding the global rise of dance music culture. Successful automatic algorithmic composition of works are heavily dependent on the intended style and the algorithmic techniques chosen. Building on a wealth of knowledge historically available in matters of harmonic counterpoint and voice leading, Cope (1991), for example, has for some time now created convincing syntheses of Bach chorales, but acceptable attempts at modelling modern pop music are only recently beginning to emerge (Ghedini et al., 2015; Deltorn, 2017; Pachet et al., 2017)[5]. Attempting to capture the entire gamut of creative musical endeavour with computational means seems a lofty, and for now, rather intractable prospect, but what about processes that offer the composer a helping hand?

Thus, in this thesis I examine computational, algorithmic or otherwise generative approaches that assist the composer in creating rhythm-centric loops suitable for dance music production. I emphasise *assist* to contrast with traditional algorithmic compositional systems that may focus on producing complete works or output with little or no interaction or intervention from the user other than setting some initial parameters. As Pasquier explains when introducing what he terms "musical metacreation":

> "*Such systems will tend to produce – ideally musically successful – variations of the same composition rather than creating completely novel outputs with each run.*" (Pasquier et al., 2017)

Like Pasquier, I emphasise *assist* to describe situations where the composer has an idea (i.e. a loop) they want to expand and develop through intelligent and systematic repetition and variation, while maintaining the impression that what is generated remains truly something they can call their own, rather than any *deus ex machina*.

My process is an iterative one, perhaps, echoing the evolution of dance music production practices, which has itself moved from primitive symbolic control of hardware devices to more signal focussed methods that manipulate and process sampled sound. I draw from knowledge in the literature, propose advancements where they are lacking, build prototypes, then most crucially, evaluate the work with other active users.

Initially I describe how symbolic methods, coupled with perceptual knowledge of similarity between rhythm representations, can be combined to create a perceptually motivated generative drum machine that seeks to create convincing patterns. Following a user evaluation of the system, I highlight some limitations we see in purely

[5] http://www.flow-machines.com/tag/flow-machines-press/

symbolic domains. These limitations I feel justify a shift to more content-based techniques such as concatenative synthesis, in order to build systems that can harness state of the art research in machine listening and the thriving field of music information retrieval (MIR) for repurposing existing audio in a logical and efficient manner. We describe concatenative synthesis from a research perspective, and offer some novel improvements for addressing some shortcomings in Hidden Markov Model (HMM) driven approaches, for instance. Subsequently, a system is proposed from the user perspective, demonstrating clearly how to distil the state of the art techniques typically found in research oriented concepts, to deliver a tool that is visually and creatively appealing, compelling and viable for the modern music producer.

For now though, the rest of the introduction will continue with a definition of my motivations and the wider context of my work within the GiantSteps project. The introductory chapter concludes with a detailed overview of the contents of each subsequent chapter in the dissertation.

## 1.1 Motivations and The GiantSteps Project

The GiantSteps project is a European Union led initiative launched in 2013 with the intention of creating the "Seven-League Boots" for music production in the next decade and beyond (Knees et al., 2016). Three main goals of the project embrace:

1. Developing and integrating musical expert agents, supportive and inspirational systems for melody, harmony, rhythm, structure or style, based on symbolic, audio and metadata analysis.

2. Developing improved interfaces and paradigms for musical human-computer interaction and collaborative control of multi-dimensional parameter spaces using novel visualisation techniques.

3. Addressing low cost portable devices by developing low-complexity algorithms for music analysis and recommendation tailored to their capabilities.

The research consortium consisted of multidisciplinary bodies drawing from academia, industry and the arts, including Universitat Pompeu Fabra, Johannes Kepler University, STEIM, Reactable Systems and Native Instruments. One of the primary successes of the project has been the constant focus on user analysis, that has been pivotal in helping guide and shape the advancement of the state of the art. Through active engagement at workshops, conferences and festivals worldwide such as Red Bull Music Academy, Sónar, Music Hack Day, considerable effort has been devoted to study DJs', producers' and composers' needs, desires, and skills; investigating their processes and mental representations of tasks and tools; and evaluating their responses to prototypes that consolidate research into practical realisations.

## 1.2 Thesis Outline

Before delving into the science, Chapter 2 offers a critical introduction to electronic dance music from a cultural and musicological perspective. Dance music is a relatively recent phenomenon compared its precursory roots in electroacoustic and contemporary composition, already the subject of many key texts as well as devoted gatherings for its academic discourse. Furthermore the context for computational and musicological analysis for earlier eras of Art music as well as popular styles like rock and jazz are historically well-established. We therefore offer the case for *why* electronic dance music is interesting to study, how its various subgenre strands have developed and its compositional traits which shall hopefully inform our analysis and enventual synthesis using computational means.

In Chapter 3 we will begin our journey by introducing algorithmic composition or generative music as that corner of computer music concerned with using computational resources and programmatic techniques to derive aesthetically pleasing music. As we delve further into this, we reveal how natural aspects of music such as rhythm can be recreated using symbolic representational structures informed by the study of how we perceive such facets when they are performed or composed by humans. We propose a system that can recreate such rhythms automatically, embedding our interpretation of this knowledge within an interactive genetic algorithm aimed at fulfilling the needs of electronic dance music producers. We pay close attention to how one can evaluate such systems from a user perspective. Based on this user evaluation, we give a frank appraisal of our own personal impressions of the system and explain how this motivates a change of direction and focus.

Chapter 4 provides a contextual basis for our shift of focus by introducing state of the art techniques in art and in academic literature that demonstrates the aesthetic reuse of content. By content we mean existing objects that exist in the wild, that can be exploited, shifted and reused, malleable for repurposing in creating new work. In terms of electronic music, we describe the particular practice of sampling - whereby existing sounds and music are directly adapted and recontextualised in the aforementioned manner - tracing its historical trajectory from the early days of tape experiments through to modern dance music and contemporary efforts. As is most pertinent to the ambitions of this thesis, we are expressly concerned with those works that seek to automate or aid the composer to perform sampling in some systematic or hopefully intelligent manner using computer resources. This leads us naturally to concentrate on granular synthesis initially, with the second portion of the chapter extensively reviewing state of the art in concatenative synthesis.

With this context in place, Chapter 5 delves in depth into the practical aspects of crafting and experimenting with concatenative systems. Building on the state of the art methods introduced in Chapter 4, theoretical, algorithmic and mathematical mechanisms are detailed. In the area of unit selection - one of core tasks in concatenative synthesis - some shortcomings are highlighted concerning the application of Hidden

Markov Model (HMM)s, so some novel techniques are proposed to overcome them. The PyConcat library is delivered as a culmination of our efforts in this course of study, intended as a research focussed, easy-to-use framework for exploring concatenative synthesis' fundamental concepts.

In Chapter 6 we return to one of the other goals of our thesis: adapting state of the art research from academia to deliver new musical software that considers the needs of our user (rather than the researcher). To this end, the RhythmCAT system is presented, an easy to use virtual software instrument that encapsulates the sampling capabilities of concatenative synthesis along with some new visual and interaction paradigms that seeks to enhance the creativity of the modern electronic dance music producer and composer.

As purported in Chapter 7, the systematic evaluation of computer music platforms is an oft-neglected activity, especially in academic efforts where the designer-researcher might also be the sole user. As the work in Chapter 6 is so user-focussed, there needs to be a well thought out framework in place to validate it for its efficacy and efficiency in addressing the needs of those users. This chapter is devoted to devising such a framework, that we hope delivers a substantial contribution to the music software making community working towards similar goals.

To conclude the thesis, Chapter 8 reflects on the overall scope and findings of the research delivered. We summarise the achievements, highlight some criticisms and lay a clear path for continuing with future work.

# Chapter 2

# A Musicological Overview of Electronic Dance Music

Electronic Dance Music (EDM) is a highly repetitive and rhythm-centric genre based on the liberal use of looping motifs consisting of layers of sampled and synthesised sound sources. Obviously, it is a style that is primarily intended for dance, at clubs, organised but illegal gatherings known as "raves" and large festivals but, as we will see, as the movement has evolved, that distinction has become quite blurred. In terms of nomenclature, it is often referred to as dance music or simply "dance" by its proponents (and opponents) in anglophonic Europe. Its abbreviation, EDM, as Reynolds - a music culture author and researcher who has been reporting on dance culture for many years - notes, seems to be a North American labelling that has been applied to a marked resurgence in dance music over the past decade primarily in the United States[6]. For the purposes of this thesis I will use the term dance music, as it is what I have always called it. Should there be ambiguity between electronic dance music and say, dance forms from Western Art music or folk, I will make the necessary distinctions clear.

Dance music is made up of a bewildering taxonomy of deeply branched sub-genres that are defended rigorously by their disciples with an almost religious fervour. A glance at Ishkur's excellent online resource "Ishkur's Guide to Electronic Music" [7] (Figure 2.1) reveals 7 high level strands that include "House", "Trance", "Techno", "Breakbeat", "Jungle", "Hardcore" and "Downtempo"[8]. Within House, for example, we can see a complicated network of arches tracing its genesis from other non-electronic dance music specific genres in the 1970s (e.g. disco) to its many different modern incarnations.

[6] https://www.theguardian.com/music/2012/aug/02/how-rave-music-conquered-america

[7] http://techno.org/electronic-music-guide/

[8] A more formal catalogue is presented by McLeod (2001), but Ishkur's guide is infamous enough to be referenced by many other articles in serious literature (Leimeister et al., 2014; Pampalk et al., 2005).

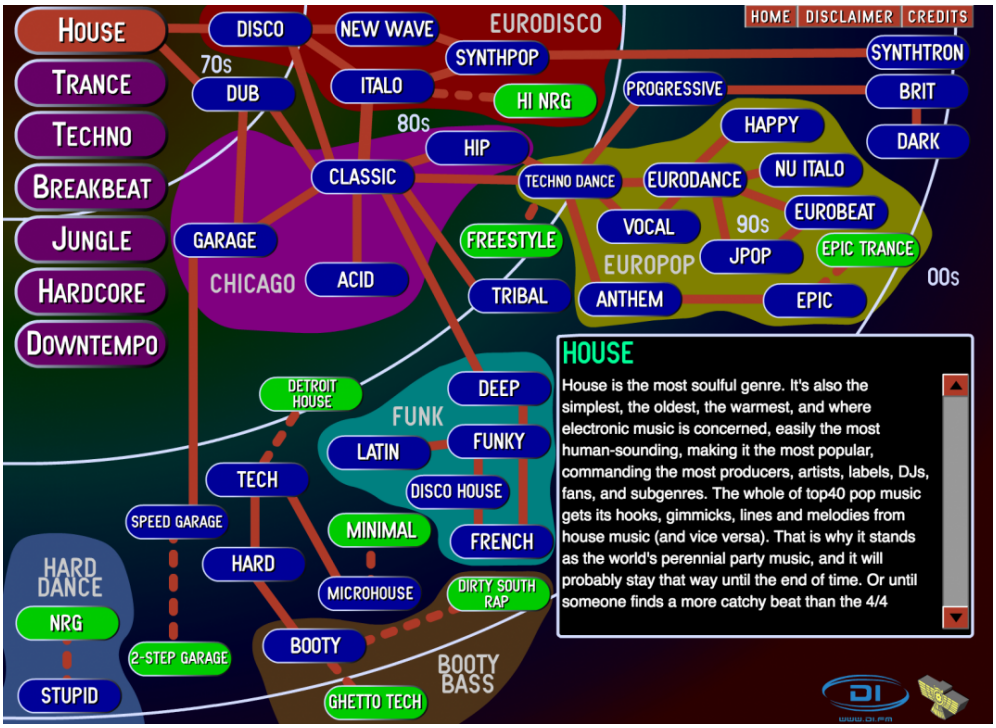



**Figure 2.1:** Ishkur's Guide to Electronic Music

Three of these top level sub-genres are what we would deem representative and ambassadorial in describing dance music, namely techno, house and jungle, and we will expand on each of them now in turn, followed by some remarks on another interesting direction in electronic music that defies easy categorisation.

## 2.1 House

The act of going to a venue purely to dance and listen to recorded music carefully selected by a DJ existed well before the arrival what we now know as electronic dance music, as is clear from the disco era in the 1970s. Clubs such as "Paradise Garage" and "Studio 54" became focal points for the general public as well as artists living in New York, and were revered for their sound systems and resident music selectors. It was at these lengthy sessions that the cult of the DJ would nurture, with one of the most notable emerging being DJ Larry Levan. Levan would pioneer many aspects of what is now understood as the classic "DJ Set", and was particularly celebrated for his ability to "read the crowd" - to adapt and change his sequencing and mixing depending on the mood of the audience of a given night. He was instrumental in developing the practice of remixing music, and in later years started to incorporate samplers and drum machines when repurposing older R&B records for his dance


**Figure 2.2:** The Roland Tr-303 (top-left), TR-808 (bottom-left) and TR-909 (bottom-right) (Image from Synth Mania)

floor.

The enduring legacy of disco left the most lasting impression on the city of Chicago, where it enjoyed heavy rotation in its venues long after its demise in other urban centres (Reynolds, 2013). Disco's fading popularity meant that new releases were less forthcoming, inevitably leading DJs in the city scrambling to remix and update existing records à la Larry Levan, in order to stay fresh and current for audiences. One of Levan's disciples, Frankie Knuckles, was a resident at the Warehouse club in Chicago, where his blending of regular disco and the electronic leaning Italo-disco from Italy (Giorgio Moroder would be its most notable export) soon became morphed and intertwined with his own bass lines and drum tracks (Butler, 2006). Eponymously, the name for this style was eventually shortened to become what is now known as "house".

Listening to classic house tracks, it is rooted unequivocally in a "four on the floor" regular 4/4 bass drum pattern, supplanted with synthesised claps and snares on the second and fourth backbeats, indicative of the equipment it was built on: the infamous TR-808 drum machine (Figure 2.4)(Blashill, 2002). Owing to its disco roots, house music is often described as the most "soulful" of dance music styles, in no small part due to syncopated baselines based around the root and the fifths, with short, jazzy electric piano style vamps providing harmonic context for catchy vocal hooks.

Towards the latter half of the 80s, the Chicago house scene gave way to a more stripped back, harder sound known as acid house. A classic example of how a singular misuse of technology can redefine an entire genre, acid house built its entire iden-

tity out of another Roland product, this time the TR-303 bass synthesiser (McLeod, 2001)[9]. The production team "Phuture" discovered that by constantly modulating the frequency cutoff and resonance parameters of the bass pattern, they managed to create a frantic new "squelching" bass sound that both diminished the roles of traditional layering techniques using keys and vocals and emphasised the value of new production tricks, deliberate or accidental (Vitos, 2014).

Acid house was to prove most influential, however, when the first records made their way across the Atlantic to the United Kingdom. In London and Manchester clubs such as Shroom and the Hacienda (the latter owned and managed by the label Factory Records along with its flagship group New Order (Hook, 2009)) heavy rotation of acid house tracks triggered what would eventually become known as the "rave" movement. The rave movement was centred around large, spontaneous but entirely unlicensed gatherings focussed on marathon sessions of dancing to acid house and, much to the chagrin of authorities, consumption of club drugs such as MDMA. Rave culture would reach its peak during 1988-1989, culminating in what is now referred to as the "Second Summer of Love" (Gore, 1997). High profile coverage in the tabloid newspapers caused 'moral panic' (Martin, 1999), and demonised and politicised rave music, so much so that the controversial 1994 Criminal Justice Act included a clause granting police full authority to halt any event involving the "emission of a succession of repetitive beats" (Gilbert, 1997) [10] .

## 2.2 Techno

Around the end of the 1980s a similar musical development to house was burgeoning in the city of Detroit, Michigan. Detroit was already a complicated city with a chequered past; it was here that Motown records was formed, with artists like Marvin Gaye producing some of the most defining soul music of the 1960s and early 1970s. The name Motown, a portmanteau of motor and town, acknowledges Detroit as the "motor city" - once the epicentre for the American car industry in the early part of the the 20th century. Towards the second half of the century this once booming complex collapsed due to increased overseas competition from countries like Japan and national economic decline. Today the graveyard of abandoned factories in downtown Detroit serve as a bitter reminder of these greater years (Sicko, 2010).

The continued de-industrialisation, suburbanisation and the eventual relocation of the legendary soul label Motown Records to Los Angeles in 1972 had a devastating impact on the cultural output of the urban centre. But bubbling under the surface in Detroit's suburbs three middle class African American high school friends, known as

[9] https://youtu.be/TLQwwtjtiY4

[10] Veteran duo Autechre (who could hardly ever be accused of basking in repetitivity) in response released *Anti* EP containing a track "programmed in such a way that no bars contain identical beats", with the warning: "...we advise DJs to have a lawyer and a musicologist present at all times to confirm the non-repetitive nature of the music in the event of police harassment" (Atkinson, 2007)

the "Belleville Three", were planting the seeds for what would become techno. Juan Atkins, Derrick May and Kevin Saunderson were obsessed with sci-fi, technology and futurism, and raised on a diet of George Clinton funk and Italo-disco coupled with frequent visits to Chicago for injections of house, however it is widely acknowledged that their greatest influence would be German electronic group Kraftwerk (Pope, 2011). This was largely due to a seminal local radio show hosted by Charles Johnson a.k.a. "The Electrifying Mojo"[11], mixing artists like Prince with European new wave, italo-disco and krautrock (Reynolds, 2013; Sicko, 2010). But it's no coincidence Atkins, May and Saunderson latched on to Kraftwerk forthright. Kraftwerk's albums revealed a group enamoured with the themes of transport, progress and movement, from the Beach Boys inspired ode to German motorways "Autobahn", to the pan-European romanticism of rail travel on *Trans-Europe Express* (Albiez & Pattie, 2010). This naturally resonated with the Detroit trio's formulating ideas of Afro-futurism, against the backdrop of a city once so intertwined with the machinery of transportation and automobilisation.

Like Chicago house, Detroit techno was constructed and produced using the tools available at the time, inevitably the gamut of Roland's drum machines and bass sequencers, coupled with whatever other synthesisers and sampling equipment was on hand. Tempo markings are also similar than house, most frequently in the 110 to 140 bpm range. So what distinguishes techno from house then? Techno at least, is considered more minimal and mechanical than house, less rooted in the more soulful influences of disco and soul as artist May points out:

> "*House still has its heart in 70s disco; we don't have any of that respect for the past, it's strictly future music. We have a much greater aptitude for experiment.*" (Collins et al., 2013)

There is a driving energy to techno that discriminates it from house tracks, the tonality is often more dissonant and ominous. Kraftwerk style vocoder driven vocals and titles such as "No UFO's" [sic] and "Time Space Transmat" highlight the futurist, utopian themes and aesthetic that makes this era of Detroit techno idiosyncratic.

Techno would go full circle and transport itself back to Germany, where clubs such as Tresor and Berghain in the capital of Berlin soon became shrines for techno lovers from around the world. Berlin takes its techno very seriously. Tresor and Berghain are housed in huge, imposing former power plants[12] and factory warehouses, with the latter recently receiving the same tax status[13] as other more historic cultural centres such as opera and concert halls [14]. These industrial settings have moulded and shaped

[11] http://daily.redbullmusicacademy.com/2015/05/electrifying-mojo-feature

[12] Suffice it to say, Kraftwerk is the German word for "power station"

[13] http://pitchfork.com/news/68211-berghain-declared-high-culture-venue-by-berlin-court/

[14] Berghain's legal representative successfully argued that "concertgoers may achieve a similar intoxicating effect from a Mahler symphony as from a DJ set"

**Figure 2.3:** Berghain Club in Berlin, housed in a former power plant (Image from Resident Advisor)

the sound of techno further, just like acid house did with its softer earlier incarnations. German techno sounds big, heavy and dark. Kick drums are brutal, relentless and reverberated, the reliance is more on metallic, machinery inspired sound design than on any easily discernable pitched, melodic or harmonic material. Nye (2013) has also traced this trajectory of evolution (or perhaps devolution?) and observes a marked 'minimalism' of the sound compared to its trans-Atlantic predecessor.

In a fitting closing observation, it is interesting to note that Kraftwerk, many years later, have acknowledged their influence on techno as well as the cultural duality shared by these two cities seemingly so far apart: on their track "Planet der Visionen" the refrain is *"Detroit/Germany/We're so electric"*.

## 2.3 Jungle and Drum 'n' Bass

As the innocence and hope of rave dream started to disentangle and disintegrate in an increasingly volatile post-Thatcherite Britain, a radically new sound rose up from the ashes. Jungle, hardcore or drum 'n' bass drew its inspiration from two rather disparate influences. Firstly, in New York in the 1970s, DJ Kool Herc realised that by sequencing two copies of the same record, and looping the isolated "break" section - a short passage of drum soloing from funk and R&B records - he could create a 5 minute performance for the purposes of "break" dancing and for an MC (Master of

Ceremonies or Microphone Controller) to rhyme over (Smith, 2000).

As such, Kool Herc essentially devised the underpinnings of a musical framework for emerging hip hop culture and rap music. Hip hop, as opposed to say, house and techno, is *sample-oriented* music, meaning that manipulation of existing recordings of audio are utilised far more than raw signal generating synthesisers. Jungle builds on hip hop's sampling aesthetic and heavy reliance on breaks but takes it to completely new extremes. With the development of sophisticated hardware production tools such as the Roland MPC (Music Production Centre) and, subsequently, the personal computer, endless new possibilities for complex and finely grained sample manipulation was within reach of homegrown bedroom producers.

Jungle's other roots is the direct result of United Kingdom's past as a colonial power. Migration from the Commonwealth Caribbean island of Jamaica[15] to London brought with it reggae, dub music and sound system culture in general. Dub reggae was already a technology-centred form of music way before dance producers began twiddling with Roland gear. Engineers like Lee Scratch Perry would take instrumental versions of reggae tracks and remix them live using a slew of mixing desk tricks and exaggerated drenching of outboard effects (most prominently using the Roland Space Echo), blurring the lines between what constitutes the boundaries of the producer, performer and composer. Dub was performed at dedicated events with massive sound systems and MCs, somewhat prototypically anticipating central aspects of hip hop and rave.

Just as reggae's basic rhythm inverted the traditional downbeat and upbeat pattern of rock, jungle completely upends the traditional symmetry and repetition of house/-techno dance music. Equally, just as the combined rhythm section of the bass guitar and drums (referred to as the *riddim*) takes a more pivotal role in reggae, the drums in jungle and drum and bass are easily the most dominant element. As Simon Reynolds observes:

> "*In jungle, the rhythm is the melody; the drum patterns are as hooky as the vocal samples or keyboard refrains*" (Reynolds, 2013)

Where jungle departs greatly from its spiritual predecessor however is the actual arrangement of rhythmic patterns and its tempo. Reggae songs like chugs along typically around the 80 BPM mark; jungle can easily reach BPMs of almost double that[16]. Drum production in jungle music is based around creating endlessly complex and dizzyingly intricate resequencing and reinterpretations of breakbeats. Many breakbeat samples have been used over the past decades, but the most infamous and enduring undoubtedly remains the "Amen Break", a short 4-6 second drum solo on the

[15] In fact, DJ Kool Herc was born in Kingston, Jamaica, and would have been exposed to dancehall culture before emigrating to the United States at the age of 12 (Chang, 2007).

[16] And wreaking havoc on BPM detectors!

song “Amen Brother" by 1960s R&B group The Winstons (Collins, 2007b). Along with an excerpt from “Funky Drummer" by James Brown, they hold the contentious titles of being the most sampled breaks in music, contentious because the artists in question have received little or no royalty payments for its reproduction.

## 2.4 Warp Records and *Artificial Intelligence*

By the early 1990s electronic dance music was already maturing to a point where it began to look inward at itself in reflection and contemplation. After a long night of hedonism, ravers were not likely to listen to more punishing dancefloor oriented beats but rather sought out a more subdued, laid back sound to match abating moods.

Warp Records, a Sheffield-based techno label anticipated this development and decided to capitalise on the emerging trend in a defining series of compilations known as *Artificial Intelligence*. Looking at the green cover of the first release, it depicts a lone automaton comfortably seated in an armchair, facing a stereo system and a shelf of vinyl records. The intention is clear: this is not music for dancing, and as if the image wasn't enough, the subtitle below proclaims “Electronic Listening Music From Warp”. The *Artificial Intelligence* series would go on to release seven other albums, proving highly influential in the trajectory of dance music and bolstering Warp Records' reputation as a bastion for “groundbreaking music” and, later on, branching into video and cinema. They cleverly managed to usher in an era of album format electronica, thus securing their longevity by doing away with the single release model responsible for the short lifespan of many a prior dance music imprint.

The early music released by Warp Records was swiftly retitled by fans as Intelligent Dance Music (IDM), a catch all label that largely grew out of discourse between members of Usenet forums discussing rave music to describe “music that moves the mind, not just the body” (Alwakeel, 2009). Immediately the term was controversial, even by those who would be considered its most respected innovators. The problem with the IDM label is that it not so much describes a sound, rather than a collective ethos that just simply eschews conventional dancefloor geared sounds and production aesthetic. Richard D. James (Aphex Twin) takes further umbrage at the supposed elitist connotations it conveys:

> “*I just think it's really funny to have terms like that. It's basically saying 'this is intelligent and everything else is stupid.' It's really nasty to everyone else's music. It makes me laugh.*” (Haworth, 2015)

Comparing Warp Records' stalwart roster of Aphex Twin, Squarepusher, Autechre and Boards of Canada under this umbrella term proves how divisive and varied the actual musical output is. Scottish duo Boards of Canada blend an uneasy mix of woozy, tape modulated hip hop beats with nostalgic samples from old VHS cassettes on their classic albums *Music Has The Right To Children* and *Geogaddy*. This is light


**Figure 2.4:** Cover Sleeve for Warp Records' *Artificial Intelligence* (Image from Warp.net)

years removed from, for example, the increasingly cold, angular and austere precision that has characterised most of Autechre's exhaustive back catalogue. Starting out with reasonably symmetric post-techno repetitions and melodic analog synthesis on albums like *Incunabula* and *Tri Repetae*, Autechre's music has become more abstract and hard to pin down over time, owing as much sonically and aesthetically to Greek composer/theorist Iannis Xenakis as to any Detroit pioneer.

Mostly this transformation has arisen out of their shifting and maturing use of technology. Albums like *Confeld* betray a pair of experimentalists enamored with algorithmic composition, particularly using the Max/MSP environment. Regarding their continuing use of interactive software in live contexts they have described[17] how their processes differ from more traditional laptop performances:

> "*I mean, there's no actual music "there"—it's not like we make music, then use the system to replay it in new ways. The system itself is making the music each time, it's all about the capabilities of the system dictating what the music's like*"

Indisputably however, it is Aphex Twin who has become the bona fide poster boy for Warp Records and IDM as a whole. The Cornish producer started out with two double albums exploring ambient techno and beatless ambient music on *Selected Ambient Works 85–92* and *Selected Ambient Works Volume II* respectively. Later albums

[17] https://www.residentadvisor.net/features/2756

delved further into extreme recontextualisation and interpretation of earlier jungle and drum ‘n’ bass - often referred to by its detractors as “drill ‘n’ bass” due to its harshness and intensity - culminating in his most personal and broad record: *Drukqs*. Over two discs, this album leaps from Satie-esque furniture music on “Avril 14th” to musique concrète (“Gwarek2”, “Gwety Mernans”), as well his many brutally complex post jungle workouts on tracks like “Vordhosbn” and “Mt Saint Michel + Saint Michaels Mount”.

Richard D. James and his labelmates, with their supposed “intelligent" take on dance music, managed to win much favour and legitimacy in contemporary and art music circles. The 2006 compilation *Warp Works & Twentieth Century Masters* mixed pieces of Aphex Twin and Squarepusher pieces alongside performances of Ligeti, Varèse and Cage, while London-based chamber ensemble Alarm Will Sound have recorded careful and faithful arrangements of Aphex Twin and Autechre on albums such as *A/Rhythmia*. What’s striking about listening to these experiments is how natural and appropriate they sound compared to many forced and kitsch attempts at orchestrating rock music for instance.[18]

As suggested in the introductory chapter, this has been an attempt to provide a short familiarisation with dance music from cultural, musicological and, in places, sociopolitical perspectives. It was not my intention to be exhaustive nor authoritative; I understand there are countless subgenres of dance music that I have omitted for the sake of brevity and enthusiasts and experts alike may take umbrage with those I’ve chosen to retain and my summaries of them. What I would hope is that the reader gains a flavour of my understanding of dance music and appreciate its scope for our forthcoming study.

[18] Perhaps this is more of a personal aversion - I played in orchestras in my youth and I absolutely loathed those light medley arrangements of Beatles and Beach Boys that sanctified all the dangerous elements of rock ‘n’ roll.

# Chapter 3

# Generative Symbolic Music

## 3.1 Introduction

As long as computers have been available, a curiosity has prevailed as to whether computers are capable of being creative (Cardoso et al., 2009). Indeed the multidisciplinary area known as computational creativity (Boden, 1998; Wiggins, 2006; Boden & Edmonds, 2009; Boden, 2009; Colton & Wiggins, 2012) gathers researchers from artificial intelligence, cognition, philosophy and the arts, in order to investigate this lofty aspiration. In the wider sphere of computer music, algorithmic composition (Jacob, 1996; Fernández & Vico, 2013) endeavours to construct programs[19] that could automatically generate (or help generate) works of music for human listeners. With some composers, the goal is to model style conforming compositions (Cope, 1987, 1991) so convincingly that they pass some interpretation of the Turing test (Ariza, 2009), while, others accept algorithmic composition for what it is - a different type of music, hence accepting all its limitations or possibilities, or as Pease puts it:

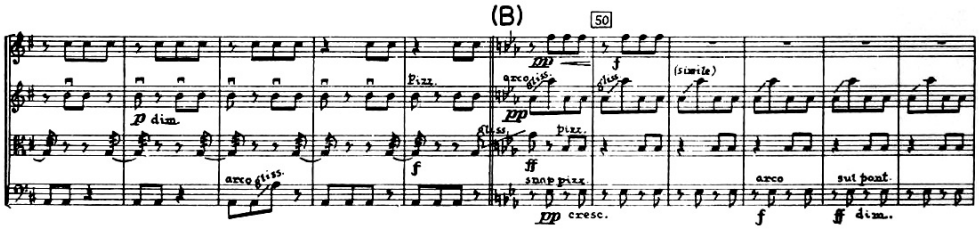



**Figure 3.1:** Excerpt from the Illiac Suite by Lejaren Hiller. Image from Morgan & Legard (2015)

[19] Although most texts introducing algorithmic composition will be at pains to emphasise that algorithmic composition - in the context of some systematic application of rules for composing, like in Mozart's Dice game or in twelve-tone technique and serialism - in fact predates the computer.

> "*Turing Test is largely inappropriate for the purposes of evaluation in Computational Creativity, since it attempts to homogenise creativity into a single (human) style*" (Pease & Colton, 2011)

In the early days of computer music composition, digital synthesis was only in its theoretical infancy, so algorithmic composition experiments typically involved generating some form of symbolic medium for later realisation as something audible. For instance, composers such as Hiller used supercomputers to generate a numerical computer score (Figure 3.1) which was then transcribed manually to staff notation and performed by a live string quartet (Hiller & Isaacson, 1979). Iannis Xenakis, in his treatise *Formalized Music* (Bradshaw & Xenakis, 1973), explains how he harnessed supercomputer capability to generate large stochastic distributions that generated parameters for written compositions also to be realised by live performers. It is interesting to hear how Xenakis calls attention to the role that computer programs play in *assisting* him in his creative practice, rather than holistic generation of entire works[20]:

> "*the computer has not actually produced the resultant sound; it has only aided the composer by virtue of its high-speed computations"* (Cope, 2000)

Later, with the advent of General MIDI, music-oriented programming environments such as Max, Pd and OpenMusic were used to procedurally control note information and parameters of external hardware synthesisers and samplers automatically. One of my prior works (Ó Nuanáin & Sullivan, 2014) explored tangible table-top interfaces modelled on the Reactable (Jordà et al., 2005) for interactive control of real-time compositional algorithms, in a bid to expand the controller space offered by other real-time composition practitioners such as Karlheinz Essl (Essl, 1992). These experiments comprised a collection of Max/MSP patches that sent streams of algorithmically generated MIDI control information to a virtual bank of synthesisers and samplers hosted in a DAW.

## 3.2 Algorithmic Composition Techniques

Programmatic techniques employed in algorithmic composition range from the very rudimentary (simple random number generators) to very sophisticated arrangements of agents borrowing from artificial intelligence. Algorithmic composition systems can be qualified as stochastic (the output is unpredictable), deterministic (the output is predictable) or hybrid, where one or more aspects of the system are determined.

[20] There are obvious parallels to be drawn here with Autechre's statement regarding their live work in Chapter 2.

There exists a wealth of conceptual approaches exist for algorithmic composition that have fallen in and out of flavour over the years. We summarise key ones here that we feel are representative of the current state of the art and pertinent to the arguments put forth within the thesis. For a thorough treatment of historical methods for algorithmic composition we suggest consulting the Computer Music Tutorial (Roads, 1996) or surveys reported by Fernández & Vico (2013) and Papadopoulos & Wiggins (1999).

### 3.2.1 Stochastic Processes

Stochastic processes are largely concerned with strategic applications of probabilistic procedures. When composers defer some aspect of their creative activity to some element of chance they are invoking a stochastic process; whether this be the flip of a coin to choose which section to enter in a large musical structure or using a random number generator to select pitch values on their synthesiser, for example.

#### 3.2.1.1 Probability Distributions

Composers using random processes in a serious way (like Hiller and Xenakis) naturally gravitate towards methods of influencing the outcomes in some systematic manner. They utilise probability table lookups to shape a specified probability distribution geared towards a desired musical elemental goal. Most computer music oriented software environments have some sort of facility for creating such distributions interactively, and in fact Max/MSP provides a visual table object for drawing distributions explicitly. They can be used to create standard mathematical distributions such as linear, bell-shaped, skewed etc., and *The Computer Music Tutorial* (Roads, 1996) gives derivations for many of them in the chapter dealing with algorithm composition. Ames (1990) also gives insight into similar statistically grounded modes of composition.

Sometimes a more musically specific distribution is required. Using the example of a probability distribution influencing the outcome of pitches available from the 88 notes of a conventional keyboard, we could envisage a distribution that only allows notes from the major scale, and favours the root, fourth and fifth as prevalent in many Western musical idioms.

Thinking further about this, if an analysis can be performed on a suitable set of representational data (such as a score, MIDI file or features extracted from audio), the distribution of pitches in the scores of Chopin or based on pitch histograms in a selection of Javanese Gamelan recordings, one may surmise a very elemental manner of encapsulating the notion of style within a composition system.

Instinctively however, the fundamental feature lacking is any information about the appropriateness of the pitches given the surrounding context and overall trajectory. One extremely applicable method of addressing this is through the use of Markov processes.

#### 3.2.1.2 Markov Processes

A Markov chain is a probabilistic system that satisfies the Markov property of memorylessness. It consists of a set of discrete states with each state having an associated probability weighting of moving to every other state in the system. At any point in discrete time, the probability of a future event is solely determined by the current state (or a few states depending on the *order*) of the system without knowledge of the preceding past events. This can be expressed more formally by Eq. 3.1.

$$\begin{aligned} P &= (X_{t+1}|X_t = x_t, X_{t-1} = x_{t-1}, ..., X_0 = x_0) \\ &= (X_{t+1}|X_t = x_t) \end{aligned} \tag{3.1}$$

where $X_t$ is the state occupied by the state machine at time $t$ in its discrete history. To build a Markov chain, a transition matrix is defined that determines the probability of moving from one state to each other, often by analysing or modelling some real world examples. The order (i.e. the number of states in the right hand side of the second line of Eq. 3.1) of this transition matrix is the number of states that are considered when determining the probability of the next state. A first order Markov chain, for example, considers the probability of jumping to the next state based on the current state, while a second order chain would use the current state and the previous state in determining where to move to next.

Markov chains have enjoyed widespread application in simulating stochastic processes, and in music they are employed routinely for algorithmic composition (Fernández & Vico, 2013; Eigenfeldt, 2009; Jordà et al., 2016). A simple Markov chain for generating notes in a particular style could be achieved by building a transition matrix from the number of times the root note of a scale transitions to every other note in a corpus of MIDI files. This could be expanded to include a concurrent transition matrix that contains probability of transition of observed note rhythm values. Some of our colleagues in GiantSteps have also addressed drum pattern generation with a system of Markov Chains derived from style specific corpora addressing various strands of electronic dance music Jordà et al. (2016); Gómez-Marín et al. (2016).

With The Continuator, Pachet (2002) has garnered considerable exposure in the application of Markov chains in call and response style interaction with a human performer[21]. The allure of Pachet's work has been its satisfaction of a number of Turing style tests (Pachet, 2008). More recently its successor AI, FlowMachines (Pachet et al., 2008), has stirred up great publicity[22] with its generation of the first artificially intelligent pop song[23]. In their usage of Markov chains, they have acknowledged their facility for modelling temporal phenomena such as music, but emphasise the challenge in reconciling them for interactive control. To solve this they have proposed a

[21] And we shall see later Biles (1994) achieves something similar with his means of "trading fours" with a real-time listening GA.

[22] http://www.flow-machines.com/tag/flow-machines-press/

[23] With the caveat being a human wrote the lyrics and arranged and performed the final work.

class of hybrid Markov chain system that includes elements of constraint satisfaction, known as an Elementary Markov Constraints (EMC) (Pachet & Roy, 2011), proving fundamental for chord generation in the Continuator (Pachet et al., 2011; Barbieri et al., 2012).

Music generated by Markov Chains can be deceptive in their 'low-level' convincingness but it is still very localised in contrast to larger human-composed works with careful macroscopic attention to form. As Collins & D'Escrivan (2017) comment on The Continuator: "it is fundamentally parasitic on the duration data passed to it" and has "difficulty with longer-term structure".

To combat this the composer can increase the order of the Markov system - thereby lengthening the 'memory' of consideration in the state machine - or similarly lengthening the size of the snippets of symbolic music atoms that are to be used for chaining, but the results still baulk at a point. As Roads observes when studying hymns generated with a Markov chain by Brooks et al. (1993).

> "*...low-order chains generated meandering, random melodies, while high-order chains consist of parts of the original hymns spliced together, such as the first half of one hymn crudely spliced onto the second half of another hymn. This can be explained as a strong probabilistic tendency towards coherence over several notes that is broken suddenly by a more unlikely event, at which point a new coherent melody starts"*
>
> (Roads, 1996)

Jordà concurs in his own evaluation of his own compositions for live computer music generating jazz chord progressions.

> "*Markov music usually presents a very short correlation, sounding much better over brief fragments, than over long ones.* (Jorda, 1991)

With these criticisms in mind, maybe this makes the case for hybrid systems that can scale with the demands of larger scale computer music composition. By invoking a layered approach with multiple agents perhaps taking the role of a meta-composer for the purposes of coherent musical form. In fact Eigenfeldt does talk about this in the context of musical meta-creation (Eigenfeldt et al., 2016), and prior to that defines some mechanism that is rudimentarily capable of doing something to that effect. In *GESMI* - the Generative Electronica Statistical Modeling Instrument (Eigenfeldt & Pasquier, 2013) - for example, expert annotations of aspects of electronic dance music including subdivisions into section labels such as 'lead-in', 'intro', 'verse' and 'breakdown' are used by a genetic algorithm to dictate higher-level surface features while Markov chains generate localised motifs as usual. In another work (Eigenfeldt, 2016), another high-level meta-agent defines a series of parametric curves that affect

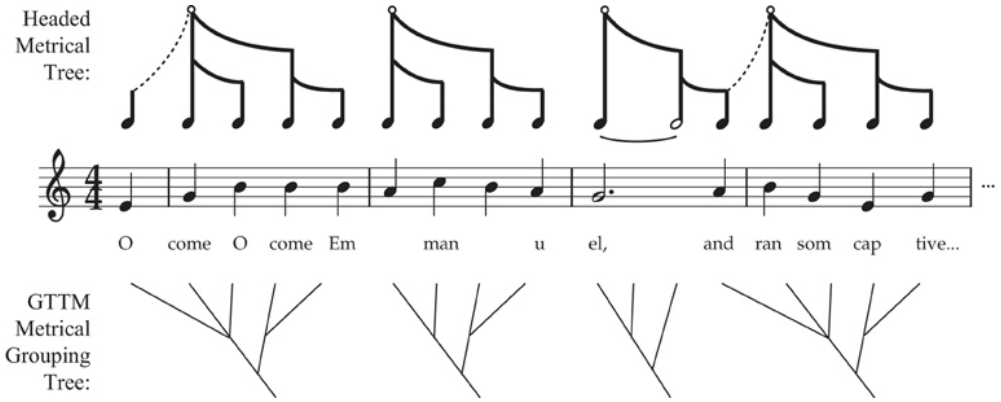



**Figure 3.2:** GTTM Metrical Hierarchy Grouping. Image from **?**.

overall compositional aspects such as speed, density and complexity that are adhered to by other reactive generative agents.

### 3.2.2 Formal Grammars

Continuing with the goal of epitomising style in computational models and generative procedures, we turn to another approach that tries to achieve this using language theory. Formal grammars have long been used to interpret hierarchical relationships in syntactical aspects of language (Chomsky, 1957), and are an essential element of compiler theory and programming language design (Garshol, 2003; Wirth et al., 1996).

The controversial debate continues as to whether music can really be considered a universal language (Campbell, 1997; Savage et al., 2015) but, nevertheless, applying formal grammar theory has enjoyed considerable utility in the systematic analysis of musical concepts as well as generative tasks. In his famous Harvard lecture series "The Unanswered Question"[24](Bernstein, 1977), the conductor Leonard Bernstein pondered whether music could be characterised using Chomsky's formal generative grammars (Chomsky, 1957). Subsequently inspired by this hypothesis, musicologist Fred Lerdahl and linguist Ray Jackendoff (1985) joined forces to compile "A General Theory of Tonal Music (GTTM)", one of the most complete compendiums to classify Western classical music using grammar techniques. A dense and complex work, it nevertheless provides a methodological framework for understanding what we hear when we listen to common practice works. The metrical structure rule for instance, positions beats in a tree structure such that it accounts for the natural strong and weak grouping of accentuation that occurs at different metrical levels of pulse as can be seen in Figure 3.2.

[24]https://youtu.be/MB7ZOdp__gQ

Lerdahl and Jackendoff's GTTM represents, as Roads suggests, an "analysis of syntactic structures" where musical events are parsed into syntactic hierarchical structures (Roads, 1996). Conversely he contrasts the analytical viewpoint with the synthetic: "starting from a specification of large-scale syntactic structure fill in the details for each of these structures".

The GTTM essentially comprises an analysis or distillation of style, the style loosely being common practice era Western art music. Composer David Cope (1991) has, using his musicological expertise, performed a similar analysis of music from this era, but with a more finely grained focus, initially on Bach chorales. More importantly the primary goal of this analysis is to inform the eventual synthesis of style. Cope distills his Experiments in Musical Intelligence (EMI) (Cope, 1987) process into three critical stages that mirror Roads' observations regarding duality of musical application involving formal grammars. In any case he identifies them as:

- Deconstruction - segmentation and analysis of existing representative work
- Signatures - identifying demonstrative motifs and traits that distinguish a stylistic genre or composer's proclivity
- Compatibility - recombine deconstructed elements into fresh cohesive and coherent musical works

Controversy aside, what Cope perhaps highlights most of all is the role of the designer in building computer composition systems. As is evident, building a software that can achieve the output of Experiments in Musical Intelligence (EMI) is no easy feat, demanding a Bach scholar with knowledge of the nuances of voice leading, counterpoint, as well as computational musicological skills that include awareness of AI methods and programming in LISP[25] for example. On a more philosophical note, it should follow that the designer then must impart some aesthetical footprint on works produced by such systems. So if we can hear which part reveals a piece to be Bach, then which part is the designer? Which part is the computer?

Cope has available 5000 examples of computer generated Chorale in MIDI format on his website, and has continued his experiments to include more larger scale works extending into the common practice era and beyond (e.g. Mozart and Mahler (Muscutt, 2007; Cope, 2009)). His efforts have had an enduring and frequently contentious impact not only in academia (Wiggins, 2008) but also in the public's psyche; by addressing well known and easily identifiable stylistic goals (verifiable once again with musical style Turing tests) he establishes clear markers in the sand for what is achievable with algorithmic composition.

[25] LISt Processor (LISP) is a programming language with a long association in the field of AI and by extension, computer music (Taube, 1991, 2004; Assayag et al., 2006).

### 3.2.3 Genetic Algorithms

Genetic Algorithm (GA)s, as introduced in seminal works by Holland (1975) and Goldberg (1989), are a class of heuristic search solutions modelled on the theory of natural selection (Koza, 1992; Srinivas & Patnaik, 1994). Along with artificial neural networks and emergent systems they can be classified as a biologically inspired computing method that employs some well understood process from natural sciences to help solve a more complex computational task (Mitchell, 1998).

In a GA, potential solutions to a conceptual problem are encoded with a genome string representation (usually with binary bitstrings). Using a series of genetically-inspired operations such as crossover and mutation these strings are "evolved" until their quality reaches a satisfactory level, hence performing an implicit search sweep over a large space of candidates. Central to the operation of evolutionary algorithms and determining genomic quality is the *fitness function*, which determines the individuals that are allowed remain in successive populations. Naturally effective representations are important to ensure effective use of evolutionary algorithms - a badly defined encoding can skew the performance towards that of exhaustive search.

Several logical stages are involved in a genetic algorithm which we now discuss in turn:

**Initiation:** To initiate the search space, a population of individuals is generated randomly, though some rules-based procedures can assist in "kick starting" a faster, more informed process.

**Fitness:** The fitness function is a problem-dependent evaluation procedure that determines the suitability of the candidate solution to the problem. There are no hard and fast rules as to what determines a "good" fitness function; this is - along with specifying an appropriate representation - part of the craft in evolutionary algorithms. An alternative is to use an interactive fitness function, which involves using an external agent (typically a human critic) to determine suitability of candidates.

**Selection:** In accordance with Darwinist theory, the fittest $f$ individuals are more likely to be chosen for reproduction and generation of future population members. This is in essence probabilistic, and one popular selection schema is roulette wheel selection, or fitness proportionate selection. Here, the probability assigned to an individual is given by the proportion of the individual fitness to the overall population (Eq. 3.2).

$$p_i = \frac{f_i}{\sum_{i=1}^{N} f_i} \tag{3.2}$$

Where $p_i$ is the probability of individual $i$ being chosen and $f_i$ is the fitness of that individual.

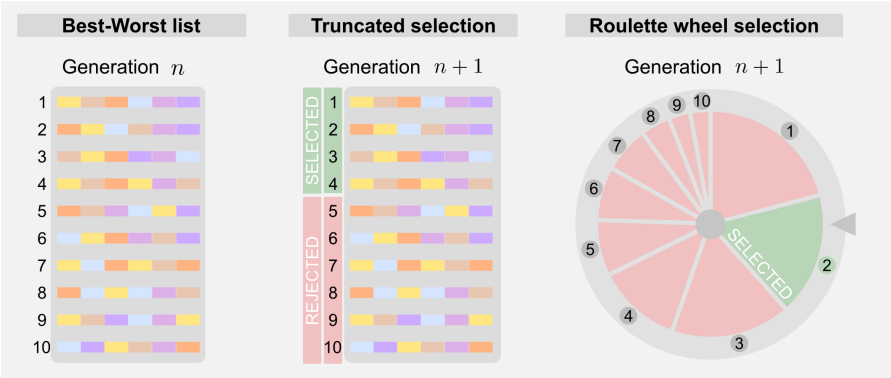



**Figure 3.3:** Sorted best to worst list of candidates (left). Truncated selection of top 4 candidates (middle). Probabilistic roulette wheel selection (right). Image from Dissecting Reinforcement Learning[26]

Roulette wheel selection derives its nickname from being akin to reserving a portion of a spinning roulette wheel in a casino. In Figure 3.3 this can be graphically compared to truncation selection which simply selects the top *N* members from the list when sorted by fitness.

To create the offspring, genetic operators are applied to the parent individuals, the two most common being:

- *Crossover* - pick a splice point s in the bit strings of the genome, take bits 0 - s from parent A and combine with bits s - n from parent B.
- *Mutation* - a certain (small) portion of the newly generated population is chosen at random and within these individuals a random bit is flipped. As in nature, the mutation procedure allows for genome sequence combinations that may not occur using crossover alone.

#### 3.2.3.1 Evolutionary Music

Genetic algorithms have certain qualities that make them an attractive choice for artificial creativity, so much so that there exists an ardent strand of research devoted entirely to what has come to be known as "evolutionary art" (Romero et al., 2008; McCormack, 2013) and furthermore, evolutionary music (Miranda & Biles, 2007) .

Undoubtedly, they are an elegant and easy to comprehend search heuristic, and the underlying concept of generating a bunch of candidate solutions that hopefully converge on some ideal is appealing to the process of an artist. So too is the idea of the interactive fitness function, where humans can directly appraise potential candidates subjectively. Naturally this is the most optimal method aesthetically but the slowest method performance-wise, especially in highly temporal domains such as sound and

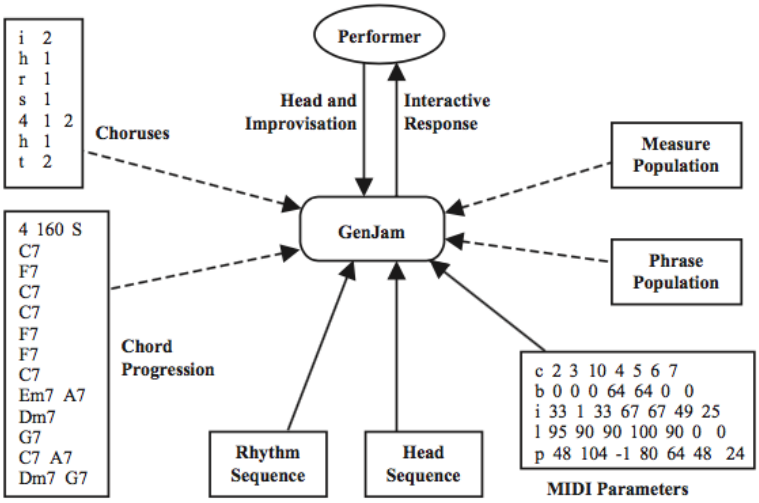



**Figure 3.4:** GenJam architecture with different populations and interaction with performer. Image from Miranda & Biles (2007).

vision, where it is typically referred to as the "fitness bottleneck" (Todd & Werner, 1999; Biles, 2001; Gartland-Jones & Copley, 2003).

One of the most well known applications of genetic and evolutionary algorithms to music has been Al Bile's GenJam system (Biles, 1994, 2002, 2003, 2007) which interactively generates streams of monophonic solo melodies in bebop jazz style. It is a classic example of an interactive genetic algorithm Figure 3.4, where the composer considers himself a "mentor" to the system, actively incrementing and decrementing fitness of generated phrases as they arise in real time from the program. Internally, the genetic algorithm actually maintains two populations of structures. The phrase population contains a list of indices that reference measures in the measure population. A measure is considered a short eight-note phrase of MIDI pitch values, with special symbols for silence and held notes. One of GenJam's unique feature is the ability to "trade fours" with the live performer whereby pitch detection is performed on a monophonic input source, and the phrase that is altered using genetic modifiers is played back to the performer.

Now that we have seen a glimpse of basic methods for symbolic algorithmic composition, we will begin our study of symbolic generation focussing on rhythm using computer processes. We will turn our attention to some key works presented in the literature and describe a new method of target-based rhythmic pattern generation using genetic algorithms. An evaluation will unearth the feasibility of such methods for rhythm pattern generation and the limitations that can be identified that informs a revision of the models used.


**Figure 3.5:** Roland TR-808 Drum Machine

## 3.3 Symbolic Rhythm Generation

### 3.3.1 Representation

As evident from such designs as Léon Theremin's 'Rhythmicon', the Wurlitzer 'Sideman' and Raymond Scott's 'Rhythm Synthesizer'[27] (Tindale, 2009; Malmberg, 2010; Arar & Kapur, 2013), there have been many esoteric early incarnations at building and incorporating drum computers, with these strange experiments in the early half of the 20th century paving the way for a more unified design aesthetic emerging towards the latter half.

Most notably, one can not underestimate the profoundly influential impact of the designs of Ikutaro Kakehashi and the Roland Corporation on the shape and history of electronic music and its culture. Originally considered a commercial flop due to its perceived inauthentic synthesis of natural drum sounds, the Roland TR-808 drum machine (Figure 3.5) has since become the icon of post 1980s dance oriented music, and its distinctive synthetic character soon became synonymous with techno and house styles (Théberge, 1997).

The TR-808 was intended as a rhythm accompaniment device, and as such was designed to be easy to use by non drummers. Its operation involved selecting the required drum sound and inputting a pattern using a row of 16 buttons corresponding to discrete points in time based on a global tempo. This binary based approach has continued on into computer music platforms and virtual instruments.

This simple but concise metaphor of representing basic patterns has been adapted also in the literature dealing with the musicological study of rhythm, most notably by Toussaint (2013) who has carried out considerable research into geometric approaches for rhythm analysis and similarity. The binary input mechanism is clearly a convenient medium for transferring to computational analysis while remaining de-

[27] The Rhythm Synthesizer would sow the seeds for a later drum machine called 'Bandito the Bongo Artist' which would be used on the ground-breaking proto-ambient record *Soothing Sounds for Baby*(Collins & D'Escrivan, 2017)

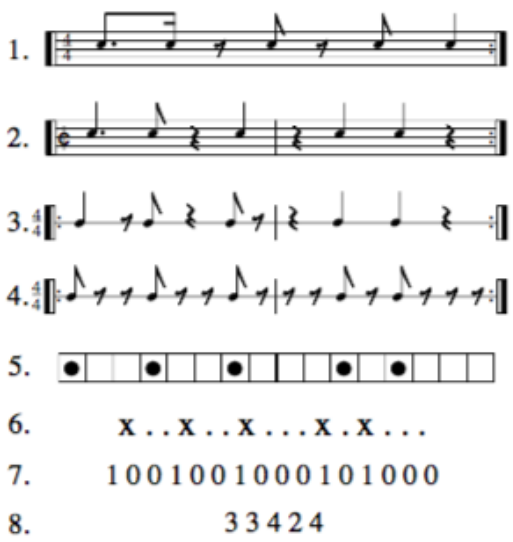



**Figure 3.6:** Different Forms of Symbolic Rhythm Representations (Image from (Toussaint, 2003)

scriptive and intuitive for human comprehension. Figure 3.6 shows how a particular rhythm can be represented in a number of different symbolic representations including traditional staff notation forms.

The fifth form in the figure, the most familiar to drum machine interfaces, is referred to as the Time Unit Box System (TUBS). It is considered easier by many ethnomusicologists in depicting and grasping complex rhythmic structures such as polyrhythms not typical in Western Art Music (Nzewi et al., 2008; Koetting, 1970).

Continuing in this line, other representational schemes have been devised in attempt to represent rhythms in a more holistic and natural manner, given its cyclical and hierarchical centricity. Often rhythms are portrayed using a circle or necklace system with beats and onsets occupying geometric portions of the structure, as can be seen in Figure 3.7. These are frequently used to depict more complex rhythmic patterns that exist in African and Latin traditions, giving a different perspective and insight into their latent balance and symmetry.

These circular representations have proved alluring for computer music practitioners, and a number of rhythm focussed applications have been developed exploiting this necklace paradigm. Xronomorph (Milne & Dean, 2015; Milne et al., 2016), provides an environment for experimenting with rhythms that conform to mathematical properties of perfect balance and well-formedness. As a unified mobile application, Rhythm Necklace[29] is a visually appealing application for iPad for experimenting with geo-

[29]http://rhythmnecklace.com/

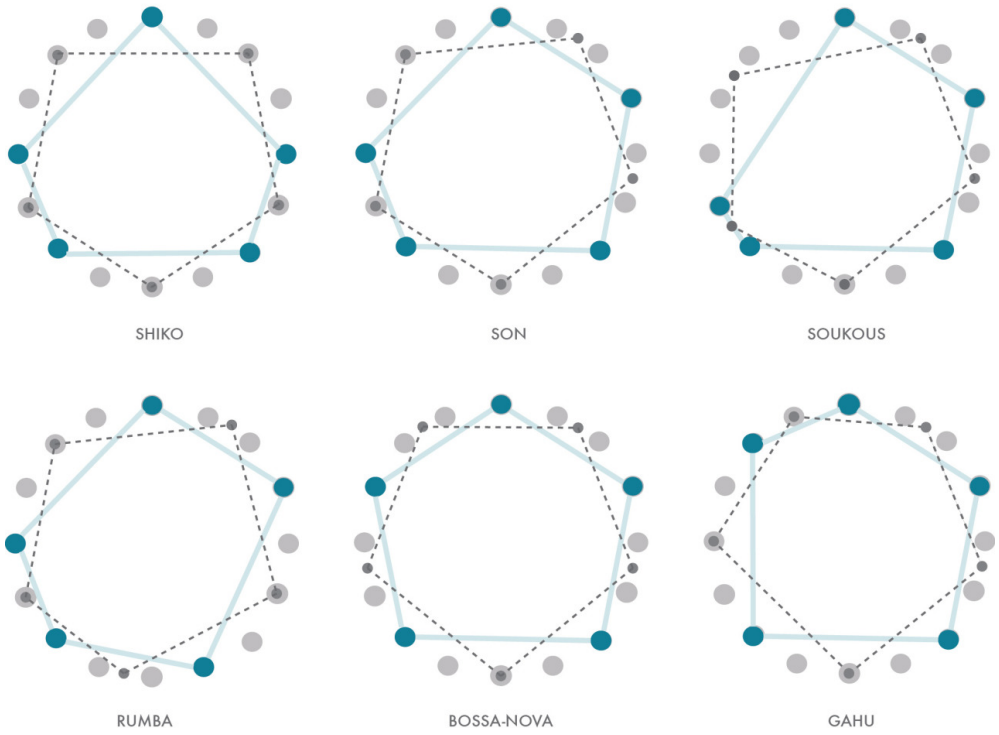



**Figure 3.7:** Necklace depictions of some popular rhythms.(Image from Ethan Hein[28])

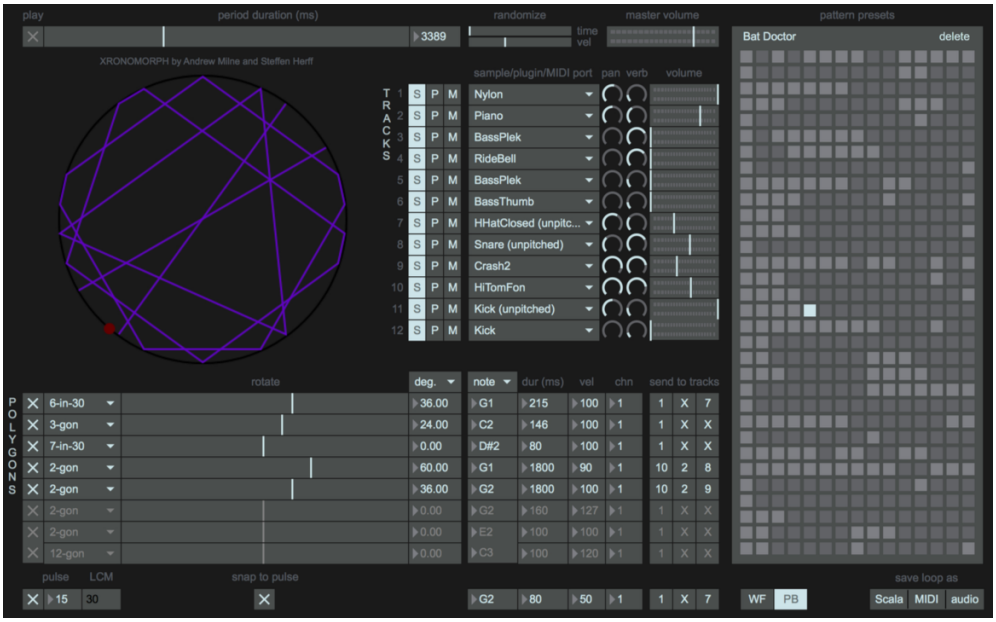
metrical approaches to rhythm generation.

### 3.3.2 Measuring Rhythmic Similarity

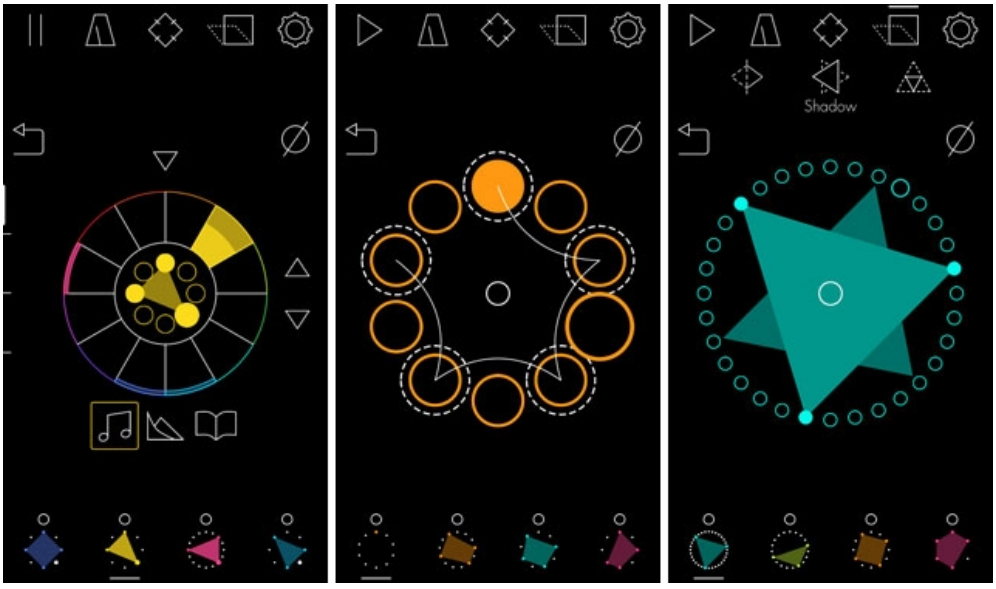
One of the primary tasks that has dominated symbolic rhythm study has been the formalisation of methods for measuring the degree of similarity between two rhythmic patterns. It was previously shown how natural rhythms can be distilled down to a very simple representational form suitable for computational analysis (and hopefully generative composition).

Obviously, there is a wealth of knowledge and systematic approaches for measuring similarity borrowing from information sciences when dealing with computer readable data, or in particular, strings. Methods of comparing strings, or string metrics, are an essential aspect of coding theory. Symbolic rhythmic study has adapted many of these metrics for its analysis and Toussaint (2004a) has summarised many of the key matching techniques. Superficially, we can take a distance measure, feed it some rhythms encoded as binary strings and do some basic normalisation/algebra to convert it to some degree of "similarity".

The key question here is how do these objective information metrics compare with the




**(a)**




**(b)**

**Figure 3.8:** (a) XronoMorph Interface (Image from Milne et al. (2016)), (b) Rhythm Necklace App (Image from Ethan Hein[30])

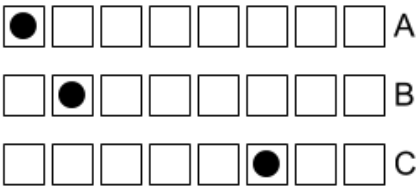



**Figure 3.9:** Same Hamming Distance for Three Patterns

cognitive human impression of similarity? With this very shallow abstraction of what a rhythm is, we must surely lose a great deal of latent information that undoubtedly must contribute to similarity perception; consider the influencing factors of accentuation, swing, timing imperfections, instrumentation, timbre, production, genre and surely, culture.

Regardless, the foundations of understanding rhythmic similarity need to begin somewhere. We turn our attention now to methods of estimating similarity of simple binary string representations of rhythmic structures and discuss their relevance and application for generative purposes in a practical system in due course.

#### 3.3.2.1 String Similarity and their Application to Rhythms

The primary advantage of the simple binary representation exemplified by the TUBS notation is its malleability for simple bit comparison and string analysis using common algorithms. We summarise a number of them here for clarification, but Toussaint offers a more thorough synopsis of many of them and their application to rhythm.

**Edit Distance:** The edit distance or Levenshtein distance is a widely-used measure of difference between two strings of information. It counts the number of insertions, deletions and substitutions required to transform one string into the other. The Levenshtein distance between the two strings “kitten” and “sitting” for example returns 3, based on two substitutions and one addition. The edit distance can operate on strings of differing lengths, and is solved using recursion or dynamic programming techniques.

**Hamming Distance:** The hamming distance counts the number of positions in which two equal length strings differ. A simple measure that is easy to compute for the machine and easily comprehensible for the human. Hence it is a restricted version of the edit distance, limiting itself solely to substitutions of symbols.

**Swap Distance:** As Toussaint (2013) notes, one of the problems with the edit distance and the Hamming distance is that they don’t capture the horizontal aspect of string similarity so well. In Figure 3.9 for instance, comparing string A to string B

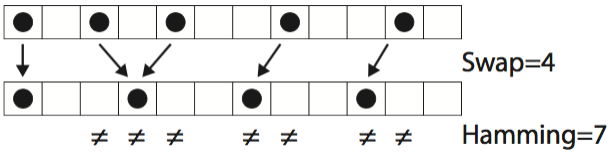



**Figure 3.10:** Swap and Hamming Distance Compared.

and string C both return a distance of 2, even though comparing them horizontally they are clearly very different.

To this effect, Toussaint proposes the use of the swap distance, actually borrowed from similar computations that take place in the realm of bioinformatics, to capture the number of placewise shifts to make one string match the other. as he did in his study of African ternary rhythms (Toussaint, 2003). In the simple case of binary encoded rhythms with equal number of onsets or 1s - known as one-to-one mapping - the distance can be computed by summing the absolute differences in their indices (Toussaint & Oh, 2016) - essentially the L1-Norm or "Manhattan" distance (Eq. 3.3) when considering element $a_i$ and $b_i$ from two strings $A$ and $B$.

$$\mathcal{D}_{\mathrm{S}}(A,B) = \sum_{i=1}^{N} |a_i - b_i| \tag{3.3}$$

Of course natural rhythms can have different numbers of onsets and this does make computing swap like similarity considerably more complex. To handle this extended case of many-to-many, Díaz-Báñez et al. (2004) have applied what is known as the directed-swap in their analysis of flamenco patterns, which stipulates, when comparing a pattern A with more onsets than pattern B that:

1. Each onset in pattern A must move to an onset in pattern B.

2. Every onset in pattern B must receive at least one onset from pattern A.

In Figure 3.10, we can see a visual comparison of the directed-swap distance as compared to the Hamming distance, as well as the different distance values they return. Conceptually, we can at least surmise that the directed-swap, with its ability to encode the horizontal displacement of onsets, should give a richer description of the similarity between two rhythms, such as embodying an impression of the syncopation introduced. if we were to move between both patterns.

Actually computing this distance is also a more involved task. One proposed, which works in $O(n^2)$ time, solution is to consider the task as finding the shortest path within a weighted acyclic graph, in the context of solving what is referred to as the restric-

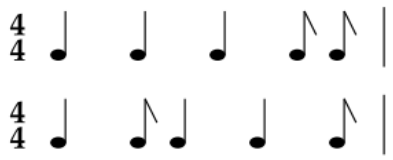


**Figure 3.11:** Comparing the identical IOI of two very different rhythms. Image from Dixon et al. (2004)

tion scaffold assignment problem in computational biology (Colannino & Toussaint, 2005).

#### 3.3.2.2 Other Symbolic Rhythmic Descriptors

Here we summarise other useful descriptors pertinent to symbolic rhythmic analysis that are not exclusively based on determining similarity between two rhythms.

**Inter-Onset Interval:** Rather than computing the difference between two rhythmic patterns based on the location of the onsets in time, the Inter-Onset Interval (IOI) summarises a rhythm based on the time elapsed between each onset - or in other words the offset time (Toussaint, 2013). For example the clave son would be encapsulated in the string of time elapses as {3,3,4,2,4}, using some unit of reference such as 1 = $\frac{1}{4}$ note.

Clusters of inter-onset intervals that occur periodically, at a regularly rate that can be quantified per unit time, are referred to by some in the literature as "temporal atoms" or tatum (Bilmes, 1993; Sethares, 2007; Jehan, 2005). Essentially we can consider the tatum of a score or recording as the fastest note division in which it is written or performed (Sethares, 2007). We shall learn later in Chapter 4 that the notion of tatum is important in extracting the tactus or, in musical information retrieval parlance, beat tracking. The computational task of beat tracking involves determining from a signal where a human is likely to tap their foot[31] when listening to music with a clearly discernable rhythm.

While IOIs are unedeniably one of the most important contributors to the impression of accent and pulse perception in music (Parncutt, 1994), they are not the most useful feature for comparing rhythms (which is what we are trying to achieve). Firstly as Dixon et al. (2004) correctly point out, two rhythms can have the same distribution of inter-onset intervals, but depending on the length of the notes themselves, the perception of their similarity can turn out to be very dissimilar indeed. In Figure 3.11 the two rhythms have identical inter-onset intervals but the rhythm above is more typical of a Cha Cha rhythm while the rhythm below is clearly more syncopated and betrays more stylistically that of Rumba.

[31] Or more formally, the period at which the conductor swings their baton

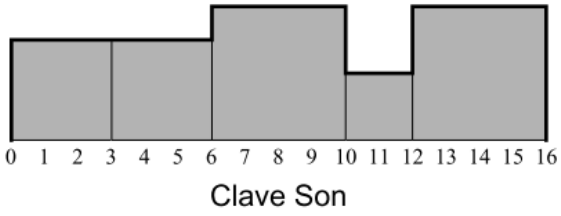



**Figure 3.12:** Chronotic Distance of Clave Son. Image from Toussaint (2004a).

**Chronotonic Distance:** Addressing the aforementioned shortcomings intrinsic to IOI-based comparisons, Gustafson (1987) expands the IOI representation to a two-dimensional structure that also encapsulates interval time by the integrated area of the rectangle. This elegant depiction allows for clear visual comprehension of rhythmic patterns[32] as we can see in Figure 3.12 with the familiar Son.

(Toussaint, 2004a) explains how to compare two patterns summarised by their chronotonic distances by treating them as probability distributions and using standard statistical distance measures. While he reports that the chronotonic distance edges past the swap distance based on the criteria of simplicity, goodness of fit, clustering, and "ancestral" rhythm generation (rhythms that serve as the seed for many many others), Guastavino et al. (2008) provide experimental evidence that the directed-swap correlates more highly with classically trained musicians' perceptual impression of dissimilarity when comparing rhythms.

## 3.4 A Genetic Algorithm based on Similarity

The previous section introduced symbolic representation schemas for rhythm, and symbolic rhythm descriptors, with a particular focus on similarity measures between two rhythms. In this section we will explore how this systematic knowledge can be encapsulated in generative systems. We will begin with a glimpse into some existing relevant work, firstly looking at generative systems that deal with Toussaint style measures of similarity, followed by systems that use genetic algorithms to generate rhythm. Subsequently, we introduce our own proposed work that seeks to specifically merge perceptual measures of rhythmic similarity with genetic or evolutionary algorithms in a system we call GenDrum. We discuss the evaluation of the system and our conclusions that motivates our progression from the symbolic domain to direct content-based analysis and recombination of audio.

[32] Actually originally intended for scrutiny of rhythm within phonetic study

### 3.4.1 Generative Rhythmic Systems

#### 3.4.1.1 Generative Rhythmic Systems using Similarity Measures

Publications dealing with generative applications of symbolic rhythmic similarity exist in the span of literature, but are not so widespread. The Hamming distance, with its easy comprehensible and computable qualities has naturally served a number of systems. Paiement et al. (2007) describe a paper centred around utilising the Hamming distance for generative modelling and prediction of simple rhythms trained in jazz and pop idioms. Vogl & Knees (2017) present a tablet application that uses the Hamming distance combined with a restricted Boltzmann machine - a type of artificial perceptron neural network - trained on many drum styles from the Native Instruments Maschine[33] collection. A modified extension has also been extended to database interpolation from that same collection in an earlier publication (Vogl et al., 2016). Other distance metrics have been exploited less so, and we have found no instances of the directed-swap distance being used in generative tasks.

#### 3.4.1.2 Generative Rhythmic Systems using Genetic Algorithms

Generative systems addressing rhythm with the means of GA are more abundant. Eigenfeldt (2006) describes his Kinetic Engine as a software component that generates rhythms in general, not specific to drum sounds. His approach to fitness evaluation is perhaps a controversial one, but not uncommon in musical applications. As in Al Biles' GenJam system, the role of fitness is simply eliminated. Thus the only real elements of genetic algorithms conserved are crossover and mutation. One may rightfully suspect that such a simplification renders the algorithm commensurate with a randomised search. Consequently a common solution used in such scenarios is to seed the initial population with a known dataset of good input. Another approach could be to embed some rule-based logic that restricts what type of candidates can be generated legally in the seeding process, as we carried out in (Ó Nuanáin & Sullivan, 2014).

Bernardes (2010) approaches genetic drum pattern generation from the point of view of style emulation. Like Eigenfeldt, they do not incorporate a fitness function, but choose to perform prior analysis on user-supplied or preset MIDI files in a particular style. This process gathers a probability distribution of weighted possible onset times in a 16-step pattern. The population is then seeded with candidates that have patterns generated according to this distribution.

A short paper by Horowitz (2004) suggests a multi-dimensional objective fitness function based on the combination of functional evaluations of syncopation, density, downbeat, beat repetition, etc.. Unfortunately the publication is rather scant on actual details regarding the implementation and evaluation of such a method.

[33] https://www.native-instruments.com/en/products/maschine/

Kaliakatsos-Papakostas et al. (2013) also use the notion of a a target pattern in their evoDrummer software, and define their fitness function by determining what they refer to as divergence in terms of "mean relative distance" to a base rhythm. The distance is computed based on a set of 40 features extracted from patterns, including descriptions of density, syncopation and loudness intensity. Indeed this is perhaps the most related work to our proposed system.

### 3.4.2 Developing a Genetic Algorithm that can Generate Rhythm

We set out to design a rhythmically motivated genetic algorithm that was capable of self-evaluation based on the similarity of the patterns it outputs compared to a user-provided seed target pattern. One can envisage this concept being extended to a system that can perform self-evaluation against a previously analysed dataset of patterns that conform to some notion of *style*.

This section introduces the tools we created in our research: SimpleGA, the genetic algorithm itself that can target the two well known music programming environments Pd and Max, and subsequently encapsulated in the real-time Max for Live instrument GenDrum. Max for Live is an extension of the Max programming environment inhabited in the popular DAW Ableton Live (Manzo & Kuhn, 2015). Ableton Live[34], in turn, is one of the most popular DAWs in the dance music community due to its non-linearity and grid-focussed arrangement of loops.

We continue with a detailed discussion of the implementation of the distance measures and representational issues. Finally there is a description of the design of the experiment for evaluating the research.

#### 3.4.2.1 Software Implementation

SimpleGA is a basic, general purpose genetic algorithm external object we developed for the Pure Data (Puckette, 1997) and Max/MSP environments in C++ using the cross-platform Flext framework (Grill, 2004) . It can handle binary, numerical and alphanumerical strings. A target string is supplied to the object and fitness is determined by measuring its distance to the generated strings using the Hamming or directed-swap distance as was introduced in Section 3.3.2.1. Bang[35] messages to the leftmost "hot" inlet causes the genetic algorithm to undergo a generational stage of evolution, then to output its fittest individual from the pool. Figure 3.13 shows the help patch for the object with this functionality explained visually.

Next we created the GenDrum instrument, a Max for Live device that uses the SimpleGA genetic algorithm to create polyphonic drum patterns of synthesised kick, snare, hi-hat and crash/clap type sounds that emulate the style of a pre-conceived

[34] https://www.ableton.com/

[35] The "bang" is a fundamental message in Pd and Max that basically means "do your thing!" (Winkler, 2001)

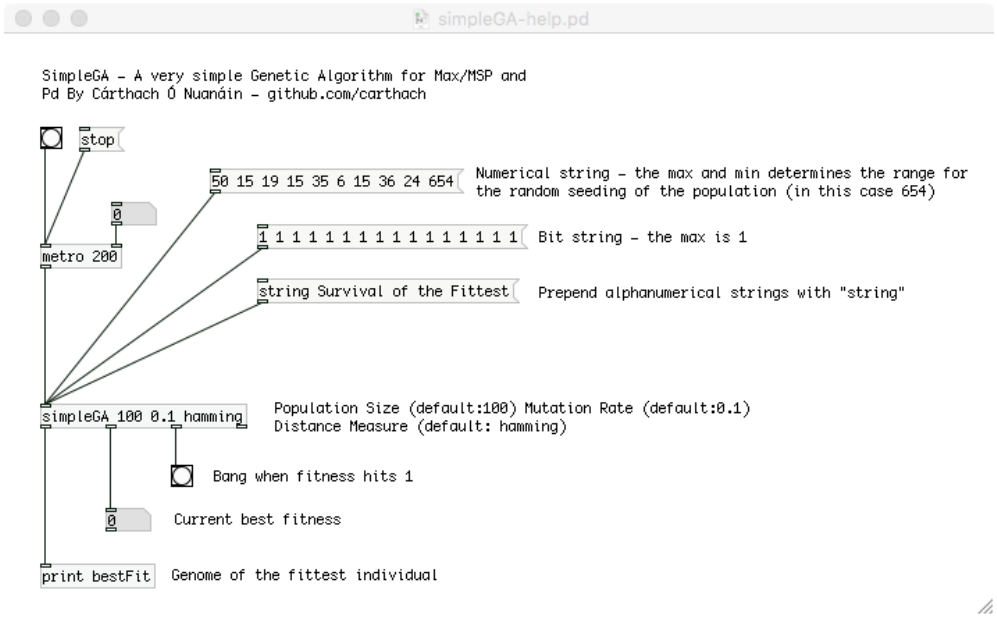



**Figure 3.13:** *SimpleGA* Pure Data Help Patch

(by input from the user) pattern (Figure 3.14). These four sounds are mapped onto a single 16-step row to create a 4x16 drum pattern matrix as is evident in the figure. The operation of the instrument is intended to be as simple as possible. The user inputs a desired pattern into the pattern matrix and assigns this as the target pattern. New target patterns can then be generated by clicking on "Get New Pattern". This assigns the best candidate in the current population to the grid and reports its fitness in the number box. In the yellow number box a target fitness can be issued to the genetic algorithm and clicking "Accelerate!" will increase the speed of the algorithm until it reaches this target. Patterns that increasingly approach perfect fitness now emerge from the algorithm, repeating and contrasting with the target pattern through repeated stages of GA evolution.

The instrument also attempts to provide some visual feedback on the evolutionary process using a type of force-directed graph. The fittest individual from each population is represented as a node with its fitness determining the size and distance in relation to the target individual.

### Algorithm Details

*Linear vs. Parallel Operation*

As the previous section explains, the instrument is intended as a fully functional drum machine, and hence is polyphonic with the possibility of multiple sounds concurring in time. As seen in the literature, similarity studies are predominantly focused with

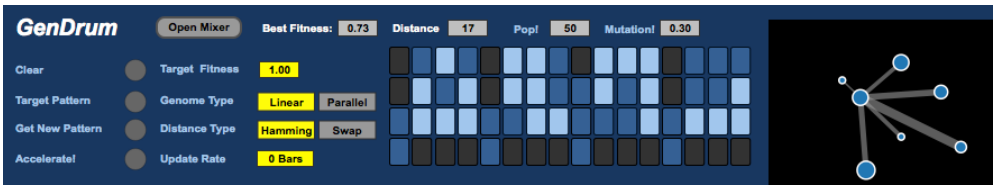



**Figure 3.14:** GenDrum Main User Interface

examining monophonic patterns such as the single sound clave son[36]. How we deal with the polyphonic implications in our research is outlined here.

Our first naive implementation of the genetic algorithm converts the 4x16 drum pattern matrix into a single "linear'" 64-digit binary string. Evidently this is a simplistic "brute force" approach that does not explicitly take into account any musical or perceptual aspect of the application. Specifically it does not embed any knowledge about the constituent sounds and the separation between them in the genome string. Essentially, we're treating it as a single sound 64-step pattern, with crossover and mutation happening at the halfway point between the first 16 bits of the kick and snare pattern and the second 16 bits of the cymbals/crashes.

Another approach is to force some kind of logical separation between the different polyphonic timbres in the pattern. By assigning a separate instance of the genetic algorithm to each timbre an overall mean fitness can be derived from the individual outputs. This we implemented and labelled as "parallel" mode. Since the genome pattern length is now split from one single 64-bit string to four concurrent 8-bit strings, the time it takes for the genetic algorithms to reach the target fitness is considerably reduced. Some tweaking of the parameters is required to reduce this convergence time and maintain a healthy level of diversity. Setting the population size parameter to 30 and increasing the mutation rate parameter to between 20% to 40% has been found to work well in our experience. Figure 3.17 shows the flowchart and design of the GenDrum instrument and it's encapsulation of the SimpleGA GA in the aforementioned linear and parallel configurations. Next we turn our attention to the distance measures used to derive the fitness function of these genome patterns.

*Hamming Distance*

In the review of the state of the art, we referenced how Post and Toussaint have surveyed the effectiveness of the edit distance in determining rhythmic similarity between two binary patterns (Post & Toussaint, 2011). Recall that the edit distance allows for insertion, deletion and substitution of symbols within the string. A simplification can be made if operations are restricted to substitutions only, and string lengths are equal (as is always the case in our representations) in which case it becomes the Hamming distance.

[36]The clave son is a two bar measure of either a 3-2 or 2-3 pattern that is integral in many forms of latin dances (Sethares, 2007).

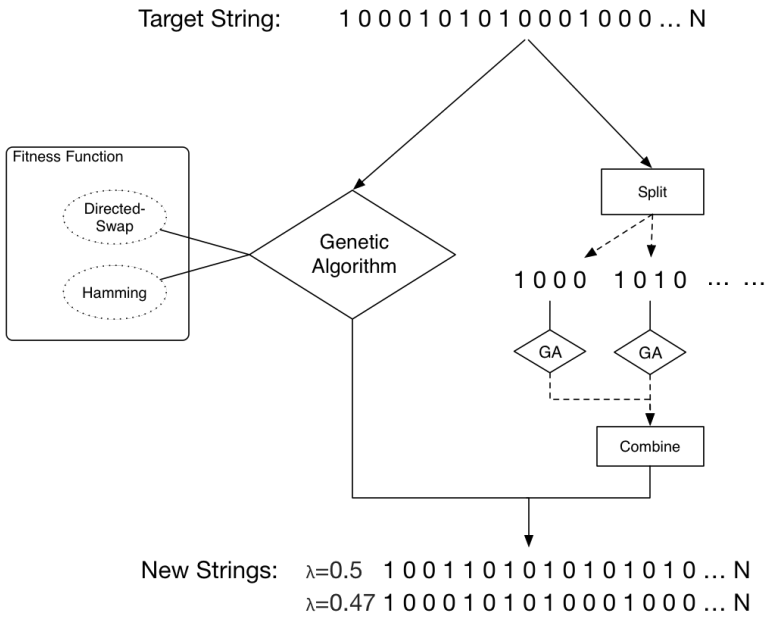



**Figure 3.15:** Genetic Algorithm Flowchart. The target string can either be fed to one single GA (left) or split by timbre, processed by 4 parallel GAs then recombined (right).

To derive a fitness function using the Hamming distance (Eq. 3.4, using the $\oplus$ exclusive OR operator to determine where the two binary strings are not equal), the number of positions where the two strings $a$ and $b$ match are counted and then divided by the total string length (64 for the linear string and 8 for each string in the parallel implementation).

$$f = \frac{1}{N}\sum_{i=1}^{N} a_i \oplus b_i \tag{3.4}$$

*Directed-Swap Distance*

The Hamming distance takes into account the correct positional scores between two patterns, but it does not give any indicator as to how different a pattern is in terms of the horizontal displacement. Intuitively one would think this horizontal displacement would reflect the important phenomenon of rhythmic syncopation. For example there may exist two different patterns with equal distance, but one pattern has more onsets aligned closer to the original pattern or exhibit a similar distribution of the strong and weak beats in the bar. For these reasons, we decided to test also the directed-swap distance metric as described by Díaz-Báñez et al. (2004) in the study of flamenco rhythms, with the hopes that it better captures the spatial comparison between the our generated patterns and the target.

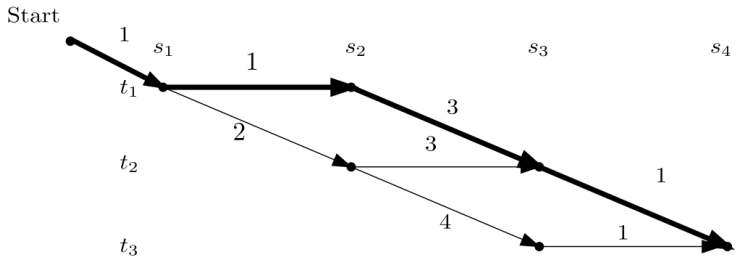



**Figure 3.16:** Solving the directed-swap distance for strings $A = \{0,2,7,12\}$ and $B = \{1,4,11\}$. Image and example taken from Colannino & Toussaint (2005)
.

Since no implementation existed for the algorithm proposed by Colannino & Toussaint (2005), we implemented it ourselves, using the Dijkstra implementation in the Boost C++ graph library (Siek et al., 2002) to compute the final distance[37]. The derivation and proofs are described at length in their article (Colannino & Toussaint, 2005) but the basic steps are as follows:

1. Convert the two binary rhythms under scrutiny to two sets of indices respectively. For example the pattern $A = \{0,0,0,1,0,1,0,0\}$ would reduce to the index vector $A = \{3,5\}$.

2. Construct a weighted directed acyclic graph from two index vectors $A$ and $B$ by firstly creating the set of vertices $v_{i,j}$ where $1 \leq i \leq |A|$ and $1 \leq j \leq (i + |B| - |A|)$.

3. For each $v_{i,j}$ add an edge to $v_{i,j+1}$ with weight $|b_i - a_{j+1}|$, and an edge to $v_{i+1,j+1}$ with weight $|b_{i+1} - a_{j+1}|$, if $v_{i,j+1}$ and $vi+1, j+1$ exist.

4. Create a start note and add an edge from that node to $v_{1,1}$ with weight $|s_1 - t1|$.

5. The directed-swap distance is given by the cost shortest path through the graph, solvable with Dijkstra's algorithm (Figure 3.16).

#### 3.4.2.2 Operation and Performance

One way to measure the performance of a genetic algorithm is to plot the fitness values it outputs over a succession of generations. In Figure 3.17 we can see the best fitness and average fitness for all possible configurations of the genetic algorithm over 100 generations. Notice that the linear string representation causes convergence on the target within 40 generations. Notice also that the parallel representation converges prematurely in areas of local maxima, particularly using the swap distance. This can

[37] https://github.com/carthach/SimpleGA/blob/master/SwapDistance.cpp

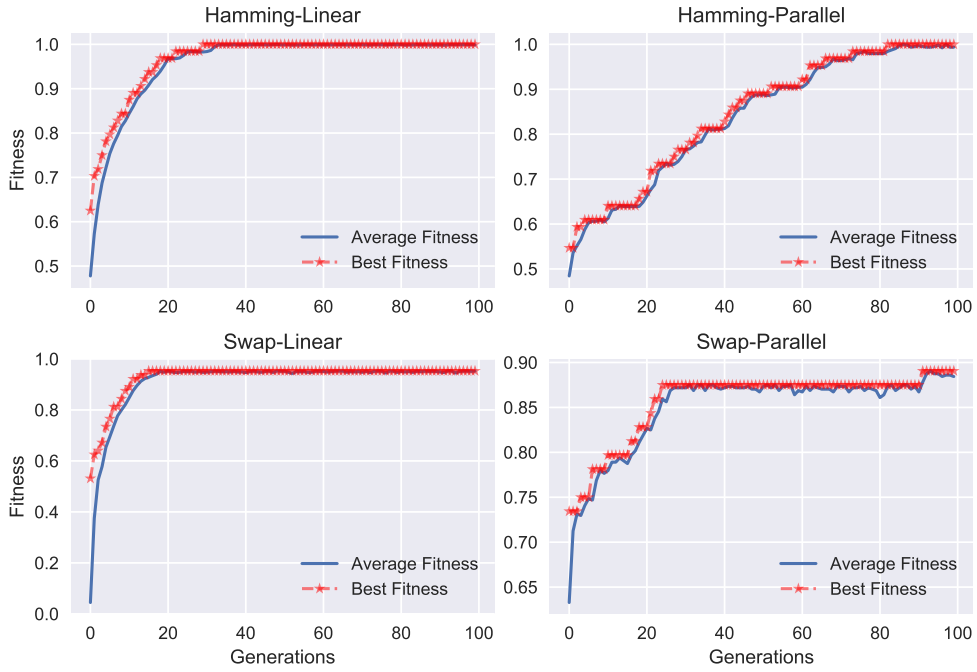



**Figure 3.17:** Fitness Plots for the Various Configurations of the Genetic Algorithm

be alleviated through some encouragement of diversity by tweaking the population size and mutation rate.

### 3.4.3 Listener Evaluation & Discussion

When evaluating this research and its resulting tools, our goals were to investigate:

- The overall correlation of distance with perceived experience.
- Comparing the impact of the Hamming distance versus the directed-swap distance on the perceived experience of similarity
- Comparing the impact of the linear versus parallel string representation.
- The more informal, subjective issue of the musical "interestingness" of the rhythmic patterns created.

We decided to conduct a simple listening survey to get listener feedback regarding on these aspects. The survey was web-based and unsupervised[38]; participants were sent a link with instructions on how to complete it.

[38] See Appendix B for more information

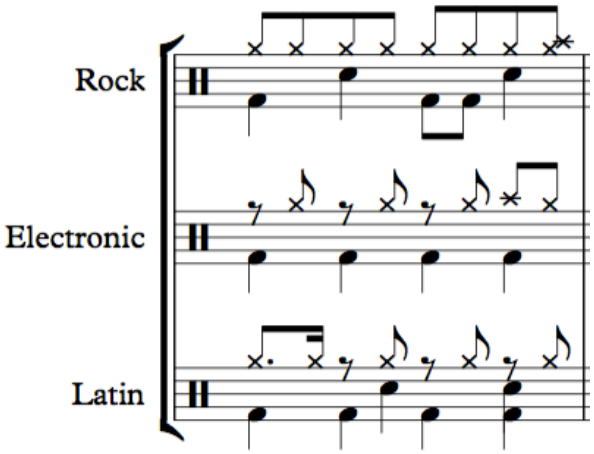



**Figure 3.18:** Rock, Electronic and Son-Based Pattern Used in Symbolic Listening Survey

The listening portion of the survey was divided into two parts. The first part examined similarity ratings by presenting the user with the target pattern and the algorithmically generated patterns. Participants were then asked to rate the perceived similarity on a five-point Likert ranging from "Highly Dissimilar" to "Highly Similar".

The second part then examined the "interestingness" of generated patterns. To enable the participant to ascertain this, the patterns were arranged in a soundfile as TPx2, GPx2, TPx2, GPx2 (TP=Target Pattern, GP=Generated Pattern) i.e. a two-bar loop of the target pattern is followed by a two-bar loop of a generated pattern and the whole sequence is repeated. This choice of configuration was quite arbitrary, but it was reasoned that in order to get a sense of the interplay between the target pattern and the generated pattern it was necessary to repeat the sequence at least once.

Once again a five-point Likert scale graded the ratings, this time with labels ranging from "highly disinteresting" to "highly interesting". Regarding the subjective interpretation of "interestingness", we instructed the participants to consider how the target pattern and generated pattern "flows" from one to another, and how the generated pattern "develops" on the target pattern in terms of introducing stimulating variation.

Three target patterns were used for the purposes of the test: a standard straight 8 rock pattern, a four-on-the-floor electronic pattern and a son-based latin pattern (Figure 3.18). Table 3.1 summarises all the variables under consideration for the evaluation. This resulted in a total of 72 Wave Audio Format (WAV) files with 3 fitness levels (inversely corresponding to the distance scores).

Twenty-two participants took part in the survey, mostly drawn from music students and researchers. All of the participants confirmed that they played an instrument, 7 of whom specified a percussive instrument. Eighteen out of the 22 participants reported the ability to read music. It took approximately 15 minutes to complete.

| Question | Measure | String | Pattern | Fitness |
|---|---|---|---|---|
| Similarity | Hamming | Linear | Rock | Low |
| Interesting | Swap | Parallel | Latin | Med |
| | | | Elec | High |

**Table 3.1:** Listener Survey Variable Summary

### 3.4.4 Results

Before carrying out the statistical analysis the responses were summarised by computing the mode[39] of the Likert scores for each stimulus. Two "interestingness" stimuli out of the total 72 (36 for similarity, 36 for interestingness) were removed due to high divergence of opinion (50% or more of the responses deviated by 2 or more Likert scale values from the mode).

Figure 3.19a shows the overall fitness (normalised from its original range of [0-100] to the Likert range [1, 5]) against the mean and standard deviation of the Likert scores by each stimulus. The dip in the middle can be attributed to the different ranges of distance and corresponding fitness produced by the two string measures; the Hamming measure outputs values in a much higher and narrower range. This is more evident looking at the divided plots in Figure 3.19b , which plots fitness against Likert scores for each combination of string measure and representation type.

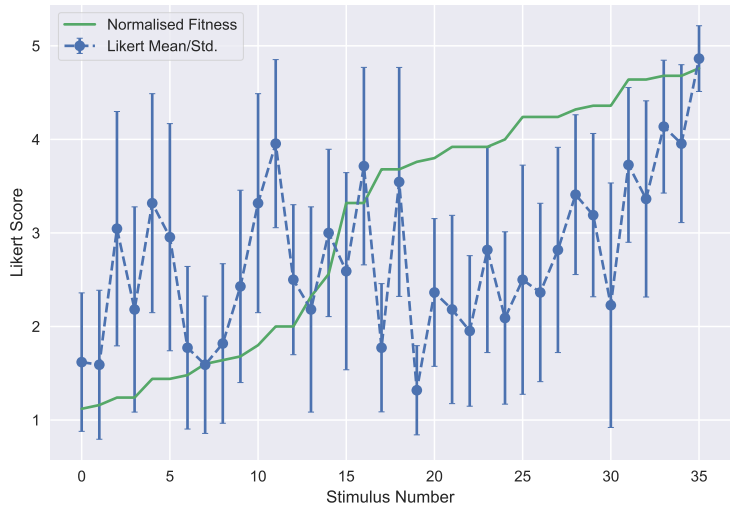
#### 3.4.4.1 Similarity Ratings

Our first task when looking at the data was to confirm whether inverse pattern distance and the fitness of the genetic algorithm correlates with the perceived similarity as determined by the participants.

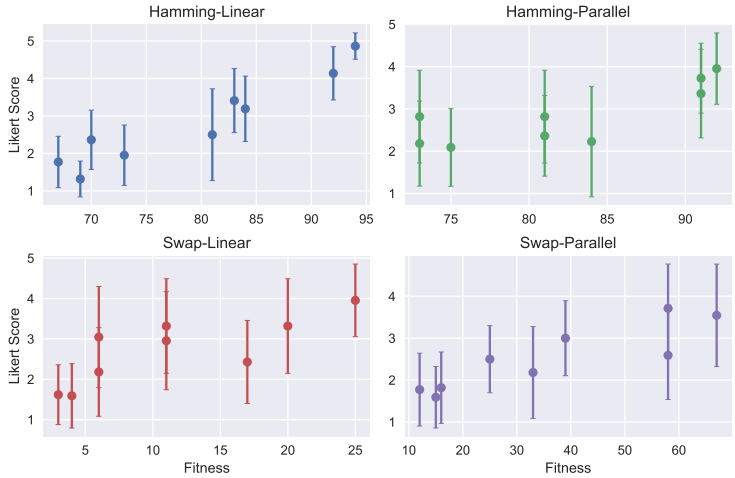
The data appears to suggest that this hypothesis is confirmed. Table 3.2 presents the Spearman ranked correlation matrix of fitness, distance and the mode scores received for each stimulus. There is a clear, strong negative correlation coefficient ($\rho$ = -0.71, $p < 0.05$) between the distance measure and the perceived similarity to the target. This correlation is not as clear with the fitness function, which can be attributed to the fact that fitness as a function of distance is evaluated differently for the two distance measures.

Figure 3.20 shows the separated distance and fitness correlations against the mode scores for the Hamming and directed-swap distance measures. The fitness correlation values are 0.784 and 0.625 respectively and the distance correlation values are -0.784

[39]Median or mode are generally considered more appropriate measures of central tendency when handling ordinal data such as Likert, and we use Spearman's Rho when performing correlation analysis for similar reason (Boone & Boone, 2012; Lantz, 2013).




**(a)**




**(b)**

**Figure 3.19:** Mean and standard deviation of fitness versus Likert scores for (a) each stimulus (b) measure and string type

| | Distance | Fitness | Score |
|---|---|---|---|
| Distance | 1.0000 | -0.4145 | -0.7134 |
| Fitness | -0.4145 | 1.0000 | 0.4353 |
| Score | -0.7134 | 0.4353 | 1.0000 |

**Table 3.2:** Overall Similarity Correlation Matrix

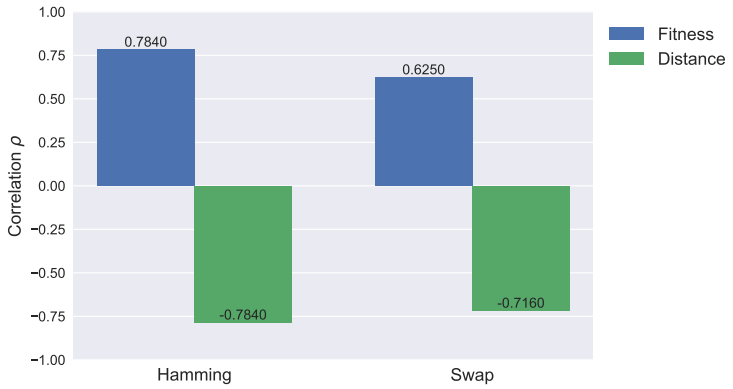



**Figure 3.20:** Fitness and Distance for Hamming and Swap Distance Metrics

and -0.716 respectively ($p < 0.05$). It can be seen that the Hamming distance has slightly better correlation.

Figure 3.21 shows the separated distance and fitness correlations against the scores when we discriminate between linear and parallel pattern strings. The fitness correlation values are 0.525 and 0.480 respectively, and the distance correlation values are -0.852 and -0.576 respectively ($p < 0.05$). The linear representation scheme appears to correlate better with human judgement.

Post hoc analysis using a t-test and a Kruskal-Wallis non-parameteric ANOVA H-test compared the two different conditions - linear vs. parallel and hamming vs. directed-swap - and discovered a significant difference for the ratings of the distance measure ($p < 0.05$), but not in the case of the string representation type. Out of interest, we also examined the participants' musical experience in reading notation or playing a percussive instrument, but they had no significant effect on the ratings ($p > 0.05$) in both instances.

We can draw some tentative conclusions based on this data. Firstly, the strong correlation between the overall distance and similarity ratings suggest that, even for polyphonic string representations of drum patterns, distances such as the Hamming and directed-swap are useful measures of perceived similarity. Secondly, splitting and recombining the bit strings by timbre, as carried out in the parallel operation, does not seem to offer improvement over the simplistic, long bit string representation. Finally, the more complex directed-swap algorithm, with its ability to capture the "horizontal" displacement between two drum patterns, does not appear to reflect an increase in similarity scores in our survey, confirming Toussaint's finding but also extending it to the case of polyphonic patterns.

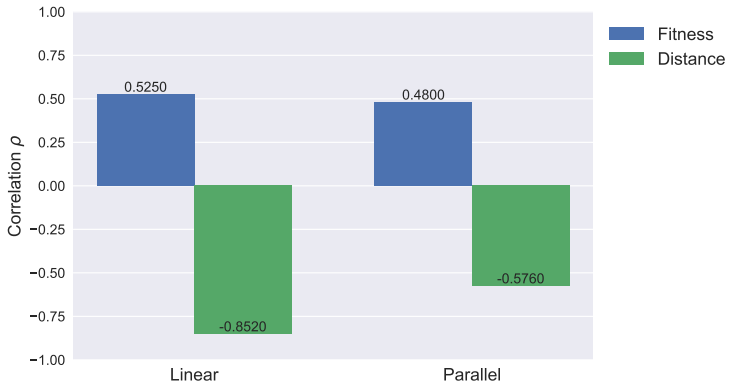



**Figure 3.21:** Fitness and Distance for Linear and Parallel String Computation

#### 3.4.4.2 Subjective "Interestingness" Evaluation

Table 3.3 presents the correlation matrix corresponding to the results of the users' impression regarding the "interestingness" of the patterns when heard in a sequence with the target. Curiously, the distance and fitness correlation coefficients are both positive, despite the fact that distance is inversely proportional to the fitness of the genetic algorithm but the p-values are so high (0.211 and 0.1887 respectively) that this data is not reliable. It is impossible to draw some meaningful or significant conclusions based on the disparity and inconsistency across subjects.

Disregarding the difficulty in appraising musically subjective output, the reason for this problematic data is largely attributable to the way in which we considered the notion of "interestingness" and how the question was formulated. Asking the participants to rate two essentially diametrically opposing qualities - i.e. variation (related to dissimilarity) and repetition (related to similarity) was a flawed approach that caused confusion. This became immediately apparent from some of the user feedback at the end of each survey session. For example:

> *"I noticed that I somehow prefer rhythms that are a natural evolution of the previous pattern, instead of being totally different. But if the similarity with the previous pattern is too high, the result is still uninteresting to me, because the resulting pattern is too predictable and loses every appeal."*

To complete the survey then, users often reverted to their own rules to determine their ratings, as evident from these comments:

> *"Interestingness" was hard for me to evaluate. In the end I rated with a 'good' interesting 'bad' interesting system: if it's weird but i like it, it's in the first case, if not, in the second case."*

| | Distance | Fitness | Score |
|---|---|---|---|
| Distance | 1.0000 | -0.5419 | 0.2201 |
| Fitness | -0.5419 | 1.0000 | 0.2310 |
| Score | 0.2201 | 0.2310 | 1.0000 |

**Table 3.3:** "Interestingness" Correlation Matrix

> *".. There are some times on experiment 2 that the new rhythm might not be interesting for a complete section but that might be useful as a bridge or as a temporal loop marking the end of a section."*

Another participant made the point that the pattern sequence may have forced some 'expectation' regarding the concept of "interestingness":

> *"... I have also the feeling that having the pattern repeated (ie AB twice), and therefore, having to come back to A again after having been in B, conditions very much the results."*

Despite the apparent issues with our method of evaluation, the data and informal feedback does suggest that the genetic algorithm creates "interesting" musical output. Nineteen stimuli out of the 34 analysed registered a mode score value higher than 4 as seen by the 56% green positive region in Figure 3.22 (there were two responses for "Strongly Disagree", but these were the two that were disregarded due to high divergence of opinion). The task ahead is to review the evaluation strategy in order to quantify and explain this aspect in a more coherent and predictable manner, decoupling the interdependency of the questioning between similarity and interdependence.

### 3.4.5 Improving the System - Adding More Features

The evaluation shows that our implemented methods do behave predictably in returning increasingly similar patterns to a target as the mean fitness of the genetic algorithm converges on its maximum. It also shows, despite deficiencies in the mode of questioning, that listeners do perceive these patterns favourably, given the sounds used and some localised awareness of style.

However we still felt that similarity methods alone were not capturing all of the subtleties inherent in symbolic rhythm patterns, features that other systems might include that prove more fruitful in analysis and generation. Another iteration of development on the genetic algorithm included three additional improvements that are described here.

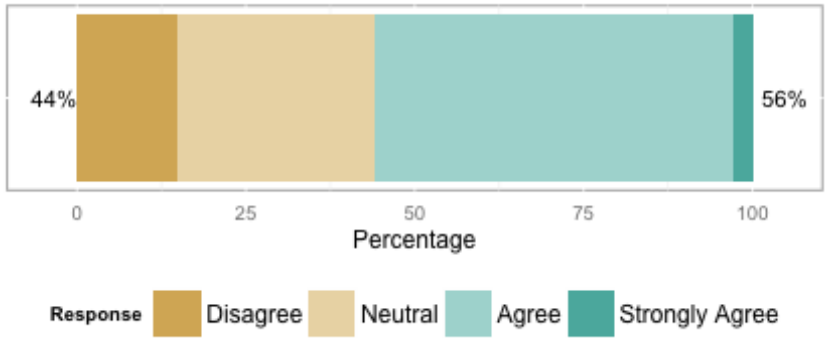



**Figure 3.22:** Distribution of Responses for 'Interestingness'

#### 3.4.5.1 Timbre Weighting

Other than allowing the separation of the pattern into its constituent timbres for discrete generation in the parallel mode, our generative system makes no additional effort to extrapolate and delineate the obvious hierarchical and emergent roles the separate timbres induce in the sensation of drum patterns.

Our brains process rhythm streams with different timbres and spectral profiles differently, and awareness and study of this fact goes some of the way to explaining why across different styles and cultures dispersion of timbres exhibit similar proportions across different frequency bands. To be precise we often see liberal placement of high-frequency, short attack timbres like hi-hats across all beats, while the kick drum usually enforces the downbeat[40] and start of bar and finally the mid-range snare drums fills in the backbeat[41]. As Merchant et al. (2015) observes in the general scope of pop music

> "*...the vast majority of drum patterns in popular music have bass drum hits on the downbeat (e.g. beats 1 and 3 of a 4/4 pattern) and snare hits on the upbeats (beats 2 and 4). While not inviolable, this statistical generalization leads listeners familiar with these genres to have strong expectations about how to assign musical events to a particular metrical position, thus using learned top-down cognitive processes to reduce the inherent ambiguity of the musical surface.*" (Merchant et al., 2015)

[40] Let's forget about reggae for now!

[41] Jazz and rock music attribute far greater importance to the backbeat though, which, much to the irritation of musicians, is not obvious to all concertgoers who insist on clapping on the 'one and three' (http://www.ethanhein.com/wp/2013/the-backbeat-a-literature-review/). A very entertaining video shows a live concert where pianist Harry Connick Jr. uses rhythmic displacement to insert a bar of 5/4 in a 4/4 piece of music. This intentionally shifts the clapping audience from 'one and three' to the 'two and four' backbeat, much to the relief and gratitude of the drummer! (https://www.youtube.com/watch?v=--qv9SI6vws)

Other researchers also point to the salience of the kick and snare drums in the inference of downbeats and upbeats in rhythm perception and, consequently, its computational analysis (Zils et al., 2002; Panteli et al., 2014; Gómez-Marín et al., 2017). Leaving aside *why* exactly this is the case [42], it is clear that not all drum timbres are created equal, and generative systems such as ours should incorporate some cognisance of that important fact.

To remedy this, a weighting scheme was used to assign different degrees of importance to the separate timbre streams when operating in parallel mode. The exact formulation weights are not important, just that they are weighted sufficiently to bias the fitness function as desired, so for example we weigh the 4 streams successively from bottom (kick) to top (perc or cymbal) with the set $W = \{2, 1, 0.5, 0.25\}$. These weights are also parametric to account for different timbres assigned to the different streams.

#### 3.4.5.2 Syncopation

> "*Syncopation is the spice of rhythm*" (Toussaint, 2013)

Just as musical ideas blossom through the balance between repetition and variation of a motif or loop, rhythms are made infinitely interesting by resolving the tension arising from anticipation and reinforcement of the beat and the deliberate disruption of that expectation. The *Harvard Dictionary of Music* defines syncopation simply as:

> "*Syncopation: a momentary contradiction of the prevailing meter or pulse*" (Apel, 1969)

Gómez et al. (2005) expand on this definition:

> *"Moreover, syncopation can be materialized either as a momentary change of the primary character of the meter or as a contradiction between strong and weak beats against other parts of the musical texture whose metrical context is fixed"*

Up until now, our algorithm has only been considering the similarity of generated rhythms in relation to the target, and has no facility to gauge the level of syncopation present in the target or the generated rhythms. It would be a useful parameter of the GA to allow the user to increase bias for selecting candidates based on relative syncopation of strong and weak beats in a particular track, and this was the next feature that was implemented.

[42] Hove et al. (2014) proposes that "its our superior time perception for lower musical pitch".

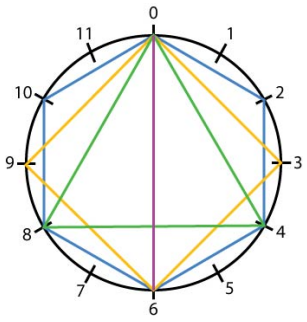



**Figure 3.23:** Toussaint's Off-beatness measure. (Image from McGill).

Eigenfeldt & Pasquier (2013) incorporate a rudimental quantifier for level of syncopation into their hybrid rhythm algorithm by simply counting the number of onsets that do not fall on the expected strong beats in the metre (4/4).

To expand our system to include knowledge of syncopation, we turned once again to perceptual literature to see if there were any established analytic methods that prove more grounded in observation rather than blindly devising our own. Gómez et al. (2005) sum up 4 such measures including:

- The Rhythmic Oddity Property (Arom, 2004) is not really a measure, it is simply a boolean value indicating whether there exists two onsets in a pattern that divide the set equally or not.
- Toussaint's *Off-Beatness* (Toussaint, 2004b) finds those geometric shapes that divides an $N$ length rhythm necklace evenly (Figure 3.23). These are the on-beats. The number of onsets that lie outside of these geometric entities give the degree of off-beatness.
- Keith's Measure (Keith, 1991) defines syncopation in terms of three events, *hesitation* (a note starts on a beat but ends off a beat), *anticipation* (a note starts off a beat but ends on a beat) and *syncopation* as a combination of of hesitation and anticipation within a rhythm. Weights are assigned to these three event categories (like our own timbre weightings used in Section 3.4.5.1, these are subjective by the author) and the weighted sum of instances of these constitute the measure of syncopation in the measure.

Gómez et al. (2005) criticises Keith's measure in its arbitrary assignment of weights but, more crucially, its restriction to binary metres (explained in depth the article). They offer a new measure to address these limitations and coalesce some elements inherent in Toussaint's off-beatness. The key tenet of the Weighted Note-to-Beat Distance (WNBD) is that "notes are supposed to end where the next note starts". Onsets that land off strong beats are given the smaller weighting and weighting increases to

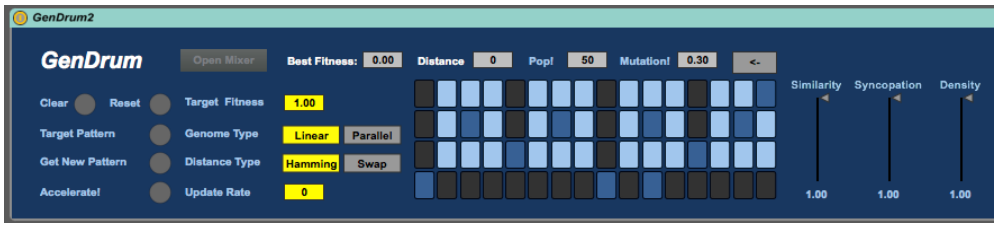



**Figure 3.24:** GenDrum with Expanded Feature Parameter Control on the right

a maximum as notes cross over strong beats - which they posit gives the strongest sensation of syncopation.

> *"the closer a note is to a strong beat, the more syncopated it sounds; and second, the syncopation effect is stronger when a note crosses over only one strong beat of the meter."* (Gómez et al., 2007)

While Song et al. (2013) has some reservations of this model and we could find no instances of surveys that evaluated its correlation with human listeners, it does seem a reasonable distillation of syncopation for our purposes, compared to the other formulations. WNBD is distance-based model that can be easily integrated into our fitness function with comparative ease. Based on a reference implementation offered in Synpy (Song et al., 2015), a Python environment for syncopation modelling, we expanded the genetic algorithm to include the WNBD in its fitness function.

#### 3.4.5.3 Density

The density of a rhythm is an easy feature to comprehend, Wiggins et al. (2012) defines it as the "mean number of events per tactus beat". We simply added another metric for the fitness function to consider, given by the average number of onsets per pattern (Eq. 3.5), where $x_i$ is onset number $i$ and $N$ is the total number of onsets.

$$d(x) = \frac{\sum_{i=1}^{N} x_i}{N} \quad \text{if } x_i = 1 \tag{3.5}$$

Having influence over density in a generative drum machine is a desirable control to have and has actually been shown to be a useful descriptor in modelling rhythmic similarity, specifically in dance music signals (Panteli et al., 2014). Within the GiantSteps project the Agnostic Density Transformer (Jordà et al., 2016) is a generative drumming system that is focussed on density control of individual drum sounds based on corpus analysis, and has since been expanded into a mobile app by Reactable Systems[43].

[43] https://reactable.com/snap/

The weighted linear combination of these three features coupled with the new parallel timbre weighting scheme comprise the new fitness function for the genetic algorithm. Since these additional features represent a considerable departure from the general string analytics offered by the SimpleGA object, we decided to encapsulate it in a separate, specialised object called RhythmGA. Figure 3.24 shows the revised interface for GenDrum that uses the new GA, along with the new similarity, syncopation and density sliders that adjust the relevant parametric weights.

### 3.4.6 Conclusions and Closing Remarks

In this chapter we presented a way of generating drum patterns automatically using genetic algorithms. Rather than relying on the commonly used interactive fitness function, our method was shown to use the notion of a "target pattern", with fitness derived from the distance of the generated patterns to the target. This was devised to address the way dance music composers derive tracks from a bottom-up approach that begins with a small loop or idea that is iteratively expanded through repetition and variation of its key features.

Following a review of various approaches to establishing the distance between rhythms as present in the literature, we demonstrated the implementation and incorporation of two such measures - the Hamming distance and the directed-swap distance - into a genetic algorithm instrument for polyphonic drum pattern creation. We believe this work, and the accompanying articles, constitute the first integration of perceptual symbolic similarity research in rhythmic pattern generation with genetic algorithms.

To evaluate the research carried out, we conducted a listening survey to determine participants' reaction to the generated patterns in terms of the similarity and "interestingness" related to the target pattern. It was shown that the distance ,and thus fitness, correlates strongly with user perception in terms of similarity. Crucially we showed that the Hamming distance alone is a worthwhile quantifier of rhythmic similarity even in the case of polyphonic patterns. Our approach to gauging users' response to the concept of "interestingness", however, needs review and presents a complex challenge for future work.

Based on the feedback ascertained during the evaluation, a second iteration of the genetic algorithm along with the accompanying instrument was developed that assimilated an expanded feature set as well as enhanced parametric control for the user. By adjusting these parameters, relative weightings of rhythmic attributes such as similarity, syncopation and densities can bias the fitness function as appropriate. While our own informal appraisal of these additions was favourable, a more rigorous evaluation and listening survey was not conducted. This was due to a shift of focus away from symbolic algorithm composition towards approaches involving content-based, MIR-driven synthesis of real audio.

The reasons for this change of tack were multifold. Firstly, while genetic algorithms coupled with an appropriately formulated fitness function demonstrably provide a le-

gitimate mechanism for generating rhythms for dance music, methods proposed by our colleagues that invoke specialised metrics (Gómez-Marín et al., 2015) or knowledge trained on large corpora representing style (Jordà et al., 2016) are perhaps more suited. This has been confirmed in a collaborative evaluation conducted by ourselves along with others in the GiantSteps consortium (Vogl et al., 2016). Our GA did not fare better in various listening tests against neural networks and a database-driven method both trained on human derived patterns from the Maschine software and hardware environment (and understandably so, as the two methods indisputably embed more knowledge). However it did return more "interesting" results and was rated higher in its facility for creating more unique fill patterns.

But most of all, based on an overall critique of the myriad approaches to generating rhythms using symbolic conventions and similarity analysis (including our own experiences building systems that utilise them), we sensed that there are some subtleties inherent in real recordings and performances of rhythms that are not encapsulated via these means. Namely we could identify:

- The notion of verticality or interdependence between concurrent elements at discrete points in time that are not completely captured by concepts such as metrical hierarchy and weighting.
- Stress and accents that affect the dynamics of onsets in a pattern, not captured by the pervasion of binary representations that exists in literature and associated realised systems. This perhaps can be addressed by analysing and estimating rhythms in terms of real or floating point numerical forms.
- Related to stress and accents there are other difficult phenomena of human factors in music that include groove and swing. These are notoriously difficult facets to define and quantify by musicologists, and even harder to estimate and recreate using computational means. The need to replicate human qualities of time within conventional music production systems is clear however. Most DAWs at least include some ability to adjust metrical positioning of note information according to some percentage factor ostensibly tantamount to groove.
- The timbral qualities - or the sounds and processing used to derive the patterns - we contend have a profound impact on the perceptual response and aesthetic impression of the loop. Simply assembling rhythms according to perceptual similarity measures using arbitrary sounds (all too frequently coarse MIDI approximations) does not always produce the desired results, at least in generative attempts at dance music.

All these factors were carefully considered and it became more apparent that perhaps the symbolic domain was not sufficient for our goals. The focus of the thesis turns to

investigating whether these methods can be refined, adapted and extended to systems that generate rhythms using signals and thus connecting time with timbre.or

# Chapter 4

# State of the Art in Sampling-Based Composition

As we move away from the symbolic leanings of the previous chapter and head towards audio content-based approaches to music generation, we pause once again from the deeper technical discourse to establish some context, both artistic and historic, before before providing a thorough definition of a concatenative synthesis - a form of synthesis that is rooted deeply in audio sampling - and its key works from the literature and community.

I use the term "audio content-based" here interchangeably here with sampling for good reason. The term actually stems from MIR parlance, used to dermarcate research that deals directly with audio data and signals from those that analyse notated information and metadata (Veltkamp et al., 2008; Casey et al., 2008). It also enforces the idea that we are working with elements extracted from found and realised objects and artefacts, which we will soon discover has a long and illustrious history as a creative aesthetic.

## 4.1 Artistic Context

Historically, reusing existing material for the purposes of creating new works has been a widely practised technique in all branches of creative arts. The manifestations of these expressions can be wholly original and compelling or otherwise derivative, uninspiring and potentially copyright infringing depending on myriad factors, including the domain of the work, the scale of the reuse, and its cultural context.

In the visual arts, reusing or adapting existing material is most immediately understood in the use of collage, where existing works or parts thereof are assembled to create new artworks. Cubist artists such as George Braque and Pablo Picasso extensively referenced, appropriated and reinterpreted their own works and the works of others (Figure 4.1), as well as common found objects from their surroundings (Greenberg, 1971). Collage would later serve as direct inspiration for bricolage, reflecting


"Ma Jolie" 1911 Picasso

"Man with Guitar" 1911 Braque

**Figure 4.1:** Cross-referential Cubist Works by Picasso and Braque.

wider postmodernist trends towards deconstructionism, self-referentiality, and revisionism that include the practice of parody and pastiche (Lochhead & Auner, 2002).

### 4.1.1 Musique Concrète and Elektronische Musik

In music and the sonic arts, its natural corollary came in the form of musique concrète (Holmes, 2008), a movement of composition stemming from the experiments of Pierre Schaeffer and, later, Pierre Henry at the studios of Radiodiffusion-Télévision Française in Paris during 1940s and 1950s (Battier, 2007). In contrast to the artificially and electronically generated *elektronische musik* spearheaded by Karlheinz Stockhausen at the WDR in Cologne (Collins & D'Escrivan, 2017; Bates, 2009), the French composers sought to conceive their works from existing recorded sound, including environmental sources like trains and speech. Seemingly unrelated and non-musical sounds were organised in such a way that the listener discovered the latent musical qualities and structure they inherently carry.

It is important to note that, in music composition, general appropriation of work predates these electronic advancements of technology. In Western art music, for example, composers like Bela Bartók – himself a musicologist - have often turned to folk music for its melodies and dance music styles (Bartók, 1993), while others such as Debussy (Tamagawa, 1988) became enchanted by music from other cultures such as Javanese Gamelan, studying its form and incorporating the ideas into new pieces. Quotations,

or direct lifting of melodies from other composers' works, frequently crossing into classical, are also quite common, and has become a frequent, and often with the intention of being amusing, rudiment in jazz music practice (Mangani et al., 2006)[44]. For more on appropriation and quotation music a good reference is compiled by Metzer (2003).

### 4.1.2 Granular Synthesis of Microsound

Granular synthesis is composition at the microscale of sound - or to use Curtis Roads' term *microsound* (Roads, 2004). It is a molecular viewpoint of music that Thomson (2004) maintains finally promises the composer the possibility for "total composition", thereby dictating the parameters of everything from higher-order forms and structures all the way down to infinitesimal sonic minutiae.

It is generally agreed that the physicist Dennis Gabor first proposed the first quantum theory of sound that suggested "large, complex sound events are composed of simple acoustic particles, or sonic grains' (Miranda, 1995). This was later reiterated and expanded on by Xenakis in *Formalized Music* (Bradshaw & Xenakis, 1973), who laboriously arranged small snippets of tape material to produce *Analogique B* (for 9 string instruments and tape, 1959) (Robindore & Xenakis, 1996).

Curtis Roads lays claim to developing the first digital implementation of granular synthesis (Roads, 1988), during the latter half of the 1970s (on punchcards!) and has composed with it and written about it capaciously ever since (Roads, 1985, 1991, 1996, 2004) Barry Truax successfully achieved real-time granular synthesis during the 1980s with the composition of *Riverrun* (Truax, 1988), using Poisson distributions and tendency masks to control the limits of typical granular parameters like frequency range, grain duration, modulation index, and delay.

Practically speaking, granular synthesis is, as Roads (1988) suggests, a type of additive synthesis. Rather than mixing pure signals generated by oscillators to generate complex sounds, small 'grains' (within the realm of 20-200ms) of existing sounds[45] are superimposed to create dense amalgamations or 'clouds' of sound grains (Collins & D'Escrivan, 2017; Thomson, 2004). Nowadays granular synthesis is widespread in computer music practice and most music programming environments like Pd, Max and Reaktor, and even commercial synthesisers and DAWs, provide extensive tools and tutorials for its pursuit. Listening to the works of glitch artists like Kim Cascone[46](Latartara, 2010), or the more meditative washes of sound from drone and

[44] Appel (2002) recounts an anecdote of Stravinsky attending a Charlie Parker gig at his residency in the Birdland club in New York and "[roaring] with delight, pounding his glass on the table" on hearing a snippet of the opening of his Firebird Suite from the saxophone of the bebop pioneer.

[45] Though these grains can, and often do, come from signal generators.

[46] The ideals of which are put forth in his influential manifesto "The aesthetics of failure: "Post-digital" tendencies in contemporary computer music", the most cited article in The Computer Music Journal (Cascone, 2000)

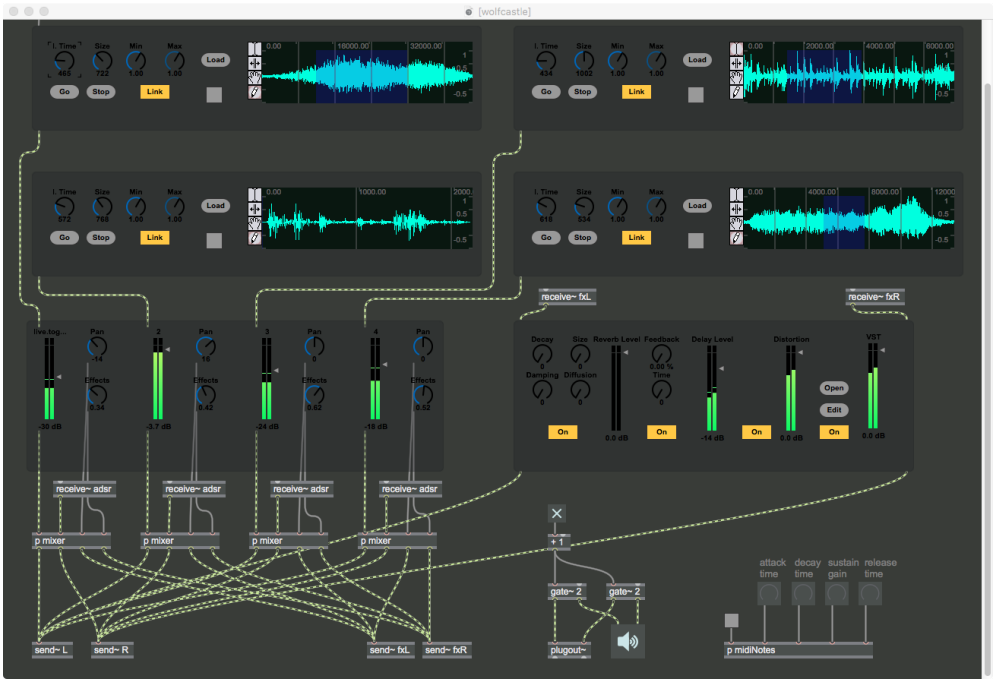



**Figure 4.2:** Screenshot of my four-channel granular synthesiser for Max[50]

noise luminaries Tim Hecker or Fennesz, reveals an instantly recognisable application of the technique.

I have used granular synthesis (Figure 4.2) frequently in my own music for generating large indeterminate textural material from smaller fragments of composed musical ideas or interesting samples I have stumbled on. Recently this included a work for violin, distortion and granular synthesis presented as part of the concert programme[47] of the Sound and Music Computing (SMC) conference[48] as well as a quadrophonic soundscape that was installed at the Sonic Environments/Australasian Computer Music Conference (ACMC) satellite event that took place as part of New Interfaces for Musical Expression (NIME) in 2016[49].

### 4.1.3 Bring the Noise - Hip Hop Culture and the Birth of the Breakbeat

In Chapter 2 we briefly highlighted the contribution of DJ Kool Herc, who used two turntables to produce an extended looping, collage of drum sections (the breakbeats) from identical vinyl records (Forman & Neal, 2004). Kool Herc's contemporaries, Afrika Bambaataa and Grandmaster Flash observed his discovery and augmented it

[47] Incidentally, Barry Truax premiered a new work during the same conference
[48] http://www.maynoothuniversity.ie/smc15/concert2.html
[49] http://www.sonicenvironments.org/program.html

with more rudiments such as backspin, punch phrasing and, of course, record scratching which would widen the idiomatic vocabulary of *turntablism* within the burgeoning subculture of hip-hop (Smith, 2000).

Vinyl would not reign king for too long however. The modern notion of sampling, at least in non-electroacoustic or contemporary composition circles, stems from the advent of the digital sampler and its eventual explosion of adaptation in hip-hop. The arrival of easy to use, self-contained hardware systems like the Akai Music Production Controller (MPC) offered the rhythmic sequencing facilities of the Roland TR range but with the groundbreaking ability to import one's own sounds (or more crucially, other artists' sounds), and all this with a reasonable price tag attached (Rodgers, 2003).

Hip-hop producers quickly latched onto the possibilities of these new tools. Artists such as Public Enemy and Beastie Boys, on albums like *It Takes a Nation of Millions to Hold Us Back* and *Paul's Boutique* respectively, painstakingly assembled bewildering permutations of musical samples, soundbites and miscellaneous other recorded materials that sought to supplant the many cultural references that permeated their lyrics (Sewell, 2013, 2014).

As was also identified in Chapter 2, the influence of hip-hop production would later inform the sample-heavy arrangements of jungle and drum and bass, in particular with its exhaustive re-rendering of the infamous "Amen Break" and James Brown's "Funky Drummer". In fact these breaks had already been dissected previously in hip-hop, long before UK producers got their digital surgical scalpels to them. The Amen Break appears on NWA's landmark gangster rap album *Straight Outta Compton* and James Brown's "Funky Drummer" shows up on tracks by Run DMC and LL Cool J (Frane, 2017), leading Oliver (2015) to observe that "it is [Funky Drummer] sampled and recontextualized so extensively as to achieve near ubiquity in a wide range of popular music genres".

Jungle and drum 'n' bass was dealt with sufficiently in Chapter 2 so it is hopefully not necessary to restate its relevance and impact on sample-based composition here. It is prudent to point out though that, since we are edging towards computational models and processes that automate audio content-based composition, that there is already a strong body of work that specifically addresses computational aspects of breakbeat analysis and its sequent generative synthesis. Nick Collins for example, has dedicated much of his academic dissemination to algorithms that can replicate the distinctive stuttering and glitching tropes that pervade in jungle, drum & bass and post jungle emergent styles like breakcore (Collins, 2001, 2002, 2006b). Hockman has also devoted significant output to MIR-oriented analysis of timbral and rhythmic aspects unique to these styles (Hockman, 2007; Hockman et al., 2012; Hockman & Davies, 2015).

### 4.1.4 Plunderphonics and Metamusic

"*I quote from others the better to express myself*"

(Michel de Montaigne)

I think it's rather appropriate that the quote above I did not discover myself, instead I lifted it from an article dealing with the appropriation and quotation of other works in music (Holm-Hudson, 1997). The article is mostly concerned with the implications arising from the work of John Oswald, an artist who has gained much notoriety in his career by directly challenging musical copyright law for artistic gain. As hip-hop producers were trailblazing a new sound through intimate operation of the digital sampler, the inevitable direction was the bootlegging of other artists' music for creative repurposing, thus sparking the huge copyright debate. Record companies and legal representatives struggled to react to the huge paradigm shift in cultural production whose commercialisation was no longer under their total jurisdiction. The ethics of all this are outside the scope of this thesis obviously, but at least a few legal commentators at the time recognised that perhaps the long established laws governing creative ownership were no longer fit for purpose:

> "*...copyright doctrine incorporates notions of Romantic authorship that assume independent and autonomous authorship and even genius in the creation of original musical works. This individualistic and autonomous vision of musical authorship, which is central to copyright law, has deemphasized the importance and continuity of musical borrowing practices generally*" (Arewa, 1979)

Returning to John Oswald, who, capitalising on the polemics and its scope for creative interpretation, coined the name "plunderphonics" and set out his artistic intentions in a suitably subtitled 1985[51] essay "Plunderphonics, or Audio Piracy as a Compositional Prerogative" (Oswald, 1985). The Canadian composer and media artist, inspired by the Dadaist cut-up literary technique practised by beat writer William S. Burroughs as well as sound collages by James Tenney (Cox & Warner, 2004), began to use tape-splicing techniques to create deliberately recognisable montages of other musical recordings, particularly pop music such as Michael Jackson or Led Zeppelin. If granular synthesis represents, as Thomson (2004) indeed suggests, a path to "total composition", then Oswald's modus operandi for composition is, as Holm-Hudson (1997) describes it, the "total importation" of pieces involving "reinterpretation or rehearing of existing recordings" (Cutler, 1994).

[51] Interesting then, that around this time Truax and Roads were still getting to grips with granular synthesis.

Nowadays artists such as Girltalk and DJ Danger Mouse[52][53] create extremely complex and multi-referential intersections of popular music harnessing the powerful beat matching and synchronisation capabilities of the modern DAW (Humphrey et al., 2013). The style has become known widely as "mashups" or "bastard pop" (McGranahan, 2010) and straddles the fine line of what constitutes and can be protected under the auspices of fair usage (Mongillo, 2009), the very same doctrine that allows us to include images from other sources in this thesis in good faith.

Many computational works have emerged that explicitly mention and address the musical goals of mashup (Tokui, 2008; Davies et al., 2013, 2014a,b; Lee et al., 2015; Smith et al., 2015; Meroño-Peñuela et al., 2017), and we shall see that the direct consequence of concatenative synthesis can facilitate the realisation of elements of mashup.

## 4.2 Concatenative Synthesis

Recent associated research efforts in computer music, signal processing and music information retrieval afford us the opportunity to develop automated and intelligent systems that apply the aforementioned aesthetic of sampling and artistic reuse.

The term "concatenative synthesis" (or musical mosaicing (Zils & Pachet, 2001)) has been extensively used to describe musical systems that create sound by recycling existing ones automatically, according to some well-defined set of criteria and algorithmic procedures (Schwarz, 2000). Concatenative synthesis can be considered the natural heir or big brother to granular synthesis. With concatenative synthesis, the grains of sound become "units" and are more related to musical scales of length, such as notes and phrases. Most importantly, descriptive information is attached to these units of sound: crucial features that allow the temporal, spectral, harmonic and timbral characteristics of the sound to determine the sequencing of final output.

The criteria governing the procedure of concatenative synthesis is usually applied in relation to some reference sound or target that the composer wishes to emulate, and with this in mind, Bernardes gives a more thorough definition of the discipline:

> "*[Concatenative Synthesis] is a technique for synthesizing sounds by concatenating short audio segments (called units). It relies on a large database of segmented and descriptor-analyzed sound snippets (called corpus) to assemble a given target phrase by selecting the units that best*

[52] Danger Mouse availed of the tendency in rap music to realise acapella versions of records for the purposes of remixing to create *The Grey Album*: a well-received mashup of The Beatles' *White Album* with Jay Z's *The Black Album* (Gunderson, 2004)

[53] And much prior to that, experimental ensemble Negativland faced the legal backlash of record labels when they released the EP *U2*, which contained misleading artwork and unauthorised sampling of the rock group, but ironically catapulted them into public notoriety (Herman & Sloop, 1998; Lysloff & Gay Jr, 2003).

> *match the target specification according to a distance measure in the descriptors*" (Bernardes et al., 2013)

In Chapter 3 we described symbolic methods that could take a seed or target rhythm pattern typical of dance music and replicate and variate it through similarity knowledge embedded in the fitness function of a GA. With concatenative synthesis, we have a computational framework that can extend this notion of looping repetition and variation to audio.

Just as granular synthesis invokes Heisenberg's Uncertainty Principle to blur the lines of time and frequency at the microscale of sound (Truax, 2005), we can view concatenative synthesis as a macroscale manipulation of the dichotomy between originality and theft determined by the scale of the unit - larger units of scale are more fully-formed but are more likely to be recognised as being lifted from other sources, while smaller units eventually approach granular levels of abstraction.

## 4.3 Survey of Concatenative Synthesis Systems

Other authors have previously provided insightful summaries of research trends in concatenative synthesis. In particular we suggest those by Schwarz (2005, 2006) and Sturm (2006). These surveys are over 10 years old however, so we offer here a more recent compendium of state of the art systems as we see them, based on our investigations of previous publications up until now.

### 4.3.1 Unit Selection Synthesis of Speech

Before music, concatenative synthesis has enjoyed successful application in the area of speech processing, specifically in the synthesis of Text-to-Speech (TTS).

Hunt & Black (1996) were the first to propose unit selection synthesis using Hidden Markov Model (HMM)s as a means for combining phoneme units extracted from existing audio recordings of speech. Conkie (1999) remarks that "for unit selection synthesis, typically phone-based, it is possible to produce sentences that sound surprisingly natural and intelligible from a large database'. Since then many researchers (Conkie, 1999; Balestri et al., 1999; Schröder, 2001; Tihelka et al., 2010; Lakkavalli et al., 2010) and software systems (Black & Taylor, 1994; Beutnagel et al., 1999; Young et al., 2002) have cemented the popularity of the technique.

We will treat unit selection and HMMs more thoroughly in Chapter 5 but essentially HMMs extend Markov chains (Section 3.2.1.2) by assuming that the states in the state machine are now hidden but can output observable symbols. The Viterbi algorithm can return the most probable sequence of hidden states given a particular sequence of observed symbols. In concatenative synthesis, this maximum probabilistic model is inverted to facilitate minimal cost computations thereby encoding the dissimilarity of unit sounds.

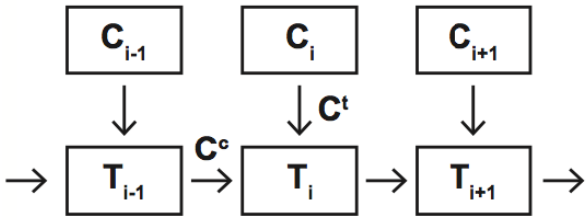



**Figure 4.3:** Target ($C^t$) and concatenation ($C^c$) costs in a TTS HMM compared.

Formulating the problem in terms of a HMM enables us to capture two important contributors to a successful synthesis via the *target cost* and the *concatenation cost* (Figure 4.3). The target cost estimates the penalty of matching a corpus unit with the current target unit based on a target specification that stipulates factors like fundamental frequency or pitch, duration, position in the syllable and surrounding context of phonemes. The concatenation cost then estimates the quality of the join between two potential corpus units if they were to be selected, hence allowing a means of penalising selections that introduce unwanted jumps in energy and pitch (for example) that could lead to an unnatural sound. The Viterbi algorithm efficiently searches the the state space of database units and outputs indices corresponding to the optimal state sequence for the target, based on a consolidated linear combination of the aforementioned costs (Black & Campbell, 1995; Hunt & Black, 1996).

### 4.3.2 Musical Concatenative Synthesis

Schwarz directly applied Hunt's HMM approach for musical purposes in his Caterpillar (Schwarz, 2003). He notes, however, that the HMM approach can be quite rigid for musical purposes since it produces one single optimised sequence without the ability to manipulate the individual units. To address these limitations, he reformulates the task into a constraint satisfaction problem which offers more flexibilities for interaction. A constraint satisfaction problem models a problem as a set of variables, values and a set of constraints that allows us to identify which combinations of variables and values are violations of those constraints, thus allowing us to quickly reduce large portions of the search space (Russell & Norvig, 2002).

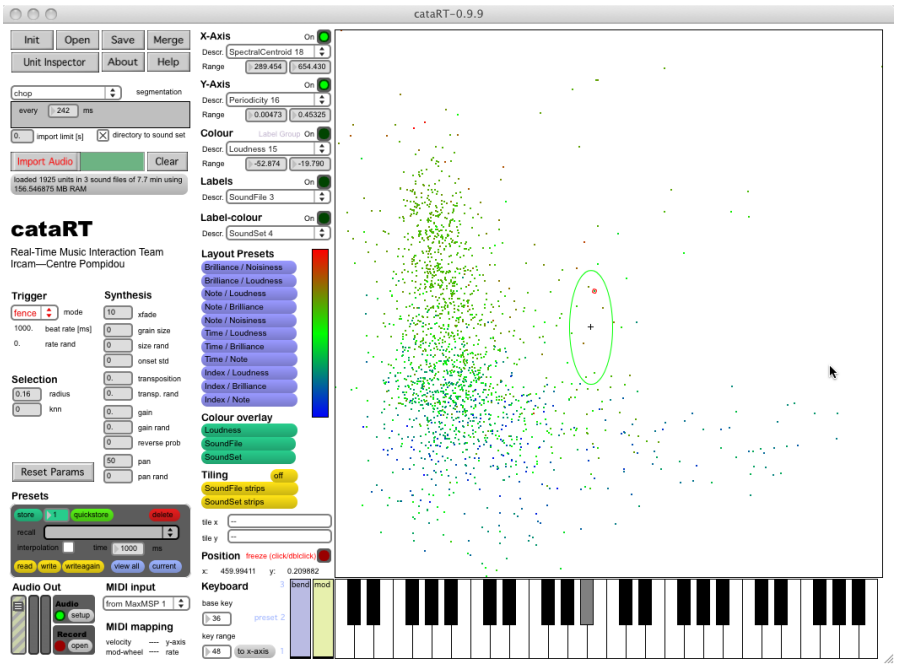
Constraint satisfaction for concatenative synthesis was first introduced previously by Zils & Pachet (2001), in what they describe as musical mosaicking or, to use their portmanteau, musaicing. They define two categories of constraints: segment constraints and sequence constraints. Segment constraints control aspects of individual units (much like the target cost in a HMM-like system) based on their descriptor values. Sequence constraints apply globally, and affect aspects of time, continuity and overall distributions of units. The constraints can be applied manually by the user, or learned by modelling a target. The musically tailored "Adaptive Search" algorithm

performs a heuristic search to minimise the total global cost generated by the constraint problem. One immediate advantage of this approach over the HMM is the ability to run the algorithm several times to generate alternative sequences, whereas the Viterbi process always outputs the most optimal solution.

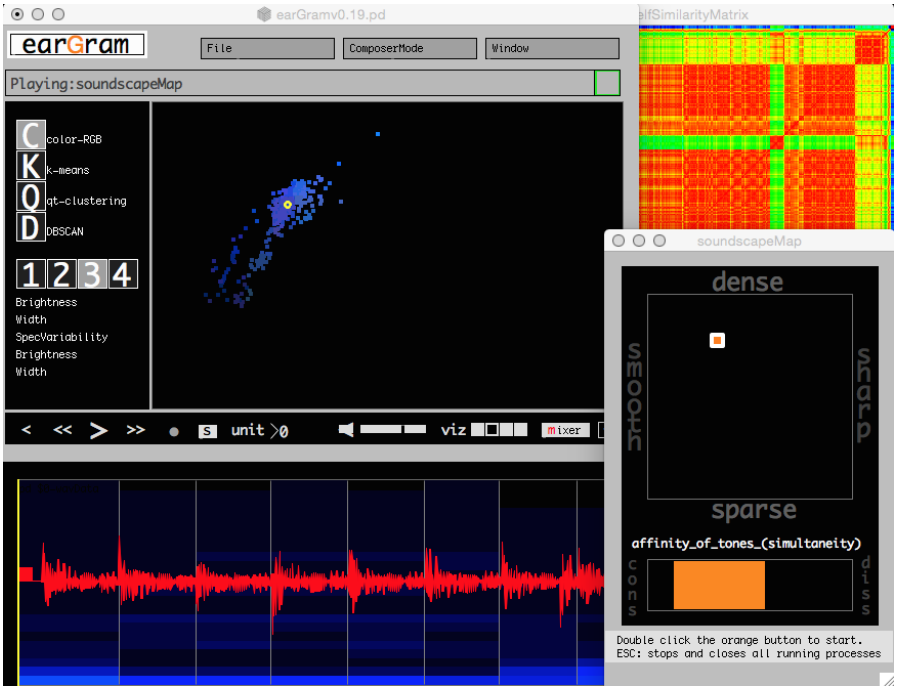
A simpler approach is presented in MatConcat (Sturm, 2004), using feature vectors comprising six descriptors and computing similarity metrics between target units and corpus units. Built for the Matlab scientific computing environment, the interface is quite involved, and the user has control over minute features such as descriptor tolerance ranges, relative descriptor weightings as well as window types and hop sizes of output transformations. On his website are short compositions generated by the author using excerpts from a Mahler symphony as a target, and resynthesised using various unrelated sound sets, for instance pop vocals, found sounds and solo instrumental recordings from saxophone and trumpet.

As concatenative synthesis methods matured, user modalities of interaction and control became more elaborate and real-time operations were introduced. One of the most compelling features of many concatenative systems is the concept of the interactive timbre space. With the release of CataRT (Schwarz et al., 2006) the authors provided an interface (Figure 4.4a) that arranges the units in an interactive two-dimensional timbre space. The arrangement of these units is made according to a user-selectable descriptor on each axis. Instead of using a target sound file to inform the concatenation procedure, the user's mouse cursor becomes the target. Sounds that are within a certain range of the mouse cursor are sequenced according to some triggering options (e.g. one-shot or looped) and, most crucially, with real-time output.

Bernardes takes inspiration from CataRT and also from Tristan Jehan's Skeleton (Jehan, 2005) to build his EarGram system for the Pure Data environment (Bernardes et al., 2013). Built on top of William Brent's excellent feature extraction library timbreID (Brent, 2010), it adds a host of interesting features for visualisation and classification. For instance, as well as the familiar waveform representation and previously described 2D timbre representation (Figure 4.4b) (with various clustering modes and dimensionality reduction implementations), there are similarity matrices that show the temporal relations in the corpus over time. Some unique playback and sequencing modes also exist, such as the infiniteMode, which generates endless playback of sequences, and the soundscapeMap, which features an additional 2D control of parameters pertaining to sound scene design. Another system that adapts a 2D timbre space is Christian Frisson's AudioGarden (Frisson et al., 2010), which offers two unique mapping procedures. The first of which, "Disc" mode, places units by assigning the length of the audio file to the radius of the unit from the centre, with the angle of rotation corresponding to a principal component of timbre (MFCCs). In the other Flower mode, a point of the sound is positioned in the space according to the average MFCC coefficients of the entire sound file. Segments of the particular sound are arranged in chronological fashion around this centre point.

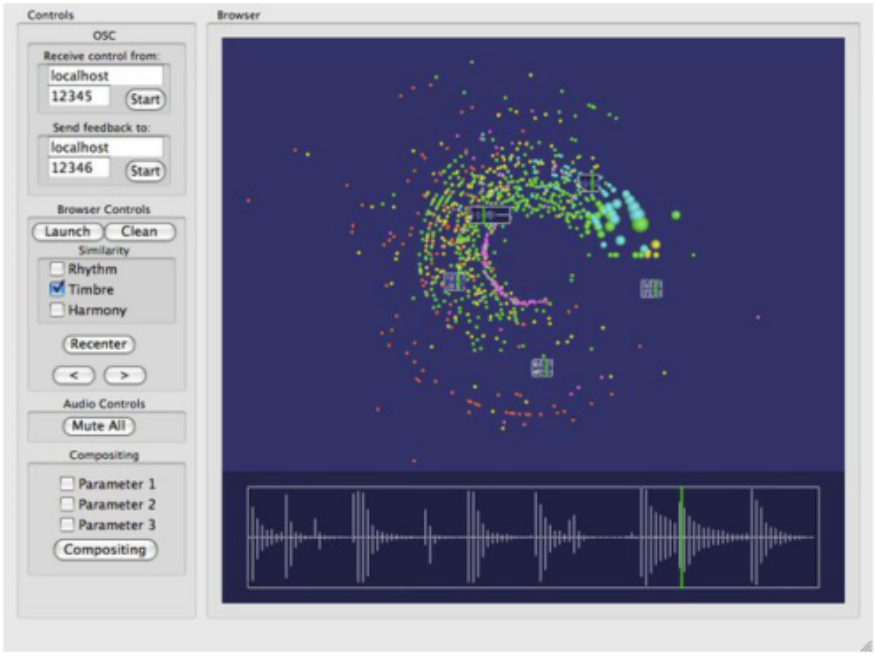



**(a)**

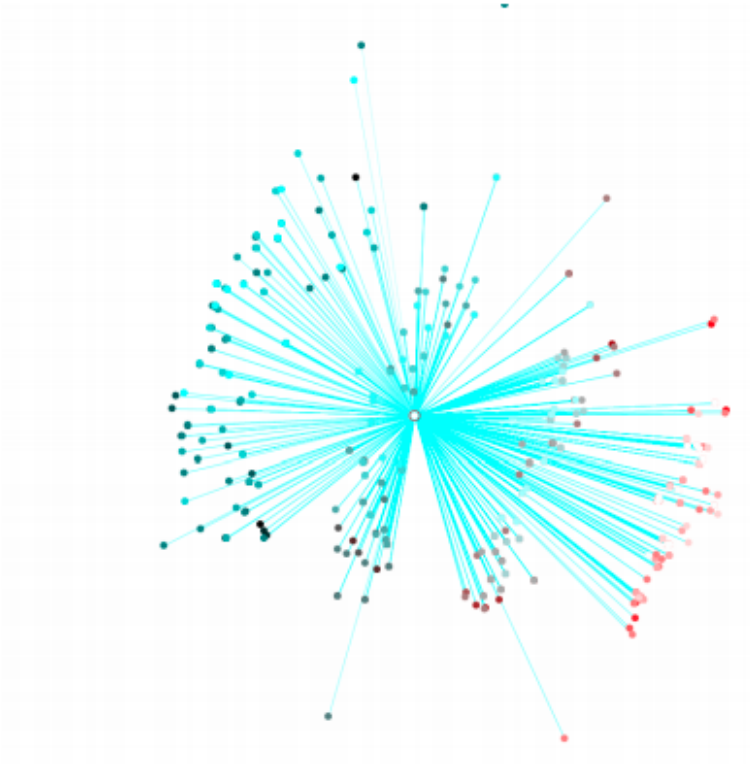



**(b)**

**Figure 4.4:** (a) CataRT User Interface (Image from IRCAM), (b) earGram User Interface (Image from SMC INESCTEC), both showing a clear realisation of an explorable 2D timbre space.




**(a)**



**(b)**

**Figure 4.5:** Two different visualisation of timbre: (a) AudioGarden "Disc" Visualisation (Image from Frisson et al. (2010)), (b) AudioGuide Clustering Visualisation around Target (Image from Schwarz & Hackbarth (2012)).

There have been some concatenative systems tailored specifically with rhythmic purposes in mind. Xiang proposes Granuloop (Xiang, 2002) for automatically rearranging segments of 4 different drum loops into a 32 step sequence. Segmentation is done manually, without the aid of an automatic onset detector, using the Recycle sample editing software. Segmented sounds are compared using the inner product of the normalised frequency spectrum, supplemented with the weighted energy. These values become weights for a Markov-style probability transition matrix. Implemented in Pd, the user interacts by moving a joystick, in a 2D space, this then affects the overall probability weightings determining which loop segments are chosen. The system presents an interesting approach but is let down by its lack of online analysis. Ringomatic (Aucouturier & Pachet, 2005) is a real-time agent specifically tailored for combining drum tracks, expanding on many of the constraint-based ideas from their prior musaicing experiments. They applied the system to real-time performance following symbolic feature data extracted from a human MIDI keyboard player. They cite, as an example, that a predominance of lower register notes in the keyboard performance applies an inverse constraint that creates complementary contrast by specifying that high-frequency heavy cymbal sounds should be concatenated.

As demonstrated in EarGram, concatenative synthesis has been considered useful in sound design tasks, allowing the sound designer to build rich and complex textures and environments that can be transformed in many ways both temporally and timbrally. Also in connection to sound design, Cardle et al. (2003) report on their Directed Sound Synthesis software as a means of providing sound designers and multimedia producers a method of automatically reusing and synthesising sound scenes in video. Users select one or more regions of an existing audio track, and can draw probability curves on the timeline to influence resynthesis of these regions (one curve per region) elsewhere. Soundscapes by Hoskinson & Pai (2001), in a nod to granular synthesis, refers to their segments as ”natural grains” and seek to synthesise endless streams of soundscapes. The selection scheme by which segments are chosen is based on a representation of each segment as a transition state in a Markov chain. Its interface features knobs and sliders for interactively controlling gain and parameters of multiple samples. To evaluate the platform they conducted an additional study (Hoskinson & Pai, 2007) to reveal whether listening subjects found the concatenated sequences convincing compared to genuinely recorded soundscapes.

More specific and applied use cases of concatenative synthesis include Hackbarth (2011), who works intimately with the possibilities of concatenative synthesis in large scale music composition. He has worked with Schwarz to provide an alternative interface for exploring variations based on a force-directed graph (Figure 4.5b) to give a visual impression of the perceptual distances units in relation to a central target, not unlike the feedback we provided in our GenDrum instrument (see Chapter 3). O’Connell describes a graphical system for Pd that demonstrates the use of higher level perceptual concepts like mood (happy vs sad) for informing selection in audio mosaics (O’Connell, 2011).

Commercial implementations also exist for concatenative synthesis. Of particular note is Loopmash[54], a software plugin and mobile application for automatically creating mashups from existing looped content. The interface consists of a number of tracks in a timeline arrangement. One track is set as a master, and slices in the master are replaced with matching slices from the other slave tracks. Users interact by manipulating "similarity gain" sliders that control the influence of each track in the slice selection algorithm. Other applications exist more as MIDI sampler systems attempting to model the performance qualities of natural sources such as orchestral ensembles[55] or the human voice[56].

There are many other concatenative systems that are too numerous to discuss in detail here. Table 4.1 compiles a summary of all the systems we came across in our research, with remarks on interaction and visualisation features, support for rhythm, and whether any user evaluation was carried out.

[54] https://www.steinberg.net/en/products/mobile_apps/loopmash.html
[55] https://www.synful.com/
[56] https://www.vocaloid.com/en

| Author (Year) | Name | Evaluation | Interface | Rhythm or Tempo |
|---|---|---|---|---|
| Schwarz (2000) | Caterpillar | No | 2D explorer | No |
| Zils & Pachet (2001) | Musaicing | No | ? | Yes |
| Hazel(2001) | Soundmosaic | No | Command Line | No |
| Hoskinson & Pai (2001) | Soundscapes | No | Waveform/Menus | No |
| Xiang (2002) | Granuloop | No | 2D Controller | Yes |
| Kobayashi (2003) | Sound Clustering Synthesis | No | ? | No |
| Cardle et al. (2003) | Directed Soundtrack Synthesis | Videos of use cases | Waveform/Menus | No |
| Lazier & Cook (2003) | MoSievius | No | ? | Looping |
| Sturm (2004) | MATConcat | No | Waveform/Menus | No |
| Lindemann (2007) | Synful (Commercial) | Not available | Knobs/Sliders | No |
| Casey (2005) | Soundspotter | Retrieval Accuracy | Native Pure Data | No |
| Aucouturier & Pachet (2005) | Ringomatic | User Experiment | Lists/Menus | Drumming Tool |
| Simon et al. (2005) | Audio Analogies | Algorithm Performance | None | No |
| Jehan (2005) | Skeleton | Algorithmic evaluation | Waveform/Menus | |
| Schwarz et al. (2006) | CataRT | No | Timbre Space | Looping option |
| Weiss et al. (2009) | None | No | Waveform Segments | No |
| Frisson et al. (2010) | AudioGarden | No | Timbre Space / Waveform | No |
| Hackbarth (2011) | AudioGuide | No | 2D CataRT based | No |
| O'Connell (2011) | None | Yes | Native Pure Data | Looping demo |
| Bernardes et al. (2013) | EarGram | Author's impressions | Timbre Space and Matrices | Looping |

**Table 4.1:** Summary of Existing Concatenative Sound Synthesis Systems

# Chapter 5

# Specifying and Exploring Concatenative Synthesis

## 5.1 Introduction

As we discovered in Chapter 4 there exists a bewildering number of systems that propose or apply some level of concatenative synthesis technique. However widespread these methods are, some commonality arises across the majority of these when they are examined in closer inspection. Some of the important stages that we identify across this landscape include:

1. *Unit Decomposition* - Automatically segmenting larger sound files into constituent units using musically and perceptually relevant criteria.
2. *Feature Extraction* - Selecting and applying meaningful combinations of multi-level music descriptors than can be utilised to summarise and compare units of sound effectively.
3. *Unit Selection* - Combining units of sound according to some criteria, commonly using some notion of a *target* to compare constituent features against.

This chapter explores in depth these key facets. We describe the most common algorithms and techniques used in the literature and offer improvements where we see fit. In practical terms, all of the research carried out is implemented and encapsulated within a framework for Python we call *PyConcat*, described in more detail in Appendix D.

## 5.2 Unit Decomposition

Musical signals and sounds convey meaningful information at hierarchical levels of temporal scale. Choosing which level to work with naturally depends on the goals

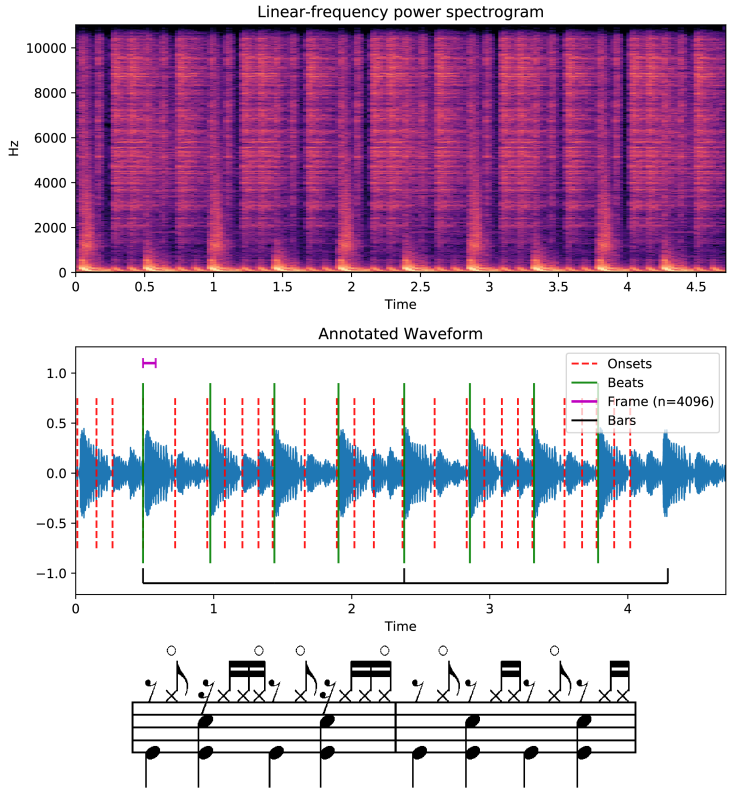



**Figure 5.1:** Annotated Waveform, Spectrogram and (approximate) Drum Transcription of "My Sound" by Joey Beltram (R&S Records)

of the composer in tandem with the musical context. Figure 5.1 visually depicts the different levels of scale that we can work with in concatenative synthesis tasks, beginning with the low-level framewise snippets in either the time or frequency domains. Framewise units are most typically used in granular synthesis but combined with feature analysis they are also found in many concatenative systems. Larger units can naturally be extracted uniformly (at regular time intervals), but the most common unit timescale used in concatenative system is as a result of performing some class of onset detection procedure. Onset detection algorithms are used to automatically detect events in audio signals. With an appropriately implemented algorithm and careful adjustment of its parameters based on the program material, the idealised output provides a list of time indices that correspond to the placement of a musical note. At the highest level of scale, we can use more involved and complex feature analysis to extrapolate units based on beat tracking and/or estimated tempo for example.

### 5.2.1 Framewise Segmentation

At the lowest level of unit decomposition is the framewise unit. In granular synthesis many combinations of microsound timescales of typically 20-200 ms are combined to create dense masses of sound "clouds". Granular synthesis systems typically control the recombination of these sounds in a parametric fashion; the user specifies or manipulates controls determining endmost pitch, length and envelope qualities. As Schwarz (2003) notes, granular synthesis is "rudimentarily data-driven" but can be significantly augmented with pre-feature analysis - what we seek to achieve by performing concatenative synthesis.

Granular synthesis is generally performed in the time domain, directly moulding and shaping the portions of the waveforms themselves using, for example, an amplitude envelope or windowing function based on Gaussian or linear patterns (Roads, 1996, 2004). Another option is to take frame sizes of orders of $2^n$ (512, 1024, 2048 etc). and transform them to the frequency domain using a Fast Fourier Transform (FFT). Resynthesised signals can be reconstructed later in the time domain using the Inverse Fast Fourier Transform (IFFT) with overlap and add. While not an overly common approach to concatenative synthesis, it has been alluded to by Kobayashi (2003), Puckette (2004), An et al. (2012), and some practical artistic applications have been summarised by Schwarz (2006). The former describes how IFFT synthesis was used in the theatre piece *La Légende des Siècles* (2002) to assemble FFT frames based on a target specification of pitch and energy. Frame-based segmentation and Fourier resynthesis are available as unit scale options in PyConcat, but have only been used and tested sporadically. The drawbacks of operating at this minuscule order of scale is the sheer level of data handling and computation it entails. Furthermore, if the composer is additionally working in the frequency domain they need to take extra care with continuity and phase, something a HMM unit selection scheme further on would help address.

### 5.2.2 Onset Detection for Unit Decomposition

#### 5.2.2.1 Onset Detection Function

At the heart of an onset detection algorithm lies the extrapolation of a detection function. The detection function, also referred to as the novelty, change or reduction function, is critical in reducing the complete audio signal from its higher sample rate to a lower frequency function of candidate peaks that reveal potential onsets. A detection function tries to detect moments of large change or flux, such that these drastic points of disparity in a signal quality like its energy signify the occurrence of a musical note or percussive transient. Time domain approaches work by taking a frame of audio samples of size $N$, summing the rectified absolute values (as in an envelope follower) or computing the Root Mean Square (RMS), then calculating the 1st-order differential of those values (Laroche, 2003; Duxbury et al., 2002).

More robust methods typically operate in the frequency domain. Critics of time domain methods point to its difficulty in recognising change of events in signals with high energy or noisy background (Grzywczak & Gwardys, 2014; Eyben et al., 2010). By operating in the frequency domain we can pick up on more information, in addition to energy/envelope changes, that might prove efficacious in the characterisation of a note event, such as the change of pitch revealed by its shifting spectrogram (Schloss, 1985; Lerch, 2012). Three of the most popular spectral methods include:

- Spectral Flux: Calculate the distance between two consecutive frames of the magnitude spectrum, using some distance measure such as Euclidean or absolute value.

$$F(n) = \frac{1}{N} \sum_{k=1}^{\frac{N}{2}-1} \sqrt{(X_k(n) - X_k(n-1))^2} \tag{5.1}$$

- Complex Flux: Bello et al. (2004) proposed extending the spectral flux method to incorporate phase information in addition to magnitude. By tracking the instantaneous phase of the complex spectrum, the real frequency can be calculated as per the phase vocoder (Roads, 1996). If the pitch of a note corresponds to a (mostly) steady state frequency then, in theory, deviations from the expected phase of a signal should reveal note events of signals where large energy changes are not such prominent indicators.

- High Frequency Content (HFC): Many percussive events are broadband in their frequency content, most notably those involving cymbals or snare sounds. To emphasise the contribution of higher frequencies to the novelty function, frequency bins of the spectrum can be weighted proportionally then summed to produce a framewise HFC descriptor more tailored for percussive sounds.

$$F(n) = \frac{1}{N} \sum_{k=1}^{\frac{N}{2}-1} W_k |X_k(n)|^2 \tag{5.2}$$

- Deep Learning Approaches: Reflecting recent trends in the arena of machine learning towards deep learning and application of neural networks, state of the art methods are also delivering some of the most promising results for creating usable novelty functions. Marolt et al. (2002) first proposed using a neural network, trained on a corpus of synthesised piano performances, to detect the presence of onsets in the amplitude envelopes of an IIR filterbank of 22 filtered signals. Lacoste & Eck (2007) proposed using supervised learning of a neural network trained to classify onset or non–onset in frequency representations of the signal, using an STFT and a Constant-Q Transform. Convolutional neural networks, such as those outlined by Schlüter & Böck (2013, 2014) borrow from object classification tasks in computer vision research to analyse streams of

spectrograms. Naturally, and in contrast to the more traditional signal processing methods, the success of supervised learning methods relies greatly on the data used for training.

#### 5.2.2.2 Thresholding and Peak Picking

The novelty functions returned by those methods described in the previous section returns candidate peaks, or, in the case of neural network oriented methods, probabilities of candidates that potentially correspond to onset times. To filter out spurious peaks and retain those local maxima, the novelty function needs to be processed through thresholding and application of a peak picking algorithm.

Fixed thresholding retains only those peaks from the novelty function that exceed a constant threshold value. This works well for program material occupying smaller dynamic ranges, as in highly compressed music or percussive recordings. For material that varies in dynamics over time it is rather crude and inflexible - for such scenarios an adaptive thresholding procedure is preferable. Adaptive thresholding essentially computes another function that gives an instantaneous threshold value for each sample of the novelty function based on some analysis of that very function. Standard methods proposed by Bello & Daudet (2005), Dixon (2006), Böck et al. (2012) and Böck & Widmer (2013), choose a window of time $w$ around the current novelty function sample and derive a rolling average or low pass filtered version of the function, and reject peaks below some multiple of that threshold function.

Peak picking, as summarised by Böck et al. (2012) can finally be applied by constraining the novelty function $F(n)$ to a set of criteria as in:

$$\begin{aligned} P(n) &= max(F(n-w:n+w)) \\ P(n) &\geq mean(F(n-w:n+w)) + \delta \\ n - n_{last} &> \beta \end{aligned} \tag{5.3}$$

where $n$ is the current frame number, $w$ is a window of frames around $n$, $\delta$ is a threshold multiplier to be applied to the rolling average and $\beta$ is used to reject potential onsets if they occur within a number of frames of a previously detected onset. Naturally, this set of criteria assumes that the source signal already exists in its entirety as in a pre-recorded sound file.

For real-time onset detection it is more challenging and less accurate as we do not have access to future frames of audio, unless some pre-delay is applied (as is done in look ahead limiters and processors that need some advance information in order to duck the signal). Even if no pre-delay is applied, novelty frames can only be compared to those captured in the past, which at least incurs a delay penalty of $w * N/SR$ milliseconds given frame length $N$ and sample rate $SR$ and frame window $w$.

#### 5.2.2.3 Evaluation of State of the Art Onset Detection Algorithms

Our intention in this thesis is not to reinvent the wheel in basic building blocks for musical signal processing, thus we propose no new methods for onset detection tasks. We do however, need to evaluate the state of the art methods currently available and compare their suitability for our needs in concatenative synthesis, which is documented in this section.

**Datasets** To evaluate the respective performance of each method, we gathered a number of datasets that are used in the literature and were made available, comprising:

- ENST-Drums: Provided by Télécom ParisTech (Gillet & Gaël, 2006), contains an audio-visual annotated dataset of performances by 3 professional drummers using multi-track studio recording in a variety of configurations (single hits, phrases and accompaniments) and a variety of styles including jazz and rock.
- Modal (Musical Onset Dataset And Library): Hand annotated, contains 501 onsets across 71 files, with a mix of mostly monophonic events (Glover et al., 2011)
- JKU: Contains 321 files with 27,774 total onsets. Compiled from a number of different sources by Sebastian Böck for evaluating his SuperFlux algorithm (Böck & Widmer, 2013). The algorithm was developed especially to handle vibrato, and accordingly the dataset contains a large number of samples that purposely address vibrato, such as samples from opera or Western classical string technique.

**Algorithms** Three different exemplar onset detection methods, available in the Essentia (Bogdanov et al., 2013) and the Madmom (Böck et al., 2016) libraries, were integrated into our own PyConcat framework and their performances were estimated on the previously described datasets. The specifics of the algorithms themselves are explained as followed:

- *Essentia-OnsetRate:* Essentia's "OnsetRate" algorithm combines the HFC and Complex domain novelty function detection methods. The two functions are multiplied by each other in order to emphasise those frames in agreement with each other.
- *Madmom-Superflux:* Madmom's implementation of the Superflux algorithm as proposed by Böck et al. (2013). It expands on the spectral flux approach to handle program material that contains soft attacks, and instruments with vibrato tendencies such as strings, winds and the voice, that are difficult to capture using energy differences alone.

- *Madmom-CNN:* Uses a convolutional neural network to detect likelihood of onsets within a frame by frame basis (Schlüter & Böck, 2013, 2014). This algorithm achieved the highest scoring F-Measure and Recall and the second highest scoring Precision of the Onset Detection task as part of MIREX 2016.

**Procedure** Onsets were extracted using a batch procedure on all of the audio files with each algorithm using our PyConcat framework. For each audio file a single text file was produced with estimated onset times. These were then compared with the ground-truth annotations using the mir_eval python package, a Python library for evaluating MIR related tasks according to MIREX guidelines (Raffel et al., 2014). An onset is defined as correctly identified if the time annotation is within +/- 50 ms of the ground truth annotation. Errors are categorised as either false positives or false negatives:

- *False Negative:* No onset estimation exists for a ground truth annotation
- *False Positive:* An onset estimation outside the tolerance window for any ground truth annotation

Scores are computed for each file according to the standard information retrieval fractional measures of precision and recall as well as the F-measure, a derived measure that attempts to consolidate both previous measures for easy comparison of overall algorithms' performance. Borrowing from the definitions provided in the sci-kit learn documentation[57] but adapted for the specific task of onset detection, we can define them as follows (where $T_p$ signifies true positives - correctly identified onset, and $F_p$ signifies incorrectly identified onsets).

- *Precision:* the fraction of total estimations that are correct:

$$P = \frac{T_p}{T_p + F_p} \tag{5.4}$$

- *Recall:* the fraction of the total annotations that are correctly retrieved:

$$R = \frac{T_p}{T_p + F_n} \tag{5.5}$$

- *F-Measure:* an aggregate score incorporating both precision and recall, defined as the harmonic mean of both:

$$F_1 = 2 * \frac{P * R}{P + R} \tag{5.6}$$

[57] http://scikit-learn.org/stable/auto_examples/model_selection/plot_precision_recall.html

| Algorithm | Precision | | Recall | | F-Measure | |
|---|---|---|---|---|---|---|
| | Mean | Std. Dev. | Mean | Std. Dev. | Mean | Std. Dev. |
| **CNN** | **0.9208** | **0.1452** | **0.9073** | **0.1603** | **0.9062** | **0.1568** |
| SuperFlux | 0.8719 | 0.1817 | 0.8702 | 0.1849 | 0.8550 | 0.1898 |
| OnsetRate | 0.8412 | 0.2166 | 0.7523 | 0.2287 | 0.7779 | 0.2128 |

**Table 5.1:** Onset Detection Algorithm Results

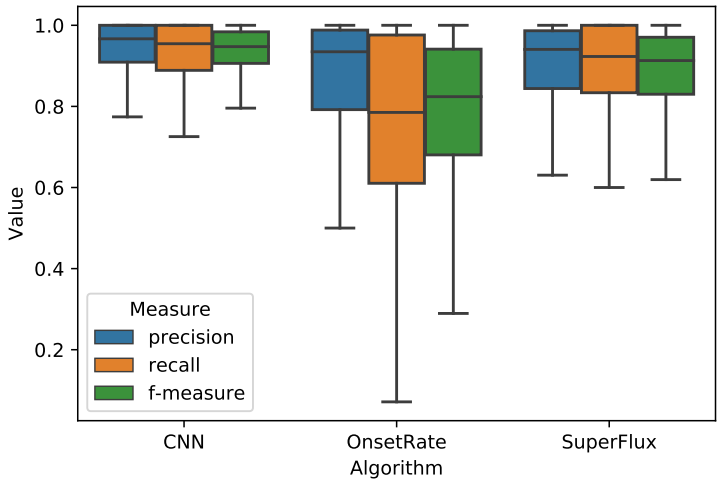



**Figure 5.2:** Boxplots of Results for each algorithm and measure.

#### 5.2.2.4 Results

Results for all the files from each dataset were collated and statistics computed using mean and standard deviations, as summarised in Table 5.1, while Figure 5.2 provides corresponding box plots visually depicting the distributions. As is evident, the onset detector based on the Convolutional Neural Network (CNN) is the highest scoring and most consistent of the onset detectors measured based on overall mean values and the small degree of spread in the box plots. Superflux also delivers excellent results, albeit with the introduction of more variance across samples. The combined methods present in Essentia's OnsetRate algorithm performed the most poorly, with the highest degree of spread across all measures.

Table 5.2 shows in more detail the statistics for each algorithm separated by the particular datasets evaluated and with the highest precision, recall and F-measure highlighted for each. It is interesting to note that all the algorithms deliver good results on the ENST dataset, which is entirely percussive. Superflux and CNN perform much

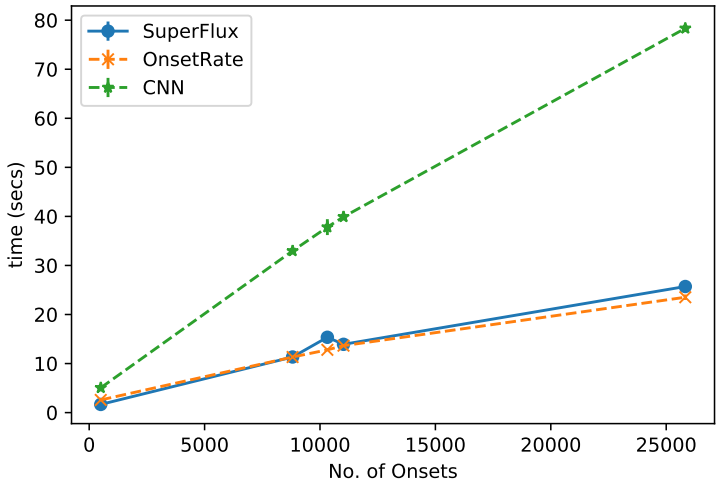



**Figure 5.3:** Mean and error of the running times of each algorithm over n=10 runs

better on the JKU dataset on which it is been designed. These algorithms demonstrate their robustness on program material of all types, including sustained vibrato sounds as well as clear transient heavy percussive material.

The smaller Modal dataset appears to cause the most difficulty for all algorithms, but relaxing the tolerance time-window constraint from its default of 25ms to 50ms (actually the tolerance time used for MIREX tasks) improves the accuracy for all algorithms by a significant factor, suggesting it could be an issue of ground truth annotation style compared with the other datasets.

In terms of running time and overall algorithm performance, Figure 5.3 compares the averaged time and standard error (not visible due to low variance of < 1.0 s between consecutive runs). Recent trends towards deep learning solutions, while displaying promising returns in terms of accuracy over other supervised learning methods, frequently pale in comparison when considering efficiency and running times of the solution. Even with Madmom's built-in facility for distributing computational loads over multiple processors, the CNN method performs much slower than the two flux-based approaches. Quite possibly the scale of the datasets is much larger than typical applications for concatenative synthesis, but the comparative performance of onset detection methods is an important point to consider when we look at real-time or at least quasi real time applications in due course.

#### 5.2.2.5 Beat Tracking and Tempo Extraction

It seems, from our impression of the literature, that in concatenative synthesis systems the typical unit length is of the onset (or to give it its perceptual equivalent, the note) (Schwarz et al., 2006; Frisson et al., 2010; Bernardes et al., 2013) - as extrapolated

| Algorithm | Dataset | Onsets | TP | FP | FN | Precision | Recall | F-measure | mean | std |
|---|---|---|---|---|---|---|---|---|---|---|
| CNN | ENST-1 | 8813 | 8057 | 697 | 756 | **0.921** | **0.937** | **0.921** | 2.5 | 5 |
| | ENST-2 | 10319 | 9738 | 484 | 581 | **0.945** | 0.956 | **0.948** | 1.6 | 5.4 |
| | ENST-3 | 11019 | 10154 | 464 | 865 | 0.932 | 0.926 | 0.928 | 2.5 | 5.4 |
| | JKU | 25827 | 23293 | 978 | 2534 | **0.941** | **0.9** | **0.916** | -3.6 | 5.2 |
| | Modal | 501 | 418 | 122 | 83 | **0.774** | 0.798 | **0.744** | -3.9 | 3.2 |
| Superflux | ENST-1 | 8813 | 7867 | 756 | 946 | 0.911 | 0.931 | 0.915 | -1.4 | 4.9 |
| | ENST-2 | 10319 | 9553 | 549 | 766 | 0.94 | **0.957** | 0.946 | -2.4 | 5.4 |
| | ENST-3 | 11019 | 10018 | 481 | 1001 | **0.943** | **0.931** | **0.935** | -1 | 5.3 |
| | JKU | 25827 | 20382 | 2820 | 5445 | 0.847 | 0.809 | 0.811 | -5.9 | 5.9 |
| | Modal | 501 | 392 | 247 | 109 | 0.714 | **0.837** | 0.706 | -4.2 | 3.5 |
| OnsetRate | ENST-1 | 8813 | 6875 | 560 | 1938 | 0.901 | 0.875 | 0.871 | -5.7 | 7.1 |
| | ENST-2 | 10319 | 8116 | 447 | 2203 | 0.941 | 0.872 | 0.895 | -6.3 | 7.3 |
| | ENST-3 | 11019 | 8456 | 572 | 2563 | 0.91 | 0.868 | 0.875 | -5.4 | 7.4 |
| | JKU | 25827 | 15736 | 3335 | 10091 | 0.793 | 0.627 | 0.686 | -7.1 | 8.8 |
| | Modal | 501 | 333 | 213 | 168 | 0.719 | 0.784 | 0.732 | -4.6 | 4.6 |

**Table 5.2:** Extended statistics for each algorithm by dataset

from a suitable onset detection process in the manner previously described. They are reasonably straightforward to compute and represent musically relevant building blocks for producing larger sequences of sounds. But larger musically relevant unit sizes are also a possibility. Building on top of state of the art techniques in onset detection, MIR researchers are actively trying to coalesce this research with knowledge of tempo and rhythm to pursue algorithms that can perform automatic beat detection within signals.

Considering that most of the time we will be working at the onset scale of segmentation, analysis and selection, an extensive evaluation of state of the art methods in beat detection such as we performed for onset detection is outside of the scope of this thesis, but we do summarise some of the key works in this area including their respective MIREX results and expose them as possible unit decomposition options within our own framework.

Figure 5.4 consolidates various rhythmic concepts from the onset to the bar level and how they relate to the waveform representation and timeline of a typical musical recording. Beats are defined as the perceptual tactus or dominant metrical pulse level that corresponds to where humans are most likely to tap their foot (Jehan, 2005; Ellis, 2007; Stark et al., 2009). A stream of beats implies a certain tempo or Beats Per Minute measure of frequency that allow us to understand and compare the speed of musical works, and beat tracking and tempo detection are considered related tasks (McKinney et al., 2007). As Korzeniowski et al. (2014) note, there are many factors in interesting musical recordings that hinder computational facility to determine beats with ease, such as syncopation, hemiola and any other rudimental technique that seeks to thwart regular pulsation. By using tempo estimation and probabilistic prediction of where beats *should* be when they aren't, beat tracking algorithms try to account for these factors.

McKinney et al. (2007), provide an excellent summary of the existing state of the art methods at their time of publication, organising the methods by differing implementations of a two-stage process. The process consists initially of the *driving* stage, which produces features directly from the signal. This is followed by *periodicity* stage, which outputs estimates of tempo and/or beats based on the analysis of those features (McKinney et al., 2007).

As was emphasised previously, beat tracking studies typically build on onset detection research, as beats essentially represent a perceptual and synchronous subset of onset locations in time. Thus most features HMM derived for the driving stage consist of applying an onset detection method to produce candidate onset times (Brossier, 2006; Dixon, 2006; Ellis, 2007; Degara et al., 2012; Zapata et al., 2014) but not always. Antonopoulos et al. (2007) for example, use self-similarity of the MFCC frames as its input. Degara et al. (2012)[58] build upon their existing offering by integrating multiple input features that include energy flux, spectral flux, difference of Mel bands

[58] Also one of the beat trackers available in Essentia

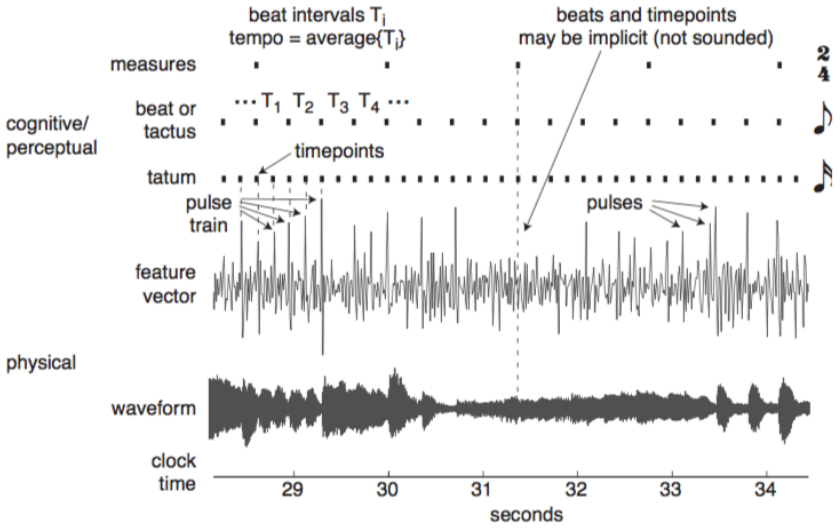



**Figure 5.4:** Perceptual and Physical Concepts of Rhythm in a Scott Joplin Ragtime. Image from Sethares (2007)

etc. Multiple candidates arising from the various feature inputs are resolved using a method of query by committee and maximum mutual agreement (MaxMA) proposed by the author previously in other work (Zapata et al., 2012, 2014).

The periodicity stage, or where the features are filtered to produce beat locations is where the algorithms deviate the most from each other in terms of approach but many methods in the literature at least start from a probabilistic standpoint. Ellis (2006; 2007) uses onset detection followed by autocorrelation to get an initial estimate of the overall tempo of the track. This information is used by the beat-tracking module to compare the idealised beat times against computing the globally optimised set of beat times using IOIs and dynamic programming in a manner not so far removed from that proposed by Alonso et al. (2007). Degara also utilises a HMM-based algorithm, augmented with an intrinsic framework that models sequences of non-beat states as well as beat states.

A multi-agent system for detecting periodicity has been employed by a number of investigators (Goto, 2001; Dixon, 2007; Oliveira et al., 2012). In the case of Dixon, for instance, an initial tempo hypothesis and potential first beat from an initial set of onsets are assigned to several agents. Predicted beat times are then generated from the initial onset and hypothetic tempo, and onsets falling within a tolerated window of those predictive times are labelled as actual beats. A complex resolution system examines the degree of agreement between agents to decide on the final beat locations.

| Key | Name | Authors | F-Measure | | | |
|---|---|---|---|---|---|---|
| | | | SMC | MAZ | MCK | Mean |
| BK1 | DBNBeatTracker.2016 | S. Böck, F. Krebs | 56.8313 | 57.5286 | 63.6093 | 59.3231 |
| BK3 | DBNDownBeatTracker | S. Böck, F Korzeniowski | 52.8343 | 73.8911 | 62.5299 | 63.0851 |
| BK2 | CRFBeatDetector.2016 | S Böck, F Krebs | 52.3142 | 55.1651 | 62.7315 | 56.7369 |
| SB9 | BeatRoot Vamp Plugin | S. Böck | 52.097 | 58.9357 | 63.8961 | 58.3096 |
| SB8 | QM Tempo Tracker | S. Böck | 49.8366 | 52.2792 | 63.8436 | 55.3198 |
| JZ1 | MultifeatureBT-Inf-7odf-Repet | J. R. Zapata | 36.8192 | 50.63 | 52.6104 | 46.6865 |
| JZ2 | MultifeatureBT-Reg-7odf-Repet | J. R. Zapata | 36.4073 | 47.874 | 53.2314 | 45.8376 |
| CD3 | QM Tempo Tracker | C. Cannam, M. Davies | 33.663 | 49.6229 | 52.8758 | 45.3872 |
| CD2 | BeatRoot Vamp Plugin | C.Cannam S.Dixon | 30.34 | 41.5218 | 52.6579 | 41.5066 |

**Table 5.3:** Mirex 2016 Beat Tracking Results

**Current State of the Art and Deep Learning Approaches** Table 5.3 shows the F-Measure scores for every algorithm submitted to the 2016 MIREX competition. The datasets under scrutiny include:

- *SMC* - 160 30-second excerpts of music in a variety of instrumentation and styles but with a stable tempo. 20% of the examples contain non-binary meter.
- *MAZ* - 367 Chopin Mazurkas with shifting tempos.
- *MCK* - 217 tracks of roughly 40s long, with 19 “easy” and 198 “hard” excerpts from Romantic music, film soundtracks, blues, chanson and solo guitar.

Beat tracking is a significantly more complex task compared to onset detection, and evidently the results reflect this. Ranges of accuracy vary greatly across the different datasets and, while no single algorithm achieves the highest F-Measure, at least Böck and Korzeniowski’s DBNDownBeatTracker manages to attain the highest mean performance. Once again we see the headway deep learning based methods are making in all manner of pattern recognition and machine learning tasks, as this method relies on a recurrent neural network armed with a bank of resonant comb filters.

### 5.2.3 Future Directions - Mixing and Source Separation

Most concatenative synthesis systems we have studied (including those techniques we propose) tend to consider the concatenation process in purely horizontal and temporally inclined terms. Naturally, it is pertinent to question whether the same process can be applied systematically in the vertical dimension characterised by the frequency spectrum. Or, in more simpler terms - can we stack units on top of each other systematically in addition to chaining them in sequence? While not the focus of this thesis

(but certainly a promising avenue for further study), we briefly summarise relevant topics of interest and techniques that are or can be used.

#### 5.2.3.1 Vertical Selection with Mixture Models

Hoffman et al. (2009) have outlined a system that seeks to recreate the target *spectrum* by, as Coleman describes it (2010), *superimposing* (using convolution) a series of spectra from a corpus. To reduce the complexity associated with high dimensionality, a search space reduction with a Markov Chain Monte Carlo (MCM) is applied.

In a version of AudioGuide by Hackbarth (Hackbarth et al., 2013), they also outline the system's facility for enabling "layered sounds" and "vertically stratified complexes" based on formulas (Tardieu, 2008) developed for a prior system in the OpenMusic environment called Orchidée or Automatic Orchestration Tool (Carpentier & Bresson, 2010). The goal of their work is to transfer some of the principles of traditional orchestration (combining and arranging for instruments to create desired blends of timbre) to the realm of computer music by allowing prediction of mixtures as well as concatenations to match a target specification of a sound.

Hackbarth (Hackbarth, 2011) describes the derived "Subtractive Spectral Algorithm" for superimposition. When an appropriate unit is selected from the corpus as a candidate match for the target, its time-varying Mel-amplitudes are subtracted from the target sound and a 'residual' target spectrum remains. The unit selection procedure is performed repeatedly until it halts when the residual target spectrum energy falls below a certain threshold. The resulting units are then mixed together additively. A visual depiction of this over several mixtures can be seen in Figure 5.5. Clearly the algorithm tends towards selection of louder units initially, with gradually softer ones emerging as the subtracted target energy diminishes.

Coleman (2015) invokes the term "mixtures" to describe concatenative synthesis problems where the corpus might not have all the features to effectively simulate a target, but a carefully selected combination of them might. He uses the analogy of several single notes combining to form a chord, except the synthesis engine would use a weighted combination of superimposed frames mixed together.

#### 5.2.3.2 Vertical Segmentation with Source Separation

As we highlighted previously in the case of the missing fundamental frequency, our auditory system is remarkable in its processing ability and inference engine. More remarkably, perhaps, is its innate capacity for on the fly separation of real-time audio signals at will, or to give it its technical term, blind source separation. In pyschoacoustic auditory scene analysis research (Bregman, 1994), source separation is illustrated usually in terms of the "Cocktail Party Effect": how is a human, at a social event, able to separate voices from background noise and home in on particular conversations? In musical source separation this is even more elusive - consider picking out

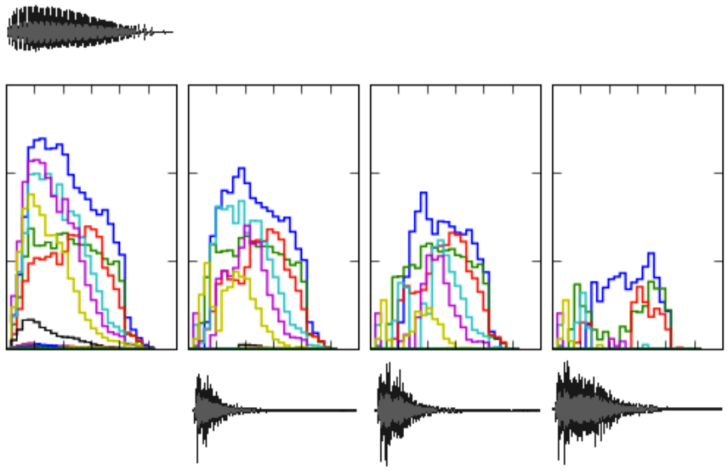


**Figure 5.5:** Visual depiction of spectral subtraction algorithm in AudioGuide. Image from Hackbarth (2011).

individual instruments in an orchestral passage, especially when, as Miron (2017) stresses, sources are likely to be correlated in time and frequency. Computational auditory scene analysis seeks to replicate this extraordinary human processing ability through digital signal processing means (Wang et al., 2006).

Initial efforts in computational source separation often involved matrix decomposition methods such as non-negative matrix factorisation, as demonstrated in the work of Fitzgerald (2004). Fitzgerald's initial focus was on automatic drum transcription via source separation but later gained wider recognition with some some successful demixing and upmixing from mono to stereo of recovered archive Beach Boys recordings (Fitzgerald, 2004). More recent state of the art approaches are once again successfully exploiting deep learning methods trained on large sets of musical data in genres like Western classical (Miron, 2017; Miron et al., 2017).

Just as a concatenative synthesis system will use onset detection to split larger sound files into smaller units along the time domain, source separation opens up the possibility of theoretically segmenting complex sounds in the frequency domain according to instrument, or more accurately, *streams* that hopefully contain related instruments (all depending of course on the material to be separated and the models upon which the system has been trained).

We envisage this as particularly useful in concatenative synthesis of drum rhythms. Perhaps a producer might want to decompose a number of kits into their constituent parts and mix and match with sounds from other sources. Mixing electronic drum sounds generated by a drum machine with naturally recorded acoustic drum sounds is a common strategy in dance music and source separation could prove invaluable here.

## 5.3 Describing Sounds with Features

### 5.3.1 Temporal Descriptors

#### 5.3.1.1 Intensity and Loudness

Physically speaking, sound is a medium transporting mechanical energy via waves. In recorded media, analog or digital signals are converted to electrical and magnetic energy that push speakers backwards and forwards in a manner analogous to the original waveform captured.

Digital signals are essentially a series of amplitude values sampled at a particular sampling frequency. Taking the absolute value at an instantaneous sample point or the maximum absolute value over a window of samples gives us the peak amplitude. Unless the peak amplitude occurs frequently, in itself it is not a very useful indicator of the overall amplitude of the signal, thus the Root Mean Square (RMS) is computed on the window of samples to get a better statistic of the general power of the signal (Puckette, 2006):

$$A_{RMS}\{x[n]\} = \sqrt{\frac{1}{N}(x[1]^2, x[2]^2, ..., x[N]^2} \tag{5.7}$$

where $N$ is the length of the discrete time signal $x$ and $x[n]$ are the individual samples. Most other works we have encountered related to concatenative synthesis use RMS as their principal loudness descriptor, particularly Sturm (2004) and Bernardes et al. (2013), but Jehan (2005) computes his loudness descriptor from energy in the frequency domain.

Human perceptual response to increments in amplitude is intrinsically logarithmic, thus to aid relative comparisons and metering purposes, amplitude values are often computed as a ratio of some reference value and reported using the deciBel (dB) scale, as Puckette (2006) indicates.

$$d = 20\log_{10}\frac{a}{a_0} \tag{5.8}$$

Intensity, power, energy etc. are all *objective* physical concepts. We use the term *loudness* to describe the subjective human perceptive experience that we feel when exposed to intensity of sound. As Lerch (2012) points out, the decibel is neither a true indicator of loudness because "equal-sized steps on the decibel scale are not perceived as equal-sized loudness steps by human listeners", or as Stevens (1955) puts it more succinctly: "equal steps do not sound like equal steps".

Perceived loudness depends on many factors, and, as the long established Fletcher-Munson equal-loudness contours reveal, it is heavily skewed by frequency. Introduced in the 1930s (Fletcher & Munson, 1933), subsequently revised as an ISO standard in ISO 226:2003 (2003), this set of curves (Figure 5.6), produced using listening

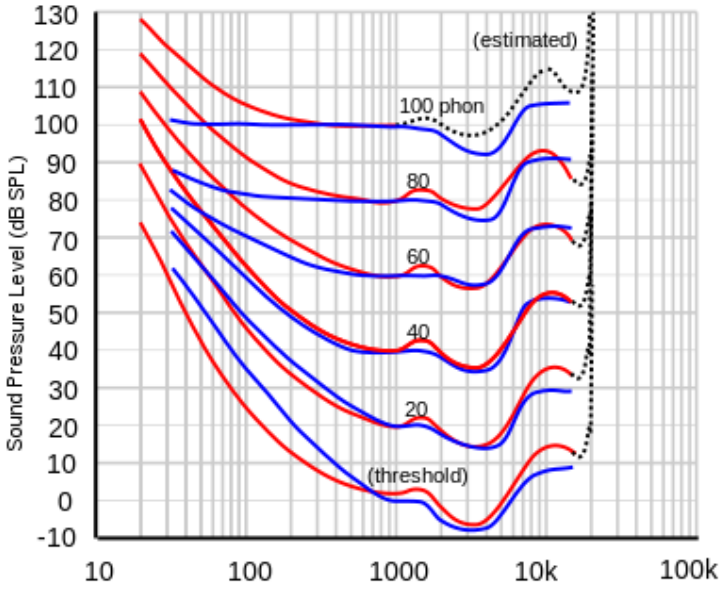



**Figure 5.6:** Equal-loundness curves in red with ISO revisions in blue. Image from Wikipedia.

tests, estimates frequency versus sound intensity pressure level where listeners perceive loudness to be roughly equal. To make matters worse, these curves change according to overall intensity, such that at the difference is more pronounced between frequency ranges at low intensities, while gradually flattening out as intensity increases.

Luckily, more pyscho-acoustically aware measures that take into account such subtleties and phenomena are implemented and available as content analysis features. The European Broadcasting Union (EBU), for example, proposes the EBU R 128 metering standard in an effort to enforce normalisation consistency across differing programme material in broadcasts[59] (ITU-R BS.1770-3, 2013; Lerch, 2012). It consists of a perceptually tuned K-weighting filter followed by mean square power estimated at momentary, short-term and longer-term intervals.

We are using content analysis mainly for our unit selection algorithm to compute similarity between feature vectors so advanced metering standards like the EBU R 128 are a bit complex for our purposes, and we haven't seen any instances of its usage in synthesis systems. Another, more basic loudness descriptor available in Essentia's library implements Stevens' Power Law. Stevens' Power Law arose out of efforts to improve on the Weber-Fechner pyschophysics laws that relate changes in physical stimuli with perceptual response to those changes (Stevens, 1975; Skovenborg & Nielsen, 2004). It is elegant:

[59] And combat au-fait advertising and music producers exploiting dynamic range compression to make their products stand out from others.

$$S(i) = k.i^n \tag{5.9}$$

where $S$ is the sensation (perceived magnitude), $i$ is the intensity, $k$ is a unit dependent constant and $n$ is an exponent that depends on the stimulus. Through many perceptual experiments, Stevens produced exponents for a wide variety of physical stimuli, ranging from brightness of light to taste when exposed to sucrose or salt, for example. In the case of loudness, that exponent is estimated at 0.67 (Stevens, 1975), and in Essentia it is implemented as the signal energy (5.10) raised to that exponent (5.11).

$$E\{x[n]\} = \sum_{n=-\infty}^{n=\infty} |x[n]|^2 \tag{5.10}$$

$$L\{x[n]\} = E\{x[n]\}^{0.67} \tag{5.11}$$

We use the Stevens' Law based descriptor in our systems as it provides (in our view) an acceptable trade off between ease of integration and grounding in some awareness of perceptual theory. Loudness is a scalar feature, and one of two features we compute solely in the time domain. It is an extremely important quantity that is essential in ensuring that units selected match not only the overall naturalness and dynamics of the target sound, but also remain consistent over the continuous trajectory of the sequence subsequent to concatenation.

#### 5.3.1.2 Log-Attack Time

The evolution of the energy in a sound is frequently described in terms of its attack, decay, sustain and release envelope (Peeters, 2004; Kim et al., 2006a; Brossier et al., 2004) and is a central facet of modular and digital sound synthesis (Russ, 2004).

Technically it is resolved by taking the logarithm of the time taken from $t0$ to $t1$, where $t0$ is some initial energy percentage of the envelope, while $t1$ is some maximal percentage of the envelope, indicating the end of the attack and the start of the decay or sustain portion (Eq. 5.12).

$$lat = log(t1 - t0) \tag{5.12}$$

Why do we take the logarithm? Well, just like other perceptual responses to physical stimuli studies have found higher degrees of correlation in studies examining timbre discrimination when the logarithm is applied.(McAdams et al., 1995, 1999; Agres et al., 2016). The log attack time is obviously very useful for discriminating sounds that have distinctive attack profiles (Herrera-Boyer et al., 2003), such as the flute versus a snare drum for example, so naturally this feature makes more sense for "one-shot" instrumental classification versus excerpts of recordings for the purposes of genre detection.

### 5.3.2 Spectral and Timbral Descriptors

Timbre is an extremely complex and elusive aspect of sound for a very simple reason: it is a quality that is likely an emergent property, with indeterminate dimensionality (Elliott et al., 2013; Siedenburg & McAdams, 2017) or even dimensionality exclusive to a particular timbre (Krumhansl, 1989). A typical definition usually follows along the lines of that issued by the Acoustical Society of America:

> "*That attribute of auditory sensation which enables a listener to judge that two nonidentical sounds, similarly presented and having the same loudness and pitch, are dissimilar*" (ANSI, 1994)

An elegant distillation undoubtably, but other researchers point out (Siedenburg & McAdams, 2017) there is difficulty in these so-called "wastebasket" reductionist attempts that leave us none the wiser:

> "*timbre tends to be the the psychoacoustician's multidimensional wastebasket category for everything that cannot be labeled pitch or loudness.*" (McAdams & Bregman, 1979)

> "*This is, of course, no definition at all. [...] The problem with timbre is that it is the name for an ill-defined wastebasket category. [...] I think the definition [...] should be this: 'We do not know how to define timbre, but it is not loudness and it is not pitch.' [...] What we need is a better vocabulary concerning timbre.*" (Bregman, 1994)

#### 5.3.2.1 Fourier Analysis (and Synthesis)

A working knowledge of Fourier analysis and awareness of the first point (ANSI, 1994) does give a starting point for many systems that deal with timbre. Fourier's theorem says that any sound in the time domain can be transformed to a sum of sinusoid functions and associated amplitudes in the frequency domain or spectrum (Roads, 1996). Overlapping frames of samples taken from a longer signal can be multiplied by a suitable window function, giving a better indication of the evolution of the sound over time, in what is known as a the STFT or spectrogram (if observing just the magnitude) (Collins, 2010)(Eq. 5.13).

$$X[n,k] = \sum_{n=0}^{N-1} \{w[n]x[mH+n]e^{-j2\pi kn/N}\} \tag{5.13}$$

where $w[n]$ is a window function, $H$ is a hopsize, $m$ is the hop number and $k$ is the frequency bin number. If two sounds have the same pitch and loudness - or objectively speaking, the same *perceived* fundamental frequency and amplitude - then perhaps

examining the spectrogram should give us a better indication as to what discriminates the two. Examining different sounds using Fourier analysis shows us how different sounds are made up of spectrograms that not only have different magnitudes at different peaks, but also these magnitudes and peaks can evolve over time according to unique envelopes. These peaks in the spectrum are known as partials. When examining musical signals in a spectrum, we call the first partial that is at the lowest frequency in the signal as the fundamental frequency or first harmonic[60]. Any partial above this fundamental frequency that is related to the fundamental by an integer ratio is furthermore labelled as the second, third, fourth harmonic and so on, forming the harmonic series (Puckette, 2006).

Fourier theory also gives us a constructionist framework for additive synthesis of sounds - by carefully mixing parameters of a bank of sinusoids we can arrive at more complex sounds and even try to mimic some of those found via analysis. But additive synthesis gets very complicated very quickly. Besides having to keep track of large sets of parameters, computing huge banks of sinusoid functions concurrently is computationally expensive. Typical analog and digital synthesisers in the music producer's arsenal approach the problem from the other direction. Complex signals such as squares, sawtooths, or certain shapes drawn in a wavetable are filtered and sculpted to arrive at the desired sound in a process known as *subtractive* synthesis. Frequency Modulation (FM) synthesis also famously uses parametric control of a carrier signal's frequency modulated by a modulation frequency to produce a wide palette of sounds, but its minimal set of parameters can prove notoriously complex and difficult in trying to fathom a preconceived sound.

Getting back to spectral and timbral analysis[61], the Fourier transform and its spectrogram is a central concept in musical signal processing, and while usually not employed directly as a raw feature in itself, it forms the basis for numerous other algorithms (recall its role in flux-based onset detection) and descriptors in MIR literature and systematic practice which we will now turn to.

#### 5.3.2.2 Spectral Centroid

The spectral centroid gives a scalar value indication of the "centre of gravity" in the spectrum of the signal, and as such is computed as the weighted mean of the spectral bin frequencies, with the magnitudes acting as weights (Eq. 5.14).

$$c = \frac{\sum_{i=1}^{N} f(i)m(i)}{\sum_{i=1}^{N} m(i)} \tag{5.14}$$

[60] Though the human cortex has the extraordinary ability to "fill in" a missing fundamental (or *pitch of residue* (Weihs & Jannach, 2009)) for a complex harmonic tone even when absent.

[61] It's also worth mentioning spectralism here, a post-Serialist movement in European composition that ran tandem to the burgeoning minimalism across the Atlantic. At IRCAM, Gérard Grisey and Tristan Murail harnessed computational analysis of spectra to inspire dense scores on chamber works such as *Les espaces acoustiques* and *Gondwanda* (Ross, 2007; Harvey, 2000)

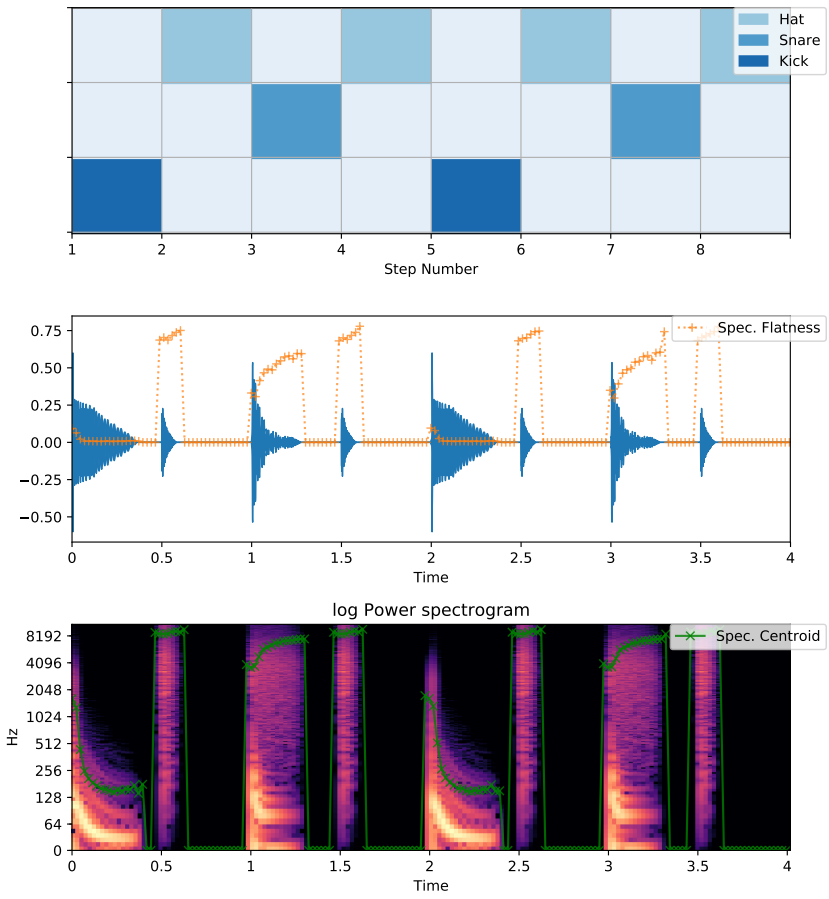



**Figure 5.7:** Symbolic grid representation of drum pattern (top). Waveform with spectral flatness (middle). Audio spectrogram with spectral centroid (bottom).

where $f(i)$ is the centre frequency of bin $i$, and $m(i)$ is its associated magnitude. It is considered a robust and reliable indicator of the perceptual "brightness" of a sound (Schubert et al., 2004), and is utilised in a variety of systems that classify sounds. In Figure 5.7 we can see three different impressions of a drum pattern (programmed in Ableton with a TR-909 preset). The top image shows the "symbolic" step pattern notation as would be used in a MIDI-based production environment such as Ableton. The middle image the shows the rendered waveform (or time-domain representation) of the drum pattern. Finally in the bottom image the log frequency/ power spectrum shows the frequency domain impression of the signal. Overlaid on the spectrogram, in green, is the spectral centroid computed for every frame of the spectrogram. As we can see the centroid follows the envelope shape of the centre of energy for the spectral evolution of the signal.

#### 5.3.2.3 Spectral Flatness

Another feature computed on the spectral profile at each frame is the spectral flatness. The spectral flatness or "Wiener entropy" gives an indication of how peaky or flat the spectrum is. Peaky spectra tend towards 0.0 and are indication of more harmonic or tonal sounds while a perfectly flat spectrum of 1.0 indicates stochastic noise. Mathematically it is defined as the geometric mean of the bin amplitudes of the spectrum divided by the arithmetic mean of the same bin amplitudes (Eq. 5.15).

$$f = \frac{\sqrt[N]{\prod_{n=0}^{N-1} X(n)}}{\frac{\sum_{n=0}^{N-1} X(n))}{N}} \tag{5.15}$$

In Figure 5.7 in the middle image we observe the spectral flatness has been plotted against the waveform view. Since the range bounds of spectral flatness fall between 0.0 and 1.0 this makes more sense than trying to normalise it into a range that can be plotted easily against the spectrogram. In any case, it can be clearly seen how the flatness function, while reasonably correlated with spectral centroid overall, deviates and approaches the origin for the kick timbre. While percussive sounds are generally considered to be less "tonal" overall compared to their non-percussive counterpoints, low frequency membranes such as kick drums and tom-toms do exhibit more discernible pitch compared to the clearly more broadband profiles in the snare and hi-hat, and this is evident simply by looking directly at the relevant time points in the spectrogram. Indeed, typical recipes for synthesising electronic kick drums (such as those found in Roland gear) often suggest combining a sine wave with an envelope controlling a sharp decay in frequency with some short impulse of filtered noise (to create a percussive "click") (Risset & Wessel, 1999; Reid, 2002).

#### 5.3.2.4 Cepstrum Analysis

Effective analysis of complex timbres in musical and natural signals typical employ derived features that exploit some understanding of human perception. The human ear is typically believed to respond to the dynamic limits of 0-120dB and frequency ranges of roughly 20Hz - 20kHz[62][63]. But as we know, human sensation to stimuli is not linear - we don't perceive phenomena in the same way physical measurements are made. The upshot of this is that important information "spreads out" at higher frequencies compared to the more compact resolution of the lower bands.

Pitch is the perceptual description we give to the way our brains organise frequency for the purposes of music. If a tone is perceived to have a pitch at a particular fre-

[62] Thanks to many years of ignoring advice on earplugs, I strain to hear anything above 14kHz!

[63] A famous anecdote recounted by revered console designer Rupert Neve claims that Beatles engineer Geoff Emerick discovered an anomaly in a desk that was attributed to an oscillation at 54kHz (Winer, 2012), but later they concede that this was probably due to suboscillatory spillage into the audible range.

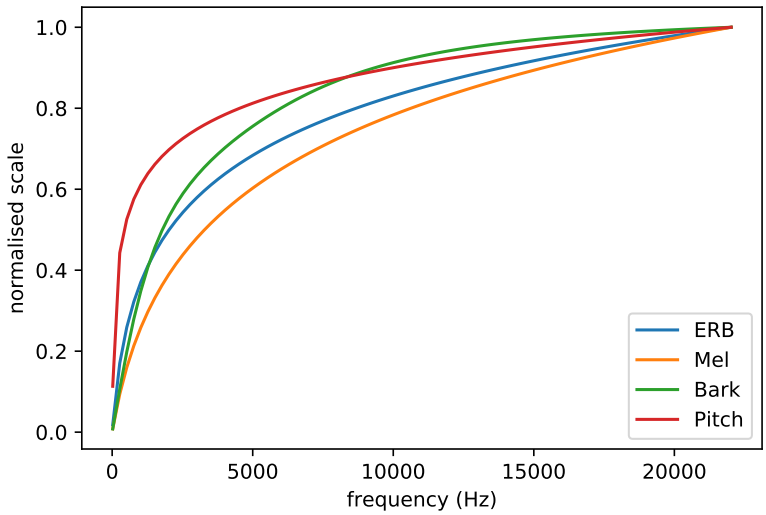



**Figure 5.8:** Different perceptual scales (normalised) for the audible frequency range

quency, doubling the frequency of that tone is perceived as the same pitch except one *octave* unit higher.

Depending on when (what era) and where (what culture) you were born in the world, how that octave is divided up can be very different (take for example, just intonation or Indian art music). Assuming the chromatic scale based on equal temperament tuning, its octave is divided into 12 semitones using the following formula:

$$p = 69 + 12 * \log_2(\frac{f}{440}) \tag{5.16}$$

What is essentially happening here is a type of frequency warping - compressing sound frequency onto a smaller, perceptual scale that is more tuned for human perception and cognition. In fact, there are other frequency warping formula in the literature that serve to warp frequency according to some conceptually pyschoacoustical or perceptual scale. Figure 5.8 shows a number of perceptual scales and their normalised output with frequency in Hz as input, notice how they all exhibit a sharp rise that flattens out as it approaches the upper bounds of human response to frequency.

Cepstrum analysis has been one of the most dominating analysis techniques in speech processing for a long time. The word cepstrum itself rearranges the first four letters of the word spectrum, and has been described variously as a type of "spectrum of a spectrum". Its usage within speech analysis stems from the fact that it allows to separate the two dominant spectra of a vocal sound - the initial glottal impulse excitation and the slower resonance of the vocal tract (Roads, 1996; Kim et al., 2006a).

To compute cepstrum is comparatively straightforward. The FFT of the signal is taken

as normal. The magnitude is retained and the log magnitude spectrum is computed. Finally, the inverse FFT (IFFT) is applied to the log magnitude spectrum. However, normally some frequency warping formula is applied to compress the spectrum in some pyschoacoustic and perceptually motivated manner, and it is here we derive the family of features that include Mel-Frequency Cepstral Coefficient (MFCC), Bark-Frequency Cepstral Coefficient (BFCC) and GFCC (Gammatone Frequency Cepstrum Coefficients warped according to Equivalent Rectangular Bands (Shao et al., 2009)). Figure 5.10 shows the computed coefficients and the ranges of these features side by side for visual comparison.

**MFCCs** MFCCs are the most common flavour of cepstrum analysis (Kim et al., 2006a). MFCCs use a Mel-spaced filterbank to convert the spectrum to energy in Mels (Figure 5.8). The Mel scale, based on perceptual experiments performed by Stevens et al. (1937), provides a mapping from frequency to Mel pitches that listeners have deemed equal to each other in distance, in a manner similar to the equal-loudness contours previously. While not visible in the normalised representation in Figure 5.8, as Kim et al. (2006a) notes, the Mel scale and frequency are actually linearly correspondent up until around $500Hz$, at which point it begins to space out across the upper range.

The formula for converting frequency in Hz to Mel pitches is given by:

$$m = 2595\log_{10}(1+\frac{f}{700}) \tag{5.17}$$

To compute the MFCCs the following steps are taken (Logan, 2000; Lyons, 2015):

1. Perform the STFT as given by Eq. 5.13.
2. Get the magnitude given by $M(n) = |X(n)|$.
3. Get the log magnitude given by $L(n) = log(M(n))$.
4. Warp the log-magnitude spectrum to the Mel scale using 40 triangular filters according to (5.17).
5. Finally take the Discrete Cosine Transform (5.18) of the Mel-Frequency log spectrum to obtain the 40 MFC coefficients.

$$X_k = \sum_{n=0}^{N-1} x_n cos[\frac{\pi}{N}(n+\frac{1}{2})k] \tag{5.18}$$

Figure 5.9 shows how MFCC extraction is performed in the Essentia environment, showing the signal flow through the various stages, conceptually encapsulated as algorithms. Essentia processes data in a graph-based network, not unlike signal flow

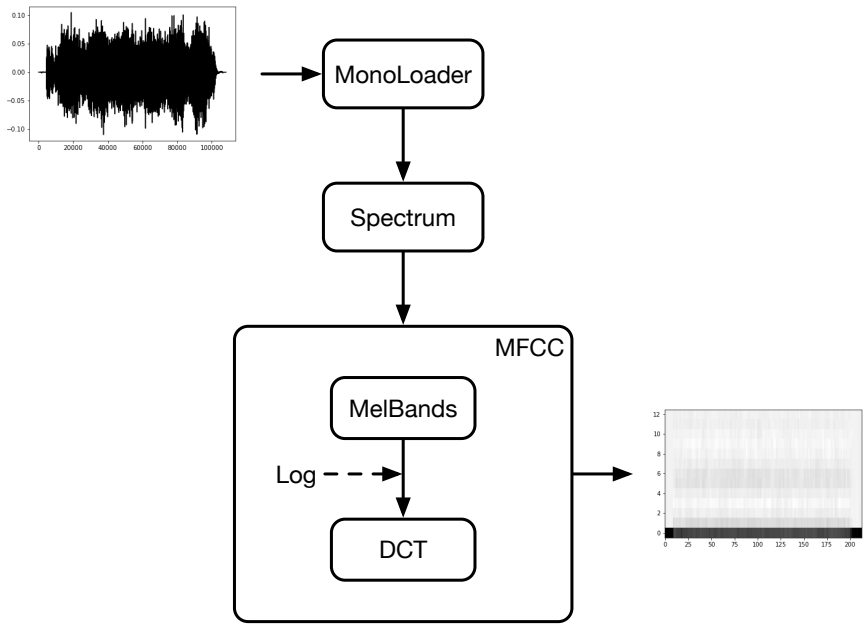



**Figure 5.9:** Signal Flow and Essentia Algorithm Graph for MFCC Computation

languages such as Pd and Max; objects (or algorithms) are connected virtually to each other, and more complex algorithms (composite algorithms) can be assembled by combining other algorithms with processing routines. This is actually the case with the MFCC algorithm, which as can be seen in the figure receives a spectrum, uses the MelBands algorithm to filter the spectrum, computes the logarithm then finally outputs the MFCC result from the DCT algorithm.

Interpreting exactly the meaning of each of the MFCC coefficients (Figure 5.10) is not as simple as is examining each bin of an FFT for instance. However, in most discourse dealing with MFCC analysis the emphasis appears to be on the first 13 coefficients as higher order coefficients are considered to contain increasingly redundant information (Logan, 2000). Additionally, the first coefficient is usually correlated most with the energy of the signal and is also frequently omitted in further usage. We follow suit also, since we are already computing energy using the specially designed Stevens' Power Law descriptor.

#### 5.3.2.5 BFCCs

If the MFCC feature uses the Mel scale for warping the spectrum, then the extremely related BFCC feature differs only in its warping procedure using the Bark scale. The Bark scale maps frequency to Barks using this formula:

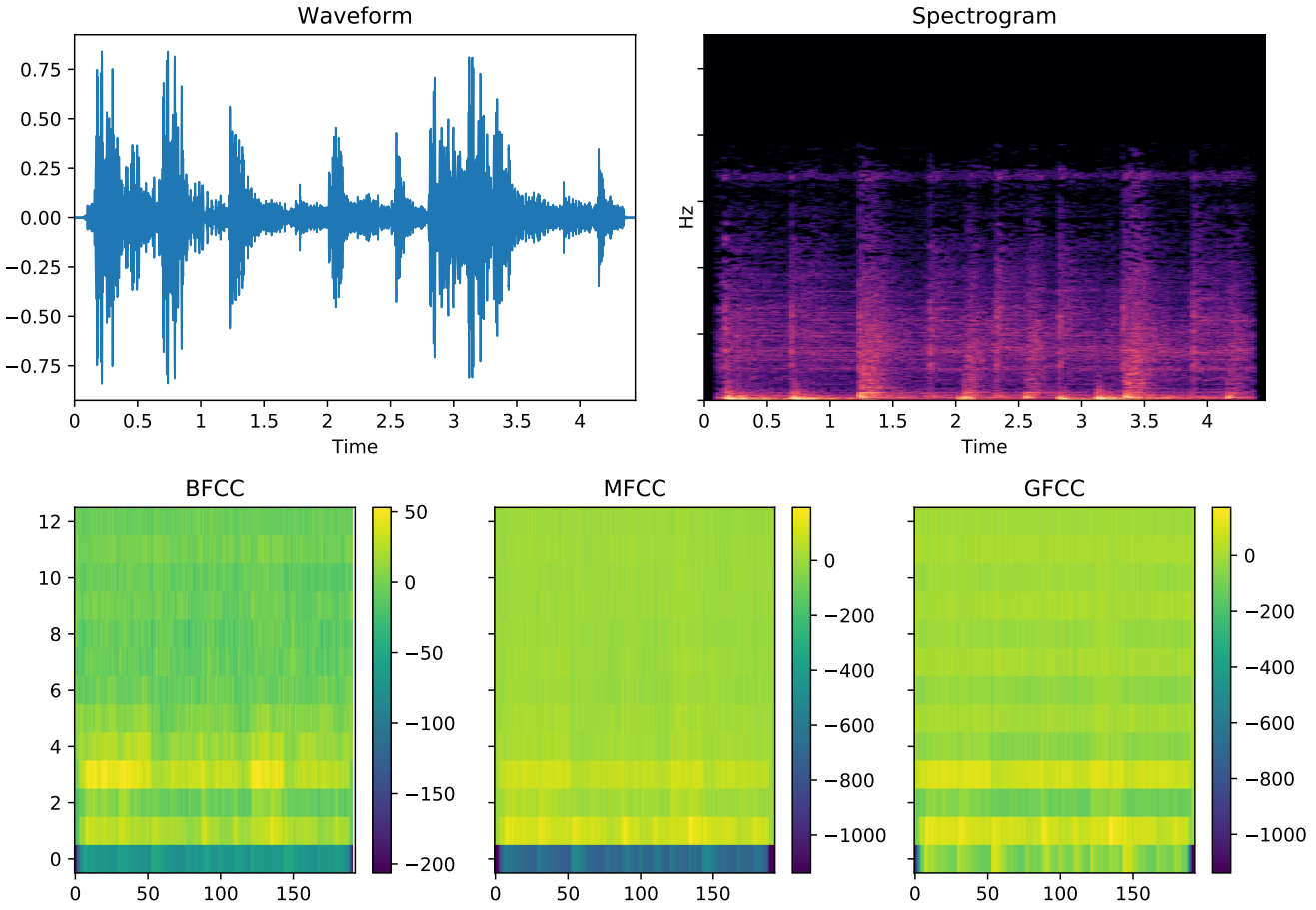



**Figure 5.10:** Waveform of drum break from "Amen Brother" (top-left). Power Spectrogram of waveform (top-right). BFCC coefficients 1-13 by frame number (bottom-left). MFCC coefficients 1-13 by frame number (bottom-centre). GFCC coefficients 1-13 by frame number (bottom-right)

| Library | Language | MFCC | BFCC | GFCC |
|---|---|---|---|---|
| Aubio (Brossier, 2006) | C & Pd | Yes | No | No |
| MIR Toolbox (Lartillot et al., 2008) | Matlab | Yes | No | No |
| LibXtract (Bullock, 2007) | Pd | Yes | No | No |
| TimbreID (Brent, 2010) | Pd | Yes | Yes | No |
| Essentia (Bogdanov et al., 2013) | C++ & Python | Yes | No | Yes |
| Librosa (Mcfee et al., 2015) | Python | Yes | No | No |
| Madmom (Böck et al., 2016) | Python | Yes | No | No |

**Table 5.4:** Availability of MFCC, BFCC and GFCC Descriptors in Common Audio Labelling Libraries.

$$b = [26.81 * \frac{f}{(1960 - f)}] - 0.53 \tag{5.19}$$

Aside from that the procedure is pretty much the same as for MFCC computation. Overlapping filterbanks tuned to the the bark scale are applied to the spectrum, the logarithm is applied and finally the DCT transform is performed to produce the coefficients.

The BFCC feature has not enjoyed the same level of popularity as its MFCC counterpart, most probably since it is not available as an extractor in the major feature extraction and analysis environments (Table 5.4).

The Mel scale is an extremely robust and reliable warping scale for many applications, but it is still prudent to experiment with other feature extractions methods depending on the application area its associated data. We decided to examine the applicability of the Bark scale following the findings of William Brent, who maintains that BFCCs are the most accurate single feature in his own experiments in percussive timbre identification (Brent, 2009a,b). His TimbreID toolkit, due to its ease of use and implementation within the Pure Data environment has been popular in creative applications, and as a consequence his suggested usage of the BFCC feature has been taken up in other works (Monteiro & Manzolli, 2011; Neupert & Gossmann, 2013; Tomás & Kaltenbrunner, 2014), including some tailored towards rhythm classification (Miron et al., 2013; Vogiatzoglou, 2016; Jathal, 2017), with the latter also reporting slightly better classification results over MFCCs. Considering the frequent rhythmic purposes of our own research, and the percussive, attack quality of dance-oriented electronic music in general, it suggests applicability in our system.

While a consolidated BFCC extractor doesn't exist in Essentia, a BarkBands filter does, and the algorithm structure in Figure 5.9 hints that all that is required is simply

```
type = 'power'
weighting = 'linear'
lowFrequencyBound = 0
highFrequencyBound = 8000
numberBands = 26
numberCoefficients = 13
normalize = 'unit_max'
dctType = 3
logType = 'log'
liftering = 22
```

**Table 5.5:** BFCC parameters for matching output of Ellis (2005)

swapping the two filter types, which we did. What was a bit more difficult was verifying if the extractor was then outputting correct coefficients and comparable coefficients compared to other implementations - due to the lack of existing implementations. However, Ellis (2005) has done some work on reproducing the feature of outputs of different implementations such as those found in Hidden Markov Model Toolkit (HTK) (Young et al., 2002) or the Auditory Toolbox (Slaney, 1998), and has some routines for producing BFCCs, so we decided to follow a similar procedure.

One of the first stumbling blocks came with the existing BarkBands filter, as it uses 27[64] pre-cooked Bark band frequencies that are hard-coded in the algorithm itself, rather than allowing parametric control of the lower and upper frequency bounds and the number of desired bands. To remedy this, we first created a TriangularBarkBands algorithm according to Ellis' specifications that provides precise parametric control of these crucial settings, in this way we can tweak settings to produce comparable feature outputs with Essentia. Next we modified a copy of the existing composite MFCC algorithm and swapped in the TriangularBarkBands to create the new BFCC extractor. After some experimentation we were able to achieve almost identical BFC coefficients as Ellis' "Rastamat"[65] implementation in Matlab (Figure 5.11) using the parameter settings in Table 5.5. With this implementation, and the existing MFCC and GFCC extractors, Essentia is now the only "off the shelf" feature extraction library that provides all three cepstrum descriptors.

[64]The standard number is typically 24, but Herrera et al. (2003) suggests, in order to increase accuracy in drum classification, increasing the resolution at lower frequencies by splitting lower order filters.

[65]http://www.ee.columbia.edu/ln/rosa/matlab/rastamat/

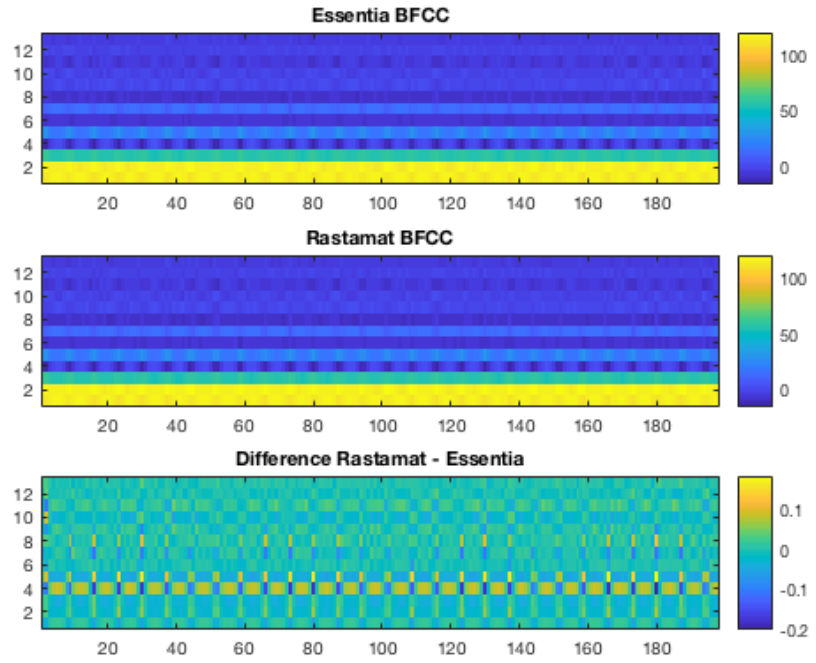



**Figure 5.11:** BFCC outputs with errors for Rastamat and our Essentia Implementations

#### 5.3.2.6 GFCCs

Our final warped cepstrum feature is the GFCC (or GTCC to give its alternative acronym by some investigators (Valero & Alias, 2012; Fathima & Raseena, 2013)). The GFCC derives its warping mechanism from the gammatone filter (Eq. 5.20), whose impulse response is purported to correlate closely with the auditory response of the cochlea in mammals (Patterson et al., 1987; Valero & Alias, 2012).

$$g(t) = at^{n-1}e^{-2\pi bt}cos(2\pi ft + \phi) \tag{5.20}$$

Here what we are seeing is basically a cosine waveform at a frequency $f$ with initial phase $\phi$ and amplitude $a$ modulated by the gamma function (the exponential component) at order $n$ with a bandwidth $b$. Practically speaking, they are implemented using as set of ERB rectangular filters tuned to the gammatone properties. The computation is slightly more involved than the Mel or Bark approaches but in Essentia, just like our port of the BFCC algorithm, these filters are modelled on a Matlab version made available in by Ellis (2009). who in turn formulated his solution as a response to the more computationally expensive routines available in Slaney's Auditory Toolbox (Slaney, 1998).

As Zhao & Wang (2013) note, the ERB scale has finer resolution at lower frequencies (where pitch information is more compressed perceptually) than the Mel scale used in MFCC computation. The other difference is the addition of a cubic root rectifier in place of the typical log. They posit that these adjustments over usual MFCC approaches provide more robustness in the case of speaker identification in speech processing tasks. Indeed, GFCC analysis has proved fruitful in many works dealing with speech (Abdulla, 2002; Schlüter et al., 2007), but has yet to prove widespread in musical or non-speech experiments. Recently however we have seen the introduction of GFCC feature extraction in singer identification (Cai et al., 2011), structural segmentation (Tian & Sandler, 2016), genre classification (Johnson-Roberson & Sudderth, 2017; Grekow, 2017), environmental sounds (Valero & Alias, 2012) with improvements over classical Mel-based scaling approaches in many cases.

#### 5.3.2.7 Tristimulus

An interesting, alternative viewpoint of timbre, first proposed by Pollard & Jansson (1982), takes inspiration from primary colour mixing model of light and attempts to model an equivalent for sound. Using information from the estimated fundamental frequency and harmonic peaks in the spectrum, the energy ratios of the first harmonic, the second, third and fourth harmonics and finally the rest of the harmonics form the three values of Tristimulus descriptor (Peeters, 2004).

Tristimulus is obviously very useful for visualisation, as we can map it directly to colour (Sequera, 2006), or to 3-dimensional space (Figure 5.12) without resorting to dimensional scaling. Also, in a more creative application of the feature, Fox & Carlile (2005) use tilt data in an accelerator equipped glove for Tristimulus-based formant and timbre control of a synthesiser.

Compared to more pitch "agnostic" timbral impressions such as cepstrum analysis, tristimulus features are more applicable for analysing signals with high degree of harmonicity, as Terasawa et al. (2006) note:

> *"[Tristimulus], however, is limited by the inability to represent inharmonicity and a rather arbitrary delineation of the three frequency components."*

For succint and effective analysis of a broad range of signals Tristimulus should likely be aggregated with descriptors that are more suited for capturing those attributes, such as inharmonicity or spectral moments (Peeters, 2004; Chudy & Dixon, 2010).

### 5.3.3 Musical Specific Descriptors

The features we have previously described and will evaluate in due course are extracted with the sole intention of matching target unit sounds with corpus units based on

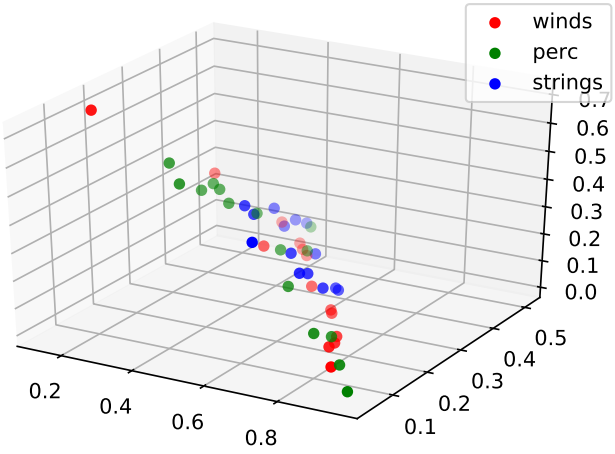



**Figure 5.12:** Tristimulus plot for 3 different classes of instruments, taken at random from the Philharmonica set

some perceived notion of "timbre". We distinguish here two features that are extracted with the intention of fulfilling more "musical" (of course timbre and music are inextricably intertwined) purposes such as:

1. Pitch Tracking: Returning units that contain the same pitch, or transforming/post-processing units to contain the same pitch.
2. Harmonic analysis: Returning units that containing similar harmonic profiles i.e. might contain similar tonality in terms of localised chord or overall key.

#### 5.3.3.1 Fundamental Frequency and Pitch

Fundamental Frequency (f0) detection or pitch tracking attempts to determine the lowest frequency within a signal, with the hopes that this matches the listeners perceptual impression of pitch (Gerhard, 2003). Algorithms for pitch detection exist in both the time and frequency domain. Time domain methods usually stem from some sort of autocorrelation of the signal, whereby the signal is compared with itself in increasingly delayed time lags to discover periodic repetitions whose reciprocal corresponds to the fundamental frequency given by $1/f = T$. Due to the harmonic nature of musical signals, autocorrelation methods often return octave errors, where multiples of the actual fundamental frequency are reported instead.

The well-known YIN[66] algorithm (de Cheveigné & Kawahara, 2002) addresses these errors and more by adding some correction stages such as absolute thresholding and

[66]The name is derived from the yin and yang principle of contrary forces in Chinese philosophy, as the algorithm itself is a tradeoff between autocorrelation and cancellation.

parabolic interpolation, and is available within Essentia. Brossier (2006) defines a YIN equivalent called PitchYinFFT in the frequency domain, taking advantage of convolution/multiplication equivalency to reduce some of the computation overhead inherent in the time domain method. Given the modular nature of feature extraction in Essentia, and since we are already computing the spectogram for cepstrum analysis, it makes sense that we use this version for our pitch tracking needs.

#### 5.3.3.2 Chroma and (H)PCPs

Just as the middle stage in the previous cepstrum analysis methods warped the frequency spectrum to a compressed non-linear scale, the chromagram feature also represents a collapse of the spectrogram (though without any further cepstrum transform), this time into musically relevant pitch classes based on Pitch-Class Profiles (PCP)s (Fujishima, 1999) or Harmonic Pitch-Class Profiles (HPCP)s (Gómez & Herrera, 2004) or chroma extractors (Orio, 2006). The resulting feature contains the integrated intensities of the spectrum with respect to the 12 semitones of the diatonic scale assuming equal temperament, although further ratio resolutions (return vectors of size 24 or 48 for example) of the semitone are also used.

HPCP computation, as proposed by Gómez (2006) is carried out as follow.

1. Perform the STFT
2. Locate peaks in the spectrum using a peak picking procedure as we saw in the onset detection stage (See Section 5.2.2.2).
3. Use the PCP procedure to map the spectral peaks to folded intensities in a set containing the desired number of pitch classes.
4. Normalise values based on the maximum.

HPCPs are less useful for timbral analysis[67], as it is not really their intended purpose. They are naturally effective features for systems that need to understand harmonic qualities such as chord recognition (Fujishima, 1999) and key detection (Gómez & Herrera, 2004). Indeed, within the GiantSteps project our colleagues have adapted customised key profiles that can extend to the stylistic peculiarities of tonality and key that pervade dance music (Faraldo et al., 2016, 2017a,b).

### 5.3.4 Feature Combination and Evaluation

To evaluate the ability of some of the features we have described here in classifying musically relevant timbres, we trained and tested two models with a number of classifiers. Our goal is not to provide a comprehensive work on timbral classification but rather serve as a broader sanity check to confirm two queries:

[67] Our classification experiment later on will add weight to this.

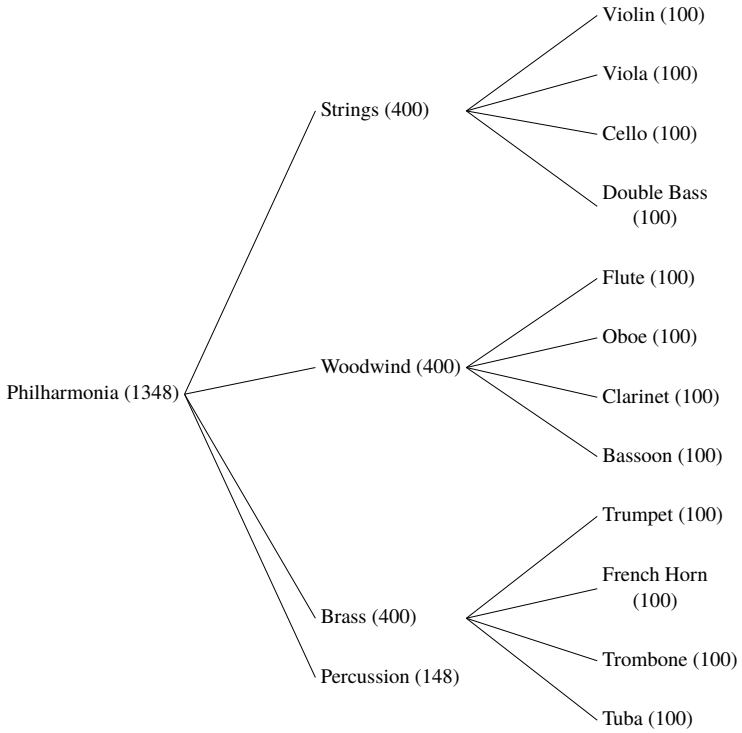



**Figure 5.13:** Summary Tree of Orchestral Samples for Classification

- Are the features we have chosen appropriate for timbral classification and similarity analysis, and more specifically, analysis for rhythmic and percussive sounds?
- Do more exotic Cepstrum warping procedures like our implemented BFCC and the GFCC outperform the more commonplace MFCC?

To address the requirements in our classification task, two distinct datasets were compiled. The first set focusses on sections and instruments of the orchestra and the second focusses on percussive timbres of a typical drum set or drum machine.

**Orchestral Dataset** The orchestral dataset comprises professionally recorded orchestral samples kindly made available for free download by London's Philharmonia Orchestra[68], and has been used in a number of existing works for both analysis (Hulshof et al., 2016; Donnelly & Sheppard, 2016; Pishdadian et al., 2017) and interactive performance (Miller & Hammond, 2010). The total number of sound files

[68] https://www.philharmonia.co.uk/explore/sound_samples

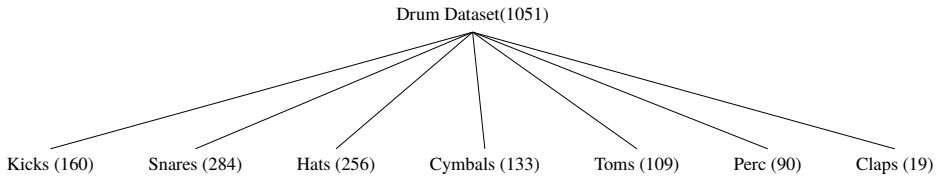



**Figure 5.14:** Summary Tree of Drum Samples for Classification

exceeds well over 10,000 so we restricted ourselves to the standard instrumentation of the three choirs (strings, woodwinds and brass) as well as percussion, and omitted extended instruments such as guitar or banjo. All 148 percussion samples were selected from the set, and ranged from typical instruments such as the bass drum and cymbals to more exotic items like the washboard and whip. Figure 5.13 shows a tree structure summary of the distribution of the samples across instrumentation and the respective sections. 200 samples from each of the instruments were taken at random, with a wide variety of dynamics and articulations.

**Drum Dataset** The drum dataset was compiled to address better the rhythmic goals of this thesis and comprises over 1000 one shot samples of drum sounds. The samples themselves were sourced from freely availably libraries on the internet including one large repository provided by the popular musician focussed website Music Radar[69]. We have found one article in the literature (Masood et al., 2015) that uses Music Radar samples for timbral classification of multiple instruments using a neural network. Our dataset contains a mixture of regular kit sounds and some percussion sounds from acoustic recordings as well as electronic drum machines like the Roland TR range. After verifying the trustworthiness with some random auditioning of the samples, we relied on the existing labelling in the filenames to divide up into appropriate classes for training and testing, which are summarised in the tree in Figure 5.14.

#### 5.3.4.1 Classifier Summary

For our classification and feature evaluation experiments we compared the performance of the various feature configurations on 3 distinct classifiers. We have relied on the availability of those available with "black-box" functionality in the scikit-learn (Pedregosa et al., 2012) library for Python, and apart from some minor adjustment of high-level parameters we have not gone too deep into their inner workings.

**Nearest Neighbours (*k*NN)** For any experiments involving supervised learning and classification of labelled observations, the *k*-Nearest Neighbours (*k*-NN) classifier

[69] http://www.musicradar.com/news/drums/sampleradar-1000-free-drum-samples-229460

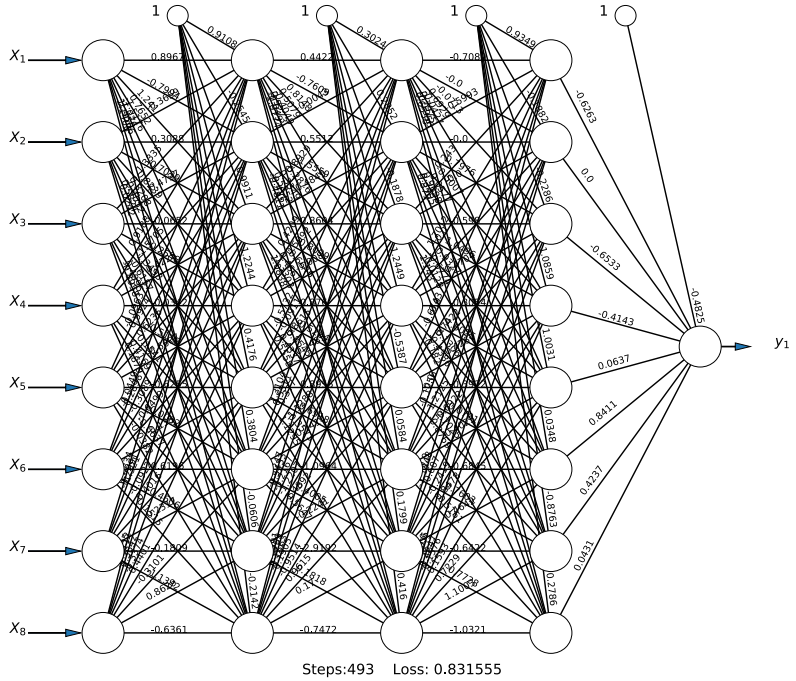



**Figure 5.15:** Input, Hidden and Output Layers for Multilayer Perceptron Classifier

should ideally be the first considered. *k*NN invokes a voting based system where $k$ is the number of neighbouring instances that decide which class a new observation belongs to based on some distance metric. *k*NN is attractive due to its conceptual simplicity, and has delivered favourable results in many timbre classification applications (Herrera et al., 2002, 2003; Herrera-Boyer et al., 2003; Mandel & Ellis, 2005b; Wang et al., 2006; Somerville & Uitdenbogerd, 2008). Unfortunately it is a lazy form of learning, meaning all computation needs to be carried out when a new observation needs to be classified, thus it might not scale sufficiently depending on the problem application. In our evaluation we choose $k = 1$ after experimenting with $k = 3, 5$ and not achieving better accuracy.

**Support Vector Machines** Support Vector Machines (SVM)s as a classification mechanism and in comparison to nearest neighbours methods are significantly more complex in their conception. In a nutshell, an SVM algorithm will try to find an idealised hyperplane that separates instances into multiclasses in multidimensional space (Chih-Wei Hsu, Chih-Chung Chang & Lin, 2008). At lower dimensionality, linear separation is not always possible, so a "kernel function" is used to artificially transform the problem set to higher dimensionality for easier separation between the classes (Xu et al., 2003). Luckily the availability of libraries such as LibSVM (Chang & Lin, 2011) (also integrated into scikit-learn) means this algorithm can be easily

| Feature Grouping | Features |
|---|---|
| Temporal (4) | loudness.mean (1), loudness.var (1) |
| | logattacktime.mean (1), logattacktime.var (1) |
| Spectral (4) | flatness.mean (1), flatness.var (1) |
| | logattacktime.mean (1), logattacktime.var (1) |
| BFCC (26) | bfcc.mean (13), bfcc.var (13) |
| GFCC (26) | gfcc.mean (13), gfcc.var (13) |
| HPCP (24) | hpcp.mean (12), hpcp.var (12) |
| MFCC (26) | mfcc.mean (13), mfcc.var (13) |

**Table 5.6:** Feature Configurations for Classifier Testing

incorporated into classification experiments. SVM have been used successfully in classification of Western and Chinese classical instruments (Liu & Xie, 2010), drum sounds (Herrera et al., 2002, 2003) as well as general instrument and timbre classification (Herrera-Boyer et al., 2003; Krey & Ligges, 2010; Agostini et al., 2003; Deng, 2008).

**Artificial Neural Networks** An Artificial Neural Network (ANN) mimics the pattern recognition capabilities of the human brain, specifically through the use of the multilayer perceptron. Multiple layers of digital neurons - TLUs that "fire" a value when a weighted sum of the inputs exceed a threshold value - are arranged in a fully-connected, feedforward (generally cycles aren't used) architecture (Russell & Norvig, 2002). They can be trained to learn from example through the use of "back-propagation", a procedure that determines the error and distributes the blame back through the network so the weights can be adjusted to finally settle on minimal error (Gurney, 1996).

Figure 5.15 shows an example topology of a network trained with 3 hidden layers of 8 perceptron units for each of the 8 features chosen from the temporal and spectral feature classes (mean and variance for loudness, log-attack time, spectral centroid and flatness). Artificial neural networks are sensitive to feature scaling and disparate ranges, thus we standardised each feature for mean centred at $\mu = 0$ and standard deviation at $\sigma = 1$.

#### 5.3.4.2 Results

To determine the interaction and ability of different feature sets we divided the base level features into temporal, spectral, harmonic and timbral groupings and experi-

| | *k*-NN | SVM | ANN |
|---|---|---|---|
| Temporal | 0.21 (+/- 0.03) | 0.24 (+/- 0.03) | 0.27 (+/- 0.05) |
| Spectral | 0.50 (+/- 0.06) | 0.49 (+/- 0.04) | 0.55 (+/- 0.03) |
| Temporal+Spectral | 0.62 (+/- 0.03) | 0.54 (+/- 0.06) | 0.67 (+/- 0.06) |
| BFCC | 0.89 (+/- 0.04) | 0.93 (+/- 0.04) | 0.93 (+/- 0.03) |
| BFCC+Temporal+Spectral | 0.91 (+/- 0.03) | 0.94 (+/- 0.03) | 0.95 (+/- 0.03) |
| GFCC | 0.91 (+/- 0.03) | 0.95 (+/- 0.02) | 0.95 (+/- 0.03) |
| GFCC+Temporal+Spectral | **0.92 (+/- 0.03)** | **0.96 (+/- 0.02)** | **0.96 (+/- 0.03)** |
| HPCP | 0.45 (+/- 0.04) | 0.44 (+/- 0.05) | 0.42 (+/- 0.05) |
| HPCP+Temporal+Spectral | 0.58 (+/- 0.04) | 0.69 (+/- 0.06) | 0.69 (+/- 0.05) |
| MFCC | 0.86 (+/- 0.03) | 0.90 (+/- 0.04) | 0.91 (+/- 0.04) |
| MFCC+Temporal+Spectral | 0.89 (+/- 0.02) | 0.93 (+/- 0.03) | 0.93 (+/- 0.03) |

**Table 5.7:** Mean Classification Accuracy and 95% confidence interval for each classifier and feature configuration with the Orchestral Dataset

mented with a number of combinations of these groupings (Table 5.6). In terms of preprocessing all features were normalised using min-max normalisation, except for the neural network which seemed to respond better to data standardisation versus normalisation. For each classifier 10-fold cross validation was performed, then the results were aggregated and reported along with their 95% confidence interval.

Table 5.7 shows the accuracy results for each classifier applied to the various configurations of descriptors when run on the orchestral dataset. The neural network and SVM returned the highest classification accuracy run on the whole collection of features using GFCC cepstral analysis.

Table 5.8 shows the accuracy results for each classifier applied to the various configurations of descriptors when run on the drum dataset. *k*-NN returned the highest classification accuracy once again with the whole collection of features using GFCC cepstral analysis.

The accuracy of classification is lower overall for our percussion dataset, which we can attribute as follows. Intuitively, the wider palette of sounds within an orchestra given the timbre of different instruments, dynamic ranges, articulations and so forth are bound to exhibit greater variance in the feature statistics that can aid us in separating the instances into their correct classes more easily. Looking at the confusion matrix (Figure 5.16) for one of the test validations (GFCC+Temporal+Spectral trained with the SVM) we can see where the trends of the false positives lie. For example

| | *k*-NN | SVM | ANN |
|---|---|---|---|
| Spectral | 0.61 (+/- 0.08) | 0.56 (+/- 0.10) | 0.62 (+/- 0.12) |
| Temporal | 0.37 (+/- 0.10) | 0.40 (+/- 0.08) | 0.44 (+/- 0.11) |
| Temporal+Spectral | 0.66 (+/- 0.10) | 0.59 (+/- 0.09) | 0.69 (+/- 0.11) |
| BFCC | 0.86 (+/- 0.05) | 0.86 (+/- 0.07) | 0.83 (+/- 0.08) |
| BFCC+Temporal+Spectral | 0.85 (+/- 0.07) | **0.86 (+/- 0.06)** | **0.85 (+/- 0.09)** |
| GFCC | 0.86 (+/- 0.08) | 0.84 (+/- 0.08) | 0.84 (+/- 0.11) |
| GFCC+Temporal+Spectral | **0.87 (+/- 0.08)** | 0.86 (+/- 0.09) | 0.85 (+/- 0.12) |
| HPCP | 0.53 (+/- 0.17) | 0.55 (+/- 0.14) | 0.50 (+/- 0.14) |
| HPCP+Temporal+Spectral | 0.64 (+/- 0.12) | 0.71 (+/- 0.08) | 0.69 (+/- 0.08) |
| MFCC | 0.83 (+/- 0.10) | 0.84 (+/- 0.10) | 0.84 (+/- 0.09) |
| MFCC+Temporal+Spectral | 0.83 (+/- 0.09) | 0.85 (+/- 0.07) | 0.85 (+/- 0.04) |

**Table 5.8:** Mean Classification Accuracy and 95% confidence interval for each classifier and feature configuration with the Drum Dataset

we some misclassification between toms and kicks which is to be expected - they are both lower frequency membranes without wires (unlike the snare). Also there is some misclassification of hi-hats and cymbals which are all broadband in their spectral profiles. It is also important to realise that, in drum machines at least, these sounds are generated using the same limit set of modules of oscillators and filters and so on, and I would challenge the most astute listener to blindly discriminate between many synthesised claps, snares, or hi-hats generated by certain vintage drum machines.

From these two experiments we can draw some conclusions. Firstly, cepstral methods alone are extremely robust descriptors on their own, with temporal and spectral additions to the configurations only contributing a few percent improvement to the overall accuracy scores in all cases. However, combining the full set of features does return the highest classification accuracy for both datasets. The second conclusion that we can draw is that alternative approaches to cepstral analysis using the Bark scale and Gammatone/ERB filterbanks do improve over classical Mel approaches. Finally of these two cepstral methods, the GFCC based has slight advantage over BFCC, and probably for good reason - its gammatone filter was first proposed as an improvement to existing Critical Band scaling methods and supposedly more grounded in models of the auditory filter system. In turn, the Bark scale itself was proposed to address certain limitations with the Mel scale, and our results reflect this progression. We would stress that these methods be given serious consideration in any classification

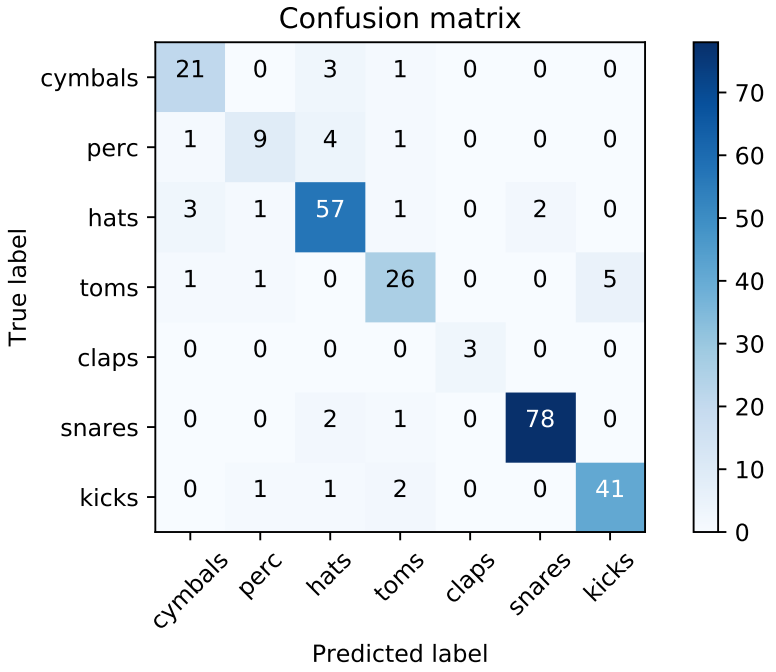



**Figure 5.16:** Confusion Matrix for Misclassification in Drum Dataset

experiment for comparison and raising their reputation amongst wider research.

## 5.4 Combining Sounds Algorithmically - Unit Selection

Unit selection solves the the problem of determining what sounds to select from the corpus and the systematic structuring of the selected sounds for outputting logical concatenated sequences, or as Schwarz defines:

> "*[Concatenative sound synthesis uses] a unit selection algorithm that finds the sequence of units that match best the sound or phrase to be synthesized, called the target. The selection is performed according to the descriptors of the units, which are characteristics extracted from the source sounds, or higher level descriptors attributed to them.*"
>
> (Schwarz, 2006)

Many unit selection schemata have been proposed and there exists no standard or best method. However some specific procedures have presented themselves repeatedly which we will summarise here:

### 5.4.1 Linear Search

At the most basic level a linear search criteria for unit selection simply computes the (dis)similarity of every unit in the target sequence with every possible unit in the corpus, according to some distance measure (Schwarz, 2011). For example, a weighted variation of Euclidean could be applied (taking care to normalise the feature vectors) that would allow the composer to place emphasis on certain features in the similarity computation as given by Eq. 5.21.

$$d(t^i, c^j) = \sqrt{\sum_{k=0}^{K-1} w_k (t_k^i - c_k^j)^2} \tag{5.21}$$

Here $d(t^i, c^j)$ refers to the distance or dissimilarity between a target unit $t^i$ and a corpus unit $c^j$, index $k$ represents the individual feature value from the full set of $K$ features and $w_k$ represents a weighting to be applied to that feature during the distance computation. The closest corpus unit $s$ that should be selected for concatenation is then given by Eq. 5.22.

$$s = \underset{0<j<N}{\arg\min}\ d(t^i, c^j) \tag{5.22}$$

It is a conceptually simple and robust technique that has been applied in numerous systems (or at least non-weighted Euclidean) (Dannenberg, 2006; Maestre et al., 2006; Sturm, 2006; Collins, 2007a; Stoll, 2013). In fact, for ease of implementation, we have applied the linear search method ourselves in developing a real-time system for concatenative synthesis of rhythms as described in (Ó Nuanáin et al., 2016b, 2017a), which we examine in more depth in Chapter 6. The overlying problem with this approach is that it only considers the disparity between the target and the corpus unit, and neglects treating the continuity or context of consecutive units within the output sequence. Depending on the size of the corpus it may also be prohibitively slow (but not in our informal usage, at least not compared to HMMs), which can be remedied by applying a structure such as a $k$-d Tree.

#### 5.4.1.1 Faster Search with $k$-D Trees

As Collins (2007a) and Dannenberg (2006) have noted, the brute force nature of exhaustive linear search computation may not lend itself to scalability for larger datasets. The former suggests using $k$-D Tree structures for more strategically optimising the search process, and has also been utilised in systems by Schwarz et al. (2009), Einbond & Schwarz (2010), Stoll (2013) and Klügel et al. (2014).

A $k$-D Tree organises points in space with a binary tree by successivly cycling through each dimension of the problem space (its full name is $k$-Dimensional Tree), splitting the data at its median and assigning the two newly partitioned sets to the branches

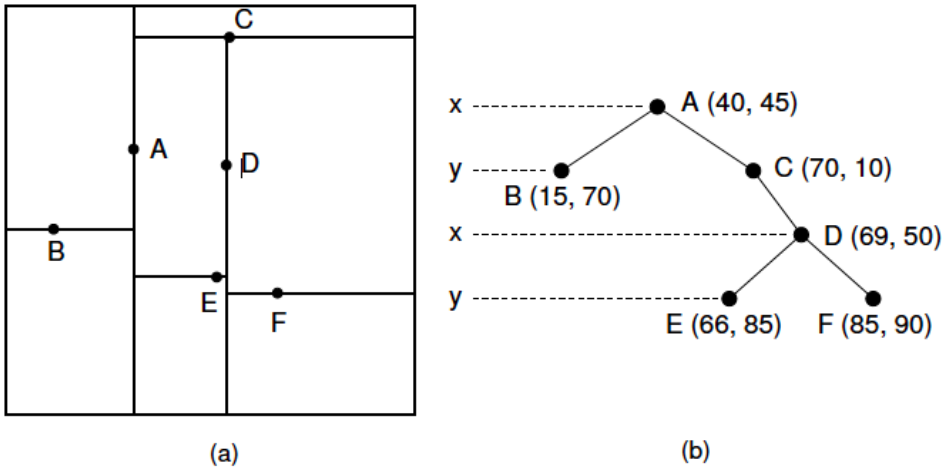



**Figure 5.17:** Hyperplane division (left) and binary tree (right) arrangement of 2-dimensional dataset with a *k*-D Tree of depth 4. The origin is the top left and the size of the grid is 128x128 units. Image from Indiana University.

of the nodes at each level. The branching aspect of binary search trees allows us to eliminate vast portions of the search space when performing nearest neighbour searches.

It is important to realise that they are only approximate solutions so they may not return the actual nearest neighbour (which is in an important factor to consider if it is to be used for accurate classification). A new instance may be routed down a branch too hastily based on its proximity in a single dimension even though the closest match based on the combined Euclidean distance may belong to a different branch. Also, as Russell & Norvig (2002) stress, *k*-D Trees are only useful so long as the number of examples exceeds the number of dimensions in the problem space, otherwise it is no more efficient than exhaustive, linear scanning. Many *k*-D Tree inspired Approximate Nearest Neighbour (ANN) algorithms exist for dealing with high dimensionality problem domains, at a possible sacrifice in accuracy (it may not return the exact nearest neighbour). For instance in OpenCV a fast approximate *k*-D Tree is implemented using a Best-Bin-First algorithm (Kaehler & Bradski, 2016). In PyConcat we avail of the implementation in the SciPy scientific package for python (Jones et al., 2014).

### 5.4.2 Unit Selection with Viterbi Decoding of Hidden Markov Models

The Viterbi algorithm was first applied by Hunt for the purposes of speech synthesis by performing unit selection of speech phonemes samples from a prior corpus (Hunt & Black, 1996). It was then adopted for musical purposes by Diemo Schwarz in his Caterpillar System (Schwarz, 2003). By representing a unit selection system as

a Hidden Markov Model we can not only consider the disparity between the target and corpus unit (encoded in the emission matrix) but also the "best fit" of continuity between two consecutive units in the output sequence as determined by the transition matrix of the hidden states (Figure 5.17).

As with shortest path problems, unit selection requires finding the *minimisation* of a set of *costs* (Hunt & Black, 1996), namely the target cost $C^t$ between target and corpus units (Eq. 5.23, which is an identical derivation to Eq. 5.21 in the linear search method) and the concatenation cost $C^c$ (Eq. 5.24) between the consecutive concatenated units (Schwarz, 2003). This is in contrast to usual goal of Markov processes which of course is concerned with *maximisation* of probabilities.

A linear combination is finally performed with weights $w^t$ and $w^c$ to give the total cost $C$ (Eq. 5.25) . The constituent costs themselves are derived by computing the dissimilarity between the feature vectors of the associated units using a suitable distance metric and a set of $w_k$. The feature sets and accompanying weights can (and typically do) differ for the two different cost functions.

$$C^t(t^i, c^j) = \sqrt{\sum_{k=0}^{K-1} w_k (t_k^i - c_k^j)^2} \tag{5.23}$$

$$C^c(c^j, c^j - 1) = \sqrt{\sum_{k=0}^{K-1} w_k (c_k^j - c_k^{j-1})^2} \tag{5.24}$$

$$C = w^t.C^t + w^c.C^c \tag{5.25}$$

One of the motivations for working with more complex unit selection schemes like HMMs is that we can specify what we consider important for target cost computation separately to the concatenation cost computation by weighting accordingly or even choosing completely different features sets. For example, in speech synthesis we might choose features and weightings for the target cost that prioritises matching of length and linguistic context versus more prosodic configurations for the concatenation cost that encourage stability of energy and pitch.

### 5.4.3 Constraint Satisfaction

Schwarz notes, however, that the HMM approach can be quite rigid for musical purposes because it produces one single optimised sequence without the ability to manipulate the individual units. To address these limitations, he reformulates the task into a constraint-satisfaction problem, which offers more flexibility for interaction. A constraint-satisfaction problem models a problem as a set of variables, values, and a set of constraints that allows us to identify which combinations of variables and values are violations of those constraints, thus allowing us to quickly reduce large portions of the search space (Nierhaus, 2009).

Zils & Pachet (2001) first introduced constraint satisfaction for concatenative synthesis in what they describe as musical mosaicking - or, to use their portmanteau - musaicing. They define two categories of constraints: segment and sequence constraints. Segment constraints control aspects of individual units (much like the target cost in an HMM-like system) based on their descriptor values. Sequence constraints apply globally and affect aspects of time, continuity, and overall distributions of units. The constraints can be applied manually by the user or learned by modelling a target. The musically tailored "adaptive search" algorithm performs a heuristic search to minimise the total global cost generated by the constraint problem. One immediate advantage of this approach over the HMM is the ability to run the algorithm several times to generate alternative sequences, whereas the Viterbi process always outputs the most optimal solution.

## 5.5 Extensions to Hidden Markov Model-Based Unit Selection

The singular output issue that Schwarz raised as a result of HMM-based unit selection piqued our interest as we were delving into the specifics of unit selection of concatenative synthesis ourselves. While it seems like an ideal mechanism for capturing the deep connectionist and interdependent arrangement of units that a successful and natural synthesis should ideally exhibit, it also runs counter productive to the very musical goals of creating and exploring heterogenous variations of a motif or loop.

To solve this limitation we propose a new method of HMM-based unit selection that can return the $k$-Best concatenated sequences given a target, a corpus of sounds and the number of desired sequences $k$ (Ó Nuanáin et al., 2017c). Before this can be introduced however, we need to examine HMMs and the Viterbi algorithm in more depth, which until now hasn't been treated with sufficient rigour.

### 5.5.1 Hidden Markov Models

Recall from Chapter 3 that a Markov chain is a probabilistic system that satisfies the Markov property of memorylessness. It consists of a set of discrete states with each state having an associated probability weighting of moving to every other state in the system. At any point in discrete time, the probability of a future event is solely determined by the current state of the system without knowledge of past events (see Eq. 3.1).

A hidden markov model $\lambda = (A, B, \pi)$ extends the concept of a Markov chain by considering the transition states as hidden (Rabiner, 1989). The hidden states have a transition matrix $A$ as before, but each hidden state also emits an observable symbol from a set of symbols that have a probability distribution encapsulated in an emission matrix $B$. Finally, to initiate the HMM there also exists the initial probability distribution $\pi$, which determines the probability of which state to commence.

The subtleties of Hidden Markov Models don't become immediately clear until we start working with some common problems and especially their role in sound and music analysis and generation, but there are three traditional problems that are usually studied, as identified by Rabiner in his tutorial (1989).

1. *Evaluation* - Given an observable sequence of emissions $O = (O_1, 0_2, O3, ..., O_T)$, what is the probability that the sequence was generated by a certain model $\lambda$. This is particularly useful when benchmarking or comparing different models.

2. *Decoding* - Given an observable sequence of emissions $O = (O_1, 0_2, O3, ..., O_T)$, what is the most like sequence of hidden states that produced that observation $O$.

3. *Learning or Training* - Given a model with its parameters $\lambda = (A, B, \pi)$ and an observable sequence of emissions $O = (O_1, 0_2, O3, ..., O_T)$, how do we adjust the parameters to maximise the probability of the observable sequence $O$.

### 5.5.2 The Viterbi Algorithm

The Viterbi algorithm solves the decoding problem in HMMs (Rabiner, 1989), namely, for a given observation sequence $O = (O_1, 0_2, O3, ..., O_T)$ we wish to determine the highest probable hidden state sequence $S = (S_1, S_2, S3, ..., S_T)$ that would produce output $O$ given by:

$$S* = argmax(P(S|O)) \tag{5.26}$$

A brute force solution applied to a $T$ observations over $N$ states would involve computing all the cartesian products of the possibilities; $N^T$ involving exponential time complexity. Viterbi's algorithm enables us to reduce this complexity to $O(T.N^2)$, using dynamic programming techniques. Rather than exhaustively computing all the possibilities, we maintain two data structures alpha ($\alpha$) and phi($\phi$). At any point $t$ in the sequence to be decoded, we store the score of the maximum probability for emitting the observed symbol for each hidden state, along with the index or argument of the maximum probable state that led there. To get the optimal state sequence we get the index of the final highest scoring hidden state and backtrack through the *phi* structure beginning with that index, returning the accumulated list. We can express this formally in the recurrence expression 5.27-5.30.

In the initialisation of the algorithm we use the initial probabilities and the observed symbol to calculate the starting probabilities of each hidden state in $\alpha$, and, as there can be no previous states, $\phi$ is set to 0.

The recursion step continues until $T$, the length of the observed sequence. We first calculate the maximum of the probability of each previous state multiplied by the

transition probability to the current state. The winning maximum probability is multiplied by the emission probability of the observed symbol and stored in $\alpha$ while the index of that winning previous state is logged in $\phi$.

The elegance of the Viterbi algorithm should be apparent here. Rather than computing the $N^{t-1}$ possible combinations of all the previous states, dynamic programming is used to store the result of the calculations in a matrix for later retrieval.

When recursion halts, the highest final probability is computed and its index is used finally the backtrack through the state machine and thus returning $S^*$, the sequence of highest scoring hidden states that most likely produced the observed sequence $O$.

**1) Initialisation:** $t = 1$

$$\begin{aligned} \alpha_1(i) &= \pi_i B_i(O_1) \qquad 1 < i < N \\ \phi_1(i) &= 0 \end{aligned} \tag{5.27}$$

**2) Recursion:** $t = 2,...,t = T$

$$\begin{aligned} \alpha_t(j) &= max_{i \in N}(\alpha_{t-1}(i) A_{ij} B_j(O_t)) \qquad 1 < j < N \\ \phi_t(j) &= argmax_{i \in N}(\alpha_{t-1}(i) A_{ij}) \qquad 1 < j < N \end{aligned} \tag{5.28}$$

**3) Termination:**

$$\begin{aligned} p^* &= max_{i \in N}(\alpha_T(i)) \\ S_T^* &= argmax_{i \in N}(\alpha_T(i)) \end{aligned} \tag{5.29}$$

**4) Backtracking:** $t = T - 1,...,t = 1$

$$S_T^* = \phi_t + 1(S_{t+1}^*) \tag{5.30}$$

The Viterbi algorithm has been around for quite a while now, and implementations exist in many frameworks and programming languages (Bird, 2016; Guéguen, 2005). It is a fundamental technique in bioinformatics and natural language and speech processing. PoS tagging for example, uses a HMM trained on a large corpus of text with each word labelled with its constituent part of speech (e.g. noun, verb, article etc.). It proves more robust results in identifying new samples of text, as the transition matrix helps determine the context of each word within the sequence.

We have provided a reference implementation of the single output probabilistic Viterbi as part of our ongoing $k$-Best Viterbi work[70]. An important caveat in the implementation of HMM software is the possibility of numerical underflow due to multiplication of ever decreasing minute probabilities, for this reason programmers

```
O = ('normal', 'cold', 'dizzy')
S = ('Healthy', 'Fever')
π = {'Healthy': 0.5, 'Fever': 0.4}
A = {
'Healthy' : {'Healthy': 0.5, 'Fever': 0.4}
'Fever' : {'Healthy': 0.4, 'Fever': 0.6}
}
B = {
'Healthy' : {'normal': 0.5, 'cold': 0.4, 'dizzy': 0.1}
'Fever' : {'normal': 0.1, 'cold': 0.3, 'dizzy': 0.6}
}
```

**Table 5.9:** HMM parameters for Wikipedia Viterbi decoding example

typically convert to log-space for allowing addition instead, which we also allow the option of.

Figure 5.18 shows a trellis diagram showing the highlighted trace of the decoded path ($S = (0,0,1)$), for the observed sequence $O = (0,1,2)$ (or 'normal', 'cold' and 'dizzy' to give it its string labels) along with its calculations, computed by the Viterbi algorithm through the state space of the example problem in the Wikipedia article on Viterbi decoding [71]. The parameters of the model are given in Table 5.9. Notice how the dynamic programming stage intuitively computes and stores the necessary probabilities for each state at each time step through the trellis.

### 5.5.3 HMMs in Musical Applications

HMMs' facility for pattern recognition has been exploited for computational musical tasks. Score following, for instance, tries to consolidate the position of a live performance of a musical piece with its score representation automatically (Orio et al., 2003). Using Viterbi decoding, an alignment can be established by extracting features for the observed live performance, and comparing them against idealised features within the model to return the expected location of the performance within the score.

HMMs also lend themselves quite naturally to the task of chord recognition (Papadopoulos & Peeters, 2007; Cho et al., 2010; Sheh & Ellis, 2003). The former demonstrate a method whereby they compare the PCP representation of the signal corresponding to 24 possible chord labels (12 notes for major and minor) indicated by the emission matrix, coupled with the most probable chord sequence defined in the

[70] https://github.com/carthach/kBestViterbi/blob/master/kBestViterbi.py
[71] https://en.wikipedia.org/wiki/Viterbi_algorithm

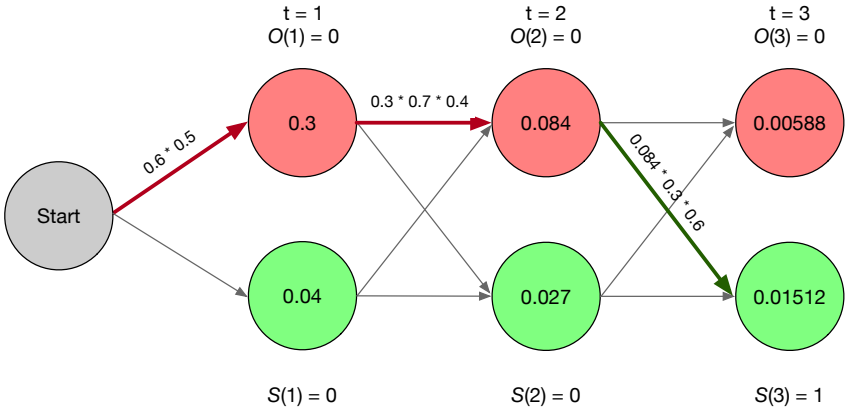



**Figure 5.18:** Viterbi decoding of Wikipedia example using our decoder

transition matrix, derived from prior musical knowledge or training on musical scores and transcriptions.

Compared to Markov chains, HMMs have been exploited somewhat less in generative or compositional applications (apart from concatenative synthesis, which we will describe presently). Some methods however are summarised in (Fernández & Vico, 2013) and (Nierhaus, 2009), with the latter making the observation that "when applied to algorithmic composition, HMMs are appropriate to add elements to an existing composition".

### 5.5.4 Adjusting Viterbi to Handle Unit Selection in Concatenative Synthesis

As already stated, Markov systems try to maximise probabilities while concatenative synthesis systems try to minimise cost. The costs should be pre-computed by calculating the target cost $C^t$ for every target unit $t_i$ and database unit $c_j$, and the concatenation cost $C^c$ for every combination of unit $c_j$. These costs form the emission matrix $A$ and transition matrix $B$ parameters for our HMM respectively. We can then discard the initial probability matrix $\pi$ and reformulate the Viterbi algorithm with the necessary changes for cost minimisation as in expression 5.31-5.34. Note that the recursion and termination steps now use the *min* and *argmin* function and sum the final costs.

**1) Initialisation:** $t = 1$

$$\begin{aligned} \alpha_1(i) &= B_i(O_1) \qquad 1 < i < N \\ \phi_1(i) &= 0 \end{aligned} \tag{5.31}$$

**2) Recursion:** $t = 2, ..., t = T$

$$\alpha_t(j) = min_{i \in N}(\alpha_{t-1}(i) + A_{ij} + B_j(O_t)) \qquad 1 < j < N$$
$$\phi_t(j) = argmin_{i \in N}(\alpha_{t-1}(i) + A_{ij}) \qquad 1 < j < N \tag{5.32}$$

**3) Termination:**

$$p^* = min_{i \in N}(\alpha_T(i))$$
$$S_T^* = argmin_{i \in N}(\alpha_T(i)) \tag{5.33}$$

**4) Backtracking:** $t = T - 1, ..., t = 1$

$$S_T^* = \phi_t + 1(S_{t+1}^*) \tag{5.34}$$

### 5.5.5 Exploring Alternatives in HMMs

One of the most thorough and cited articles dealing with $k$-Best or, "List Decoding" as the author describes it, has been provided by (Seshadri & Sundberg, 1994) who proposes two different methods to extend the regular Viterbi algorithm to multiple output. These methods he clarrifies as operating in parallel or serial. We summarise these methods here, and provide an implementation of the parallel approach for unit selection and concatenative synthesis.

As we were implementing the Parallel Decoder, we begin to notice the comparisons between the search process Viterbi takes through the hidden state space in a HMM and the way that shortest path algorithms such as Dijkstra's find the path of least resistance through an acyclic directed graph, as we already saw in the computation of the direct-swap distance in Chapter 3. In fact there are methods in graph research that can retrieve the k-Shortest Paths sequentially. To this end, we also demonstrate a method of reformulating a cost-based HMM used in concatenative synthesis as a directed acyclic graph in order to avail of shortest path methods.

#### 5.5.5.1 Parallel Decoder

The Parallel decoder (Seshadri & Sundberg, 1994) or List Viterbi decoding Algorithm (LVA) is so-called because, instead of keeping track of the winning path leading into each state at time step $t$, it now stores the top $k$ paths leading into the states in one pass through the state space or trellis. This is distinct from the serial decoder which takes several passes through the trellis in order to return multiple candidates.

To convert the regular Viterbi decoder to a $k$-Best Parallel decoder. the following steps are required.

1. Expand the $\alpha$ and $\phi$ matrices of the HMM to account for $T * N * K$ entries that correspond to $T$ discrete steps, $N$ states and $K$ sequences.

2. When computing the probabilities of the transitions from the previous states leading into the current state as per the recursion step in the equation, insert them into a continuously sorting data structure like a priority queue[72] (we use Python's *heapq*)

3. Remember that there is now the possibility of a previous state leading to the current state more than once (because that previous state now also has $k$ previous states that are potentially high probability). Thus we need to another structure (such as a dictionary) to keep track of the ranking of multiple probabilities of the same previous state if it enters the current state more than once.

4. At the termination step insert the final probabilities into the queue to find the final top $k$ scoring paths.

5. Backtrack as before while taking care to adhere to the ranking of multiple instances of the same previous state.

Curiously, we couldn't find fully-formed implementations exist for the Parallel decoder, at least not in any usable state of reproducibility, so once again we have implemented and made available ourselves a version in Python. Figure 5.19 shows how our newly implemented Parallel decoder computes the first and second highest probability paths ($k$ = 2) through the Wikipedia example previously evaluated with the regular Viterbi decoder in Figure 5.19.

#### 5.5.5.2 Serial Decoder

The Serial Decoder, also proposed by Seshadri & Sundberg (1994), differs from the Parallel decoder just implemented in that it first computes the single best state sequence using the regular Viterbi decoder, then returns the remaining $k$ one by one. Despite the less efficient connotations that the serial versus parallel labelling musters up, the authors maintain it can perform less computations as "the $k^{th}$ best candidate is computed only when the previously found $k$ - 1 candidates are determined to be in 'error"' (Seshadri & Sundberg, 1994).

We have not implemented the Serial decoder (yet - see Chapter 8 for future work!), as once again no reference code is available and its derivation is a bit harder to decipher from the paper than in the case of its parallel equivalent.

#### 5.5.5.3 *k*-Shortest Paths

We hinted that the traversal of a state space in a HMM bears considerable resemblance to that of a Directed Acyclic Graph (DAG) of vertices, edges and a set of

[72] A heap queue is a binary tree with the special condition that every parent has a value less than or equal to that of its children (this is a minimum queue, a maximum is naturally the inverse). The important function in our case is the push function, which adds items to the tree and maintains the sorted heap property in $O(logn)$ time.

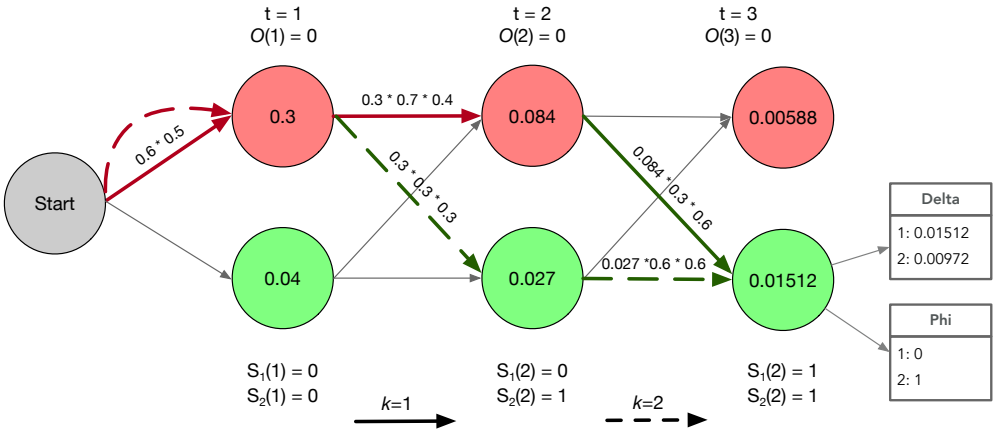



**Figure 5.19:** Viterbi decoding of Wikipedia example using the Parallel decoder. Final delta and phi matrices indicate the top $k$ highest scoring probabilities and the state indices that led there. As the two previous states are different we don't need to consider the ranking here. The path is the solid line through the trellis, while $k$=2 is indicated by the dashed line.

costs to be associated with those edges. This should also be visually apparent from the trellis diagrams presented in Figure 5.18 and Figure 5.19.

Shortest path algorithms aim to solve the problem of finding the the path between two given vertices with the least cost. One of the most well-known algorithms for achieving the shortest path are those by Dijkstra or Bellman and Ford (Russell & Norvig, 2002). Computing the $k$-Shortest paths then entails returning an ordered *list* of the shortest paths between the two desired vertices (from what is often called the *source* to the *sink*).

One of first algorithms proposed for tackling $k$-Shortest path problems is known as Yen's Algorithm (Yen, 1971). Yen's algorithm only operates on simple or loopless graphs (like DAGs or trellis diagram representations of a HMM) and uses a regular shortest path algorithm such as Dijkstra or Bellman-Ford in order to find the best $k$=1 shortest path initially. Assuming the previous $k-1$ paths have already been found, the algorithm searches the previous path for branches that deviate with higher associated cost. It is for this reason we need to have the best initial shortest path from an existing shortest path algorithm. Thankfully many implementations exist for $k$ Shortest Path routing so we don't need to worry about its inner workings too greatly and can rely on its black-box functionality in many graph-based programming libraries. We avail of Yen's Algorithm in Python using the the NetworkX Hagberg et al. (2008) graph library.

With the availability of $k$ shortest path routing methods and the already identified similarity between a HMM trellis and DAG we propose here a way to reformulate a HMM trellis as a DAG for $k$-Best Viterbi decoding with $k$-shortest paths. The steps

taken are as follows:

1. A Hidden Markov Model can be considered as a weighted directed acyclic graph or trellis graph. For a series of $T$ discrete observations and $n$ hidden states, the graph contains $n*T$ nodes and $n^2*T$ edges connecting $n$ nodes at $t=i$ with $n$ nodes at $t=i+1$. Edge weights correspond to the joint probability of the transition cost for each pair of nodes and the emission cost of the output symbol.

2. First create $n*T$ nodes for step t of the trellis.

3. Next create the necessary $n^2*T$ edges and calculate the appropriate edge weights as given by $A_{ij}*B_j*(O_t)$.

4. Add a start node and create edges connecting to each of the nodes at $t=0$ with weights encapsulating the initial probabilities $\pi_i*B_i*(O_1)$.

5. Create a sink node and connect it to all the nodes at $t=T$ with a zero weighting.

6. Before we can run the shortest path algorithm we must make one final change to address the impedance mismatch between graph-based cost minimisation and Markov probability maximisation. The edge weights need to be converted by converting to negative log-space as in Eq. 5.35 for a probability $p$.

$$c = -log(p) \tag{5.35}$$

7. To recover the original probability just apply the inverse (Eq. 5.36)

$$p = exp(-c) \tag{5.36}$$

Suffice it to say step 6 and 7 are not required in unit selection for concatenative synthesis since the state space is already based on cost. Figure 5.20 shows an auto-generated trace of the $k$ shortest paths, with $k=2$, applied to a graph reformulation of the Wikipedia HMM example. The order of the states are reversed, so the two output sequences are actually $S_1=(0,0,1)$ and $S_2=(0,1,1)$ respectively.

#### 5.5.5.4 *k*-Best Viterbi Decoding

The Viterbi algorithm has proved a robust and reliable solution in many problem applications. As we emphasise in this thesis and (Ó Nuanáin et al., 2017c) however, it only outputs the maximum probability path from the model. This has been observed by other researchers as being restrictive when wanting to explore alternative paths through the system (Schwarz, 2003; Brown & Golod, 2010). Rabiner and Juang also observe, in the context of dynamic time warping, that the single solution Viterbi

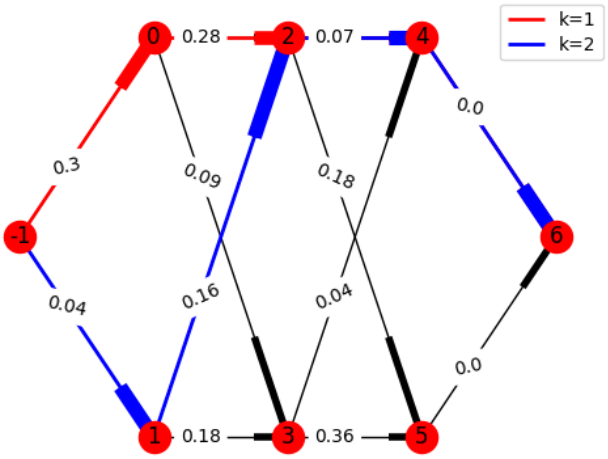



**Figure 5.20:** Shortest Paths applied to the Wikipedia Viterbi Example

is often too sensitive and it is desirable to produce a "multiplicity of reasonable candidate paths so that reliable decision processing can be performed" (Rabiner & Juang, 1993). They outline a procedure for performing *k*-Best decoding using what they term the Parallel Algorithm, which can be summarised as followed.

## 5.6 Developing a *k*-Best Concatenative Synthesiser

This chapter has hopefully presented a thorough treatise of the technical underpinnings of the key aspects of concatenative synthesis, organised by the conceptual stages of unit decomposition, analysis and unit selection, along with the presentation of two substantial contributions in the case of analysis and unit selection.

Now we turn the effort to putting all these components together to actually create some sounds. This is done in the context of PyConcat, a framework for concatenative synthesis that coalesces the previously described state of the art methods as well as novel ones like *k*-Best unit selection. Some experimental evidence is also exposed that confirms the validity of our contributions. The scope of this chapter is still general to all stylistic intentions. In Chapter 6 a system is described that tailors such methods specifically for rhythm-centric dance production as stipulated in the introduction of the thesis.

## 5.7 PyConcat - Python Framework for Concatenative Synthesis

A more detailed technical introduction to the usage of PyConcat is offered in the Appendix along with example code, but this section will give a broad enough insight in order to conduct some studies. To perform a concatenative synthesis task with PyConcat the composer first requires a corpus of interesting sounds and a target sound they wish to emulate the sonic character of. They then provide parameters and issues instructions for performing the three key stages that are required[73]

### 5.7.1 Onset Detection and Segmentation

The composer must choose the unit scale for segmenting sounds into their constituent units. The unit scales need not be the same for the target or corpus. The options we provide include

- **Framewise** - slice the samples at uniform length $N$.
- **Framewise FFT** - slice the samples at uniform length $N$ and convert to the frequency domain. Overlap-add IFFT is performed on selected units before final concatenation.
- **Onset** - slice the soundfiles according to an onset detector.
- **Beats** - slice the soundfiles according to a beat detector.
- **None** - do not perform segmentation and use the entire soundfile.

### 5.7.2 Feature Extraction

In order to effectively describe our audio content and produce meaningful similarity measures between constituent units both in the target sequence and the corpus of segmented units we need to choose a suitable set of descriptors. Selecting appropriate features is a trade-off between choosing the richest set capable of succinctly describing the signal, on the one hand, and the expense of storage and computational complexity, on the other.

We expose all the features discussed in this chapter as options for analysing sounds so the composer can combine any of the following:

- **Temporal:** loudness, attack

[73] In fact, the system is sufficiently decoupled that any of these logical stages can be performed separately for their own purpose. For example the tool can be used solely for slicing sounds, or performing batch feature analysis on a library for the purposes of MIR.

- **Spectral:** flatness, centroid
- **Timbral:** BFCCs, MFCCs, GFCCs
- **Musical:** f0, HPCPs

### 5.7.3 Unit Selection

Extracted features for the target units and corpus units and are stored in two matrices $T$ and $C$. Next the distance matrices $A = T * C$ and $E = C * C$ are calculated to avoid unnecessary computation later. Beforehand, however, it is important to perform any normalisation or standardisation and weighting of individual features as required.

A unit selection procedure is applied to return a sequence of corpus indices using:

- **Linear Search** - return an ordered list of the closest corpus units to each target unit.
- ***k*-D Tree** - return the approximate nearest neighbours for each target unit.
- **Viterbi** - return the sequence with the minimised target and concatenation costs.
- ***k*-best Viterbi** - the top $k$ sequences with the minimised target and concatenation costs using parallel list decoding.
- ***k*-shortest Paths** - return the top $k$ sequences with the minimised target and concatenation costs using $k$ shortest paths.

### 5.7.4 Transformation and Concatenation

The unit selection stage returns a vector of indices of length $N$ or a matrix of $k * N$ indices where $N$ is the number of target units and $k$ is the number of sequences from the $k$ best schemes. These indices correspond to individual sound units within our corpus. To produce the final audio representation we simply concatenate the floating point samples uniformly back to back, but there's no reason why they can't be overlapped and crossfaded at their boundaries for further smoothing as in (Schwarz, 2006).

Unlike other investigators we don't focus on complex post-selection transformations, but if there is a large discrepancy between the lengths of the target units and the corpus units or the pitch needs to be adjusted some time compression or pitch-shifting can be applied, which we facilitate through the a transformation stage utilising the Rubber Band Library[74] which is also used in the systems of Davies et al. (2013) and Smith et al. (2015).

[74] breakfastquay.com/rubberband/

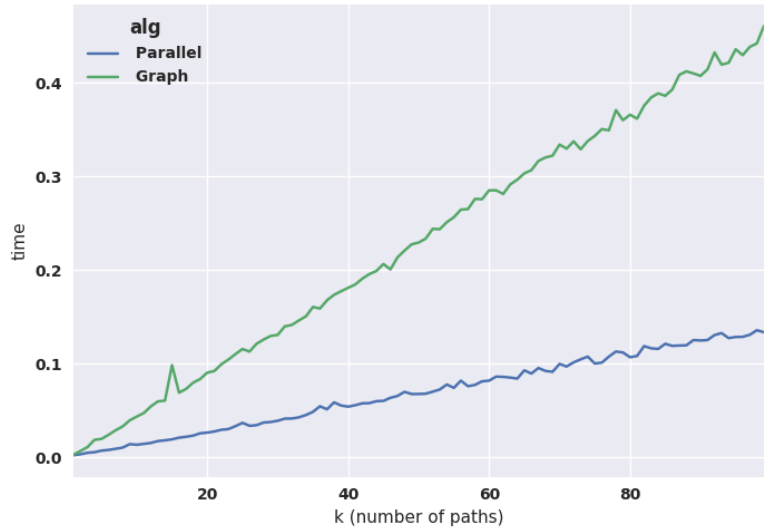



**Figure 5.21:** Running Time Comparisons for both Algorithms

## 5.8 Analysis of the *k*-Best Unit Selection

In Chapter 7 we shall conduct a stringent and rigorous evaluation of the more fully-formed concatenative synthesis system described in Chapter 6, which uses a type of linear search selection method coupled with a combination of timbral, spectral and loudness features. To conclude this chapter, however, we give a brief analysis of the two specialised *k*-best and *k* shortest selection units for comparative reference.

### 5.8.1 Algorithm Performance

To compare the running times of the two implemented algorithms we took a small recording of the 8 notes of the C major scale being performing on a piano and resynthesised it with itself. Figure 5.21 shows side by side of the running times for both algorithms based on the number of paths returned. Our implementation of the Parallel Decoder outperforms the NetworkX Graph approach considerably. This is mostly due the added layer of complexity of expanding the Hidden Markov Model to a fully connected directed acyclic graph. Others have noted however that NetworkX, on account of being implemented purely in Python, performs slower than other compiled graph libraries such as *graph-tool* or *igraph*.

It will be worth benchmarking against these implementations or reimplementing completely in C/C++. In any case, it is worth bearing in mind the real culprit in these applications: audio signal processing. The segmentation and analysis of the

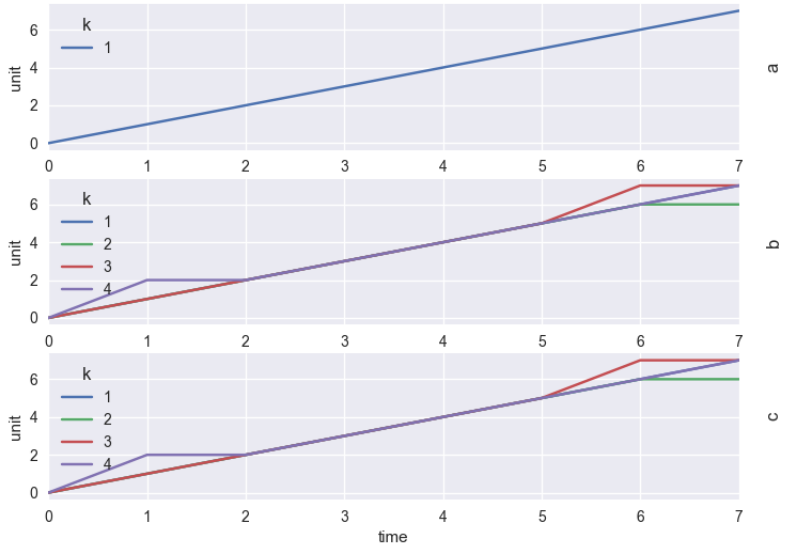



**Figure 5.22:** Equivalency of Unit Selection Algorithms

units took 0.926248 second for all runs.

### 5.8.2 Equivalence and Correctness

Figure 5.22 shows the optimal sequence generated for regular Viterbi decoding and sequences $1 \leq k \leq 5$ for the Parallel Decoder and Graph Decoder respectively. The single straight line indicates that using the chosen acoustic features and weightings, the baseline Viterbi decoder correctly reassembles its own input. The Parallel and Graph decoders both correctly return this sequence as the first optimal path also (obscured by the subsequent paths in the diagram). Looking at the other returned paths, we see very slight deviations of only one unit per path ($k$ is very small relative to the total possible paths). The possible paths, and their ordering, are identical for both algorithms.

### 5.8.3 Pitch and Timbre Preservation

To study the HMM's acoustical output, we selected energy, fundamental frequency ($f0$) and MFCC features, and chose a weighting scheme that gave preference to the $f0$ in the target cost while giving preference to the MFCCs in the concatenation cost. This allows us to focus on the specific ability of the target cost in preserving the pitch between each onset in the target sample and the selected units from the corpus, while the concatenation cost attempts to preserve a continuous and coherent evolu-

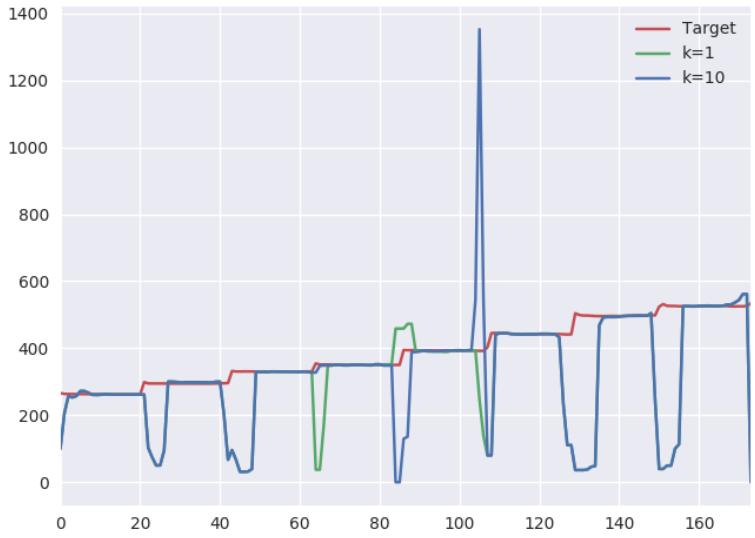



**Figure 5.23:** Running Time Comparisons for both Algorithms

tion of timbre from a variety of timbres in the corpus. We used the same piano scale sample as a target but this time using a corpus of samples from a completely different set of instruments. From a collection of orchestral samples provided freely by the London Philharmonia, 610 violin, viola, clarinet and trumpet samples were gathered with notes ranging from MIDI A3 to G7.

As we can see from Figure 5.23, and after some median filtering to remove spurious spikes, the steady states of each pitch match for generated path 1 and generated path 10 taken as examples.

MFCC plots are always a bit difficult to decipher, but while it is clear that the MFCC overall profile of the target (Figure 5.24) contrasts with the targets we can see the eight notes of the sequence reproduced and the difference in attacks that are also present in the plots of the fundamental frequencies. Again, notice lower values of $k$ produce very similar results when the size of the corpus is large, differing only by a couple of units.

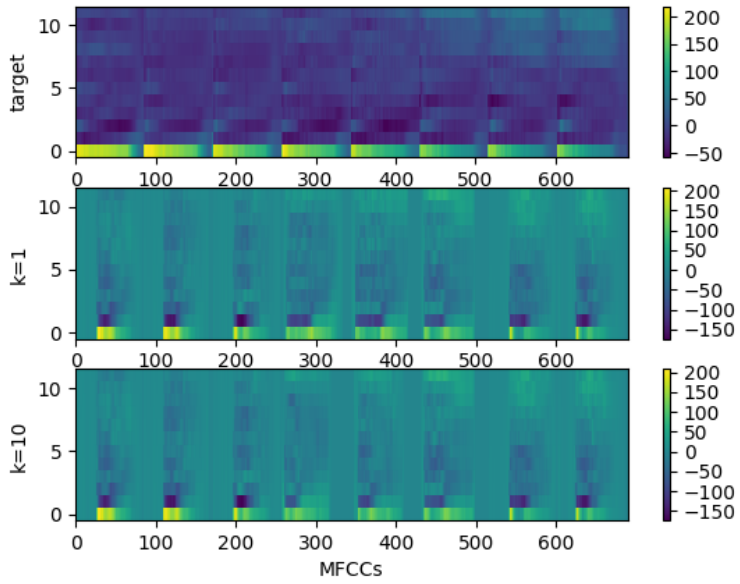



**Figure 5.24:** MFCC plots of k-Best Sequences

# Chapter 6

# Implementing User-Tailored Concatenative Synthesis Systems for Electronic Music

## 6.1 Introduction

Chapter 5 offered a deep, technical description and implementation of key methods in concatenative synthesis culminating in the proposal of some novel contributions that it is hoped improves the state of the art in a meaningful and valuable manner. This chapter, on the other hand, is all about the *practical* aspects of adapting concatenative synthesis for everyday integration into the workflows of our target user - the dance music producer. The two biggest challenges we face here can be distilled to:

1. *Interface* - devising appealing visualisation strategies and interaction metaphors for communicating gesture and feedback.
2. *Performance* - striking a balance between accuracy and feasibility in delivering usable, real-time or quasi real-time results.

The first section deals with visualisation and interaction of sounds in concatenative synthesis, and traces the emergence of its principal HCI pattern, the interactive timbre space. The following section introduces the RhythmCAT[75] software that finally amalgamates elements of our concatenative synthesis methods with a rich interactive interface for the user.

[75] Not to be confused with Rhythm Cat, an educational game for iPad that trains a user to read music with the help of a friendly feline mascot (https://melodycats.com/rhythm-cat/). We acknowledge the name needs a bit of work!

## 6.2 Visualising and Interacting with Sound

Up until now the conversation with our synthesis engine has been abstract and textual. Content is fed to the system, some parameters are set and the patient user, after a time, receives the rendered output. For many users[76] and their musical applications, this is perfectly acceptable way of creating sounds. For many other users, and we mean here dance music producers, who are accustomed to a wide range of user friendly and appealing music interfaces, it is clearly not an ideal *metaphor* for visualising their sounds or interacting with them.

### 6.2.1 Sound and Music Computing Interface Metaphors

Metaphor, in the context of human computer interaction and the craft of designing interfaces can reduce the complexity of a system for a human user by abstracting some of the difficult concepts or as (Rogers et al., 2011) qualifies, "[providing] familiar entities that enable people to readily understand the underlying conceptual model and know what to do at an interface".

Most people are already familiar with the *desktop* metaphor, that superseded the textual interface as the primary modality for interfacing with an operating system. The desktop metaphor exploits several real world objects, such as an office desk, folder and bin[77], in order to establish a relatable channel communication between man and system (Helander, 2014). In sound and music domains there are also several metaphors that have sprung up to handle its manipulation via software and hardware. Duignan et al. (2010); Fels et al. (2002) have published frequent works in a bid to classify this landscape but we highlight some of the key ones here.

**The Waveform Metaphor** The waveform metaphor is "a conventional metaphor for computer tools that deal with audio" (Duignan et al., 2004) and the dominant interface for sound editors and multi-track studio DAW software like Pro Tools and Cubase.

It offers an immediate view of the temporal evolution of the amplitude over time and allows for intuitive manipulation of the same dimensions via splicing, copying and pasting and fades that also draws its inspiration from the medium of tape.

**The Loop Grid Metaphor** The waveform metaphor provides an appropriate interface and interaction for recording and editing performances or events with a fixed start, end and progression through time, but as (Duignan et al., 2005) comment, it and derived timeline taxonomic containers and structures are "highly biased towards eager linearisation".

[76] And computer music composers of a certain vintage.
[77] Or 'thrash' for our North American English speakers.

Dance music practitioners and DJs make music differently. They usually operate "in the box" (studio jargon for mixing engineers who work with software rather than physical consoles and outboard processors) and increasingly perform live concerts solely with software. For the dance music producer the loop is king: they require an intrinsically non-linear method of on the fly branching and combining of looped material in situations that do not have the traditional fixed boundaries of a start and end.

In the early days, Berlin techno veterans Apparat and Modeselektor[78] developed their own 'looper' tools in Max to cope with the specific requirements of delivering engaging live dance music. Nowadays dance music producers are catered for mostly by Ableton's Live[79][80] software and more recently Bitwig[81][82]. The revolutionary archetype brought upon by Live (and subsequently Bitwig re-created) was the 'Session view', an intuitive, visual matrix system allowing seamless and endless combinations of that the fundamental unit - the loop - all in real time.

**The Modular Metaphor** The modular metaphor derives its interaction paradigm by visually connecting virtual cables to logical units or objects akin to connecting electronic equipment in the physical real world. The ubiquity of visual dataflow programming languages such as Pd, Max and Reaktor shows the utility of this metaphor not just in simulating audio connectivity but also as an alternative mental paradigm for general purpose computer programming.

One of the most compelling variations of the modular metaphor is the Reactable (Jordà et al., 2005), which provides a tangible table-top interface for virtual modular synthesis. Physical blocks with visual patterns engraved on them are placed on a table and represent different functions (e.g. oscillators or filters). A computer vision component recognises the functions of those blocks and visualises and synthesises the flow of audio and control data between them virtually.

### 6.2.2 Timbre Spaces and Dimensionality Reduction Techniques

The problem with the dominant metaphors for sound and music visualisation and interaction is that they lack sufficient faculty for representing the dense, heterogenous entity that is timbre. In MIR research this is also a huge challenge, and there are plenty of examples of work devoted to studying and devising visualisations of all types of musical analysis as well as timbre (Hoashi et al., 2009; Cooper et al., 2006a). One way of creating a visual and mental map of sounds organised by their timbre

[78] http://daily.redbullmusicacademy.com/2016/08/modeselektor-on-the-life-and-death-of-50weapons

[79] http://www.ableton.com

[80] Early proof of concepts of Ableton were drafted in Max/MSP by its founder, Berlin-based techno musician and computer scientist Robert Henke aka Monolake

[81] http://www.bitwig.com

[82] Founded by a group of ex-Ableton employees.

is to produce a scatter plot representing their point in space according to some parameters or dimensions related to that timbre, known as a *timbre space*. In Chapter 4, this notion of representing timbral aspects was revealed to be a central visual tool in interacting with concateantive synthesis in a more intimate and meaningful manner.

In CataRT (Schwarz et al., 2006), timbreID (Brent, 2010) and earGram (Bernardes et al., 2013) for example, the sounds are positioned in a timbre space by assigning a specific feature (or dimension of a vector feature) to each axis, with a third dimension assigned to RGB colour scale [83]. This is particularly useful in the case of scalar features with a very coherent perceptual correspondence. In Figure 6.1 we see a plot of every combination of temporal and spectral features used in the classification in Chapter 5 . Assigning loudness and spectral centroid (or "brightness" to give it some perceptual labelling), for instance, presents a meaningful organisation of sounds in 2D timbre space (top-right in the figure) and are easily comprehensible by the lay user. But choosing only two features from a rich set of descriptors in multidimensional timbre space can discard potentially important statistics that can give good separation between points. Furthermore, while loudness and "brightness" are lucid qualities for many users, expecting them to compare and decipher higher order cepstral coefficients is perhaps not so transparent.

To address some of the shortcomings of visualisation based on raw singular feature values, it is often useful to apply some form of dimensionality reduction technique that can provide an 'automatic' projection of the higher-order feature space into the timbre space suitable for visualisation of the data, while hopefully retaining most of the important trends latent within the data. Projections of higher order timbre spaces using dimensional scaling have been talked about in the literature for quite some time, one of the key innovators in the usage of timbre spaces for creative or generative purposes has been David Wessel. By taking measurements of subjective responses to timbral stimuli, dimensionality reduction schemes are used to present a map of timbral references in space that can be then used to provide a spatial reference for generation and control of additive synthesis (Wessel, 1979, 1976)

Many dimensional reduction techniques span the literature as it is an important tool in general statistical analysis and machine learning. Specifically in the context of organising and exploring multimedia collections based on similarity, Christian Frisson's doctoral thesis (2015) offers a broad and up-to-date introduction to the main techniques and is a worthwhile reference. Here we summarise the principal methods that we have experimented with in our work.

#### 6.2.2.1 Principal Component Analysis

Principal Component Analysis (PCA) is a statistical procedure applied to data that can discover trends in order to preserve those with the most variance, effectively

[83] Though Bernardes does extend the option for dimensionality reduction using Star Coordinates (Cooprider & Burton, 2007), a technique based on summing the feature vectors.

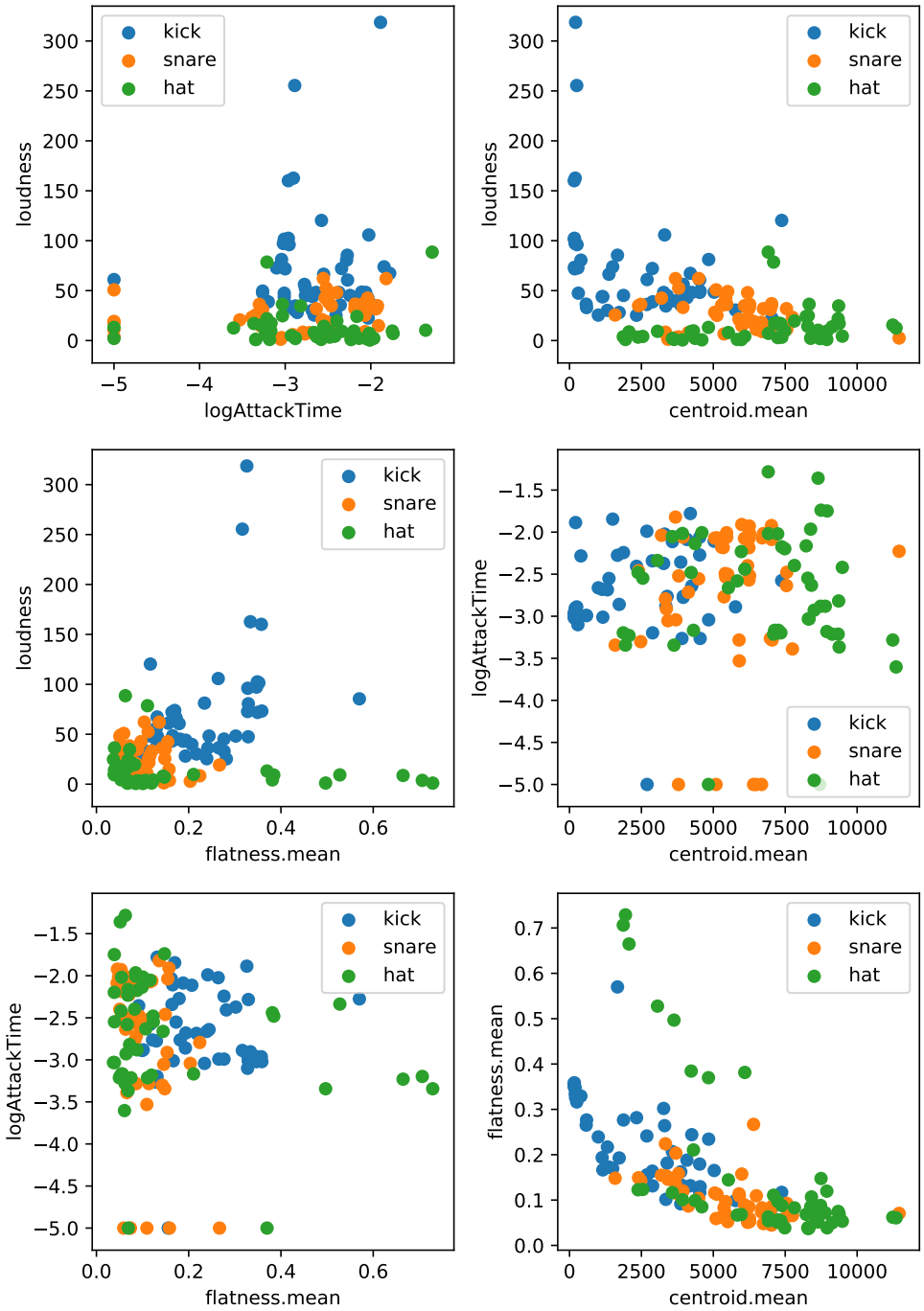



**Figure 6.1:** Visualisation of all temporal and spectral (taking the mean only) feature combinations for kick, snare and hat drum sounds

allowing us to compress data from high-dimensional space into lower-dimensional space without considerable "loss" of salience (Hackeling, 2014). Practical applications of PCA includes reducing redundant (features with low variance) data for more efficient machine learning tasks, and for producing coordinates that can make comparative visualisation of data possible in 2 or 3 dimensions.

It is the second application that interests us here, and in fact PCA has seen considerable application already in visualisation for musical applications (Cooper et al., 2006b). Most of these involve organising large musical collections. Within the GiantSteps project, our colleagues have compared PCA with MDS (see below)

PCA involves a few stages of matrix algebra and is implemented already in many libraries and frameworks so we won't concern ourselves with its derivation here. However the key stages are summarised in an excellent tutorial by Smith (2002). Basically, the method uses eigenvalues and eigenvectors to extract the axes of maximum variance from the covariance matrix of the original data, and the highest eigenvectors are the principal components. As many principal components can be retained as required, with the assumption being that the loss will be minimal even as successively more components are discarded.

#### 6.2.2.2 Multidimensional Scaling

Multi-Dimensional Scaling (MDS) actually refers to a taxonomy of methods for dimensional reductions including 'classical' or Principal Coordinate Analysis (PCoA), metric and non-metric MDS. MDS uses a distance or dissimilarity matrix as input and attempts to reduce the dimensions while preserving pairwise distance between entities, compared to PCA which tries to preserve maximum variance. Frisson (2015) puts it more succintly: "MDS uses a distance matrix whereas PCA uses a correlation matrix.".

Regarding musical applications, apart from the early experiments of Wessel, (Wessel, 1976, 1979) MDS has been used somewhat less than PCA. Within the GiantSteps project though, our colleagues have been studying the effects of symbolic rhythmic features on the organisation of symbolic rhythmic patterns in dance music when dimensionality reduction techniques such as PCA, MDS and t-SNE (to follow) are applied (Gómez-Marín et al., 2017, 2016). This expanded on previous perceptual dimensional scaling studies conducted by Gabrielsson (1973a,b) in the 1970s, who performed similarity ratings on many parameters of rhythmic similarity of drum machine patterns and tried to find the most salient parameters that influenced the positioning of the patterns in a reduced MDS space.

#### 6.2.2.3 t-Distributed Stochastic Neighbour Embedding

t-Distributed Stochastic Neighbour Embedding (t-SNE) is a relatively recent prize-winning[84] approach to data visualisation that is tailored specifically for effective reduction of high-dimensional datasets (Van Der Maaten & Hinton, 2008). Based on a previous method known as Stochastic Neighbour Embedding (SNE), one of the improvements it offers is reductional strategies for preventing crowding or over-layered clusters in the centre of the map inherent in SNE methods. Essentially this is achieved by using a Student-t distribution rather than Gaussian for computing dissimilarity between points in low-dimensional space (Van Der Maaten & Hinton, 2008).

The t-SNE algorithm is gaining considerable favour in music information tasks as a way of navigating large collections of complex data representations (Hamel & Eck, 2010; Grill, 2012; Frisson et al., 2014b,a; Flexer, 2015; Martín, 2017; Font & Bandiera, 2017). In a comparative evaluation of MDS, PCA and t-SNE based on three criteria Dupont et al. (2013) revealed that t-SNE outranks PCA which in turn outranks MDS in three tasks related to musical loops and sound effects.

Returning to our own tasks of visualising feature vectors of drum sounds, Figure 6.2 shows these three dimension reduction techniques applied to 50 instances each of kick, snare and hi-hat sounds based on the full gamut of features extrapolated with the GFCC+Spectral+Temporal configuration. One thing that's immediately apparent across all methods is their increased use of the 2-dimensional space in comparison to the individual assignment of features to axes as we saw in Figure 6.1, as well as exhibiting less crowding in overlapping clusters.

Most methods are quite capable of sorting the distinctive low frequency membrane kick sounds from the snares and hi-hats. There is, however, less separation of the more broadband cymbal and snare sound classes in both the MDS and PCA reduction methods. We see more overlapped clustering in the PCA method in this instance, but in other reduction experiments this has often been reversed. In an ideal world a concatenative system would have some trained mechanism for performing all three procedures then choosing the one with the best separation.

#### 6.2.2.4 Interacting with Sounds and Creating a Metaphor

Undoubtably, the timbre space paradigm represent a much richer modality for organising sound spatially and a provides a vital window into understanding the procedures governing concatenative synthesis. We have seen how exactly rendering these spaces is not trivial and it poses a unique set of challenges and tradeoffs to get a usable and meaningful landscape for exploration.

Equally challenging, once a timbre space has organised a visual map of sound, is how to *interact* with those sounds. How can the user effectively navigate this sonic terrain and avail of suitable gesture and metaphor to arrange the sounds in the desired

[84] https://lvdmaaten.github.io/tsne/

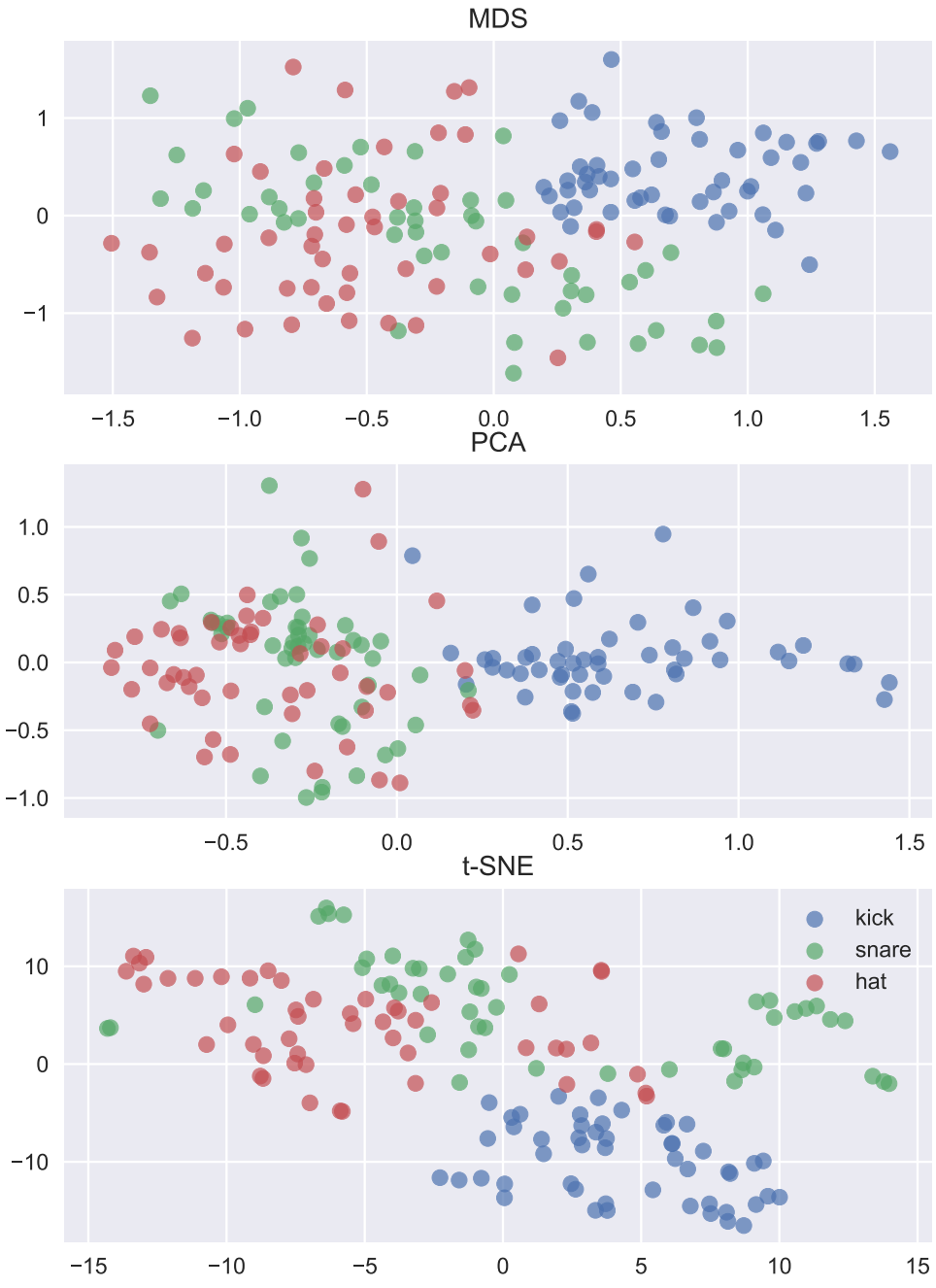



**Figure 6.2:** MDS, PCA and t-SNE dimension reduction processes applied to kick, snare and hi-hat sounds

manner? How do existing concatenative synthesis deal with these challenges? To complicate matters further, our system is inherently rhythm driven. How do those sound exploratory devices extend to our unique problem domain?

In CataRT the primary interaction metaphor is simple and intuitive: the target is the real-time position of the mouse cursor and a resizable radar around the target filters the range of sounds within its proximity. How these sounds are triggered depends on the mode selected. The triggering modes include:

- *Bow* - trigger closest unit each time you move the mouse
- *Fence* - like a stick dragging along a fence, triggers when a new unit becomes closer to the target
- *Beat* - triggers according to metronome
- *Chain* - triggers after previous unit finishes
- *Quant* - non-functional quantised mode
- *Seq* - trigger via external sequencer
- *Cont* - play back units in order

In earGram the SpaceMap mode also triggers units based on proximity to the mouse pointer target, but defines three different triggering modes distinct from CataRT:

- *continuousPointer* - continuously play units at a specified rate
- *pointerClick* - same as continuousPointer but "played in response to a controller command" (the meaning of this is not exactly clear from the description provided)
- *colorPicker* - selects units based on colours.
- *liveInput* - feature analysis of live input sets pointer position.

AudioGarden appears to adapt the mouse radar approach of CataRT also, but the positioning of multiple radars in the timbre space means multiple grains can be triggered for a single target unit in time which is a very simple and useful method of mixing corpus units vertically in addition to concatenating them horizontally.

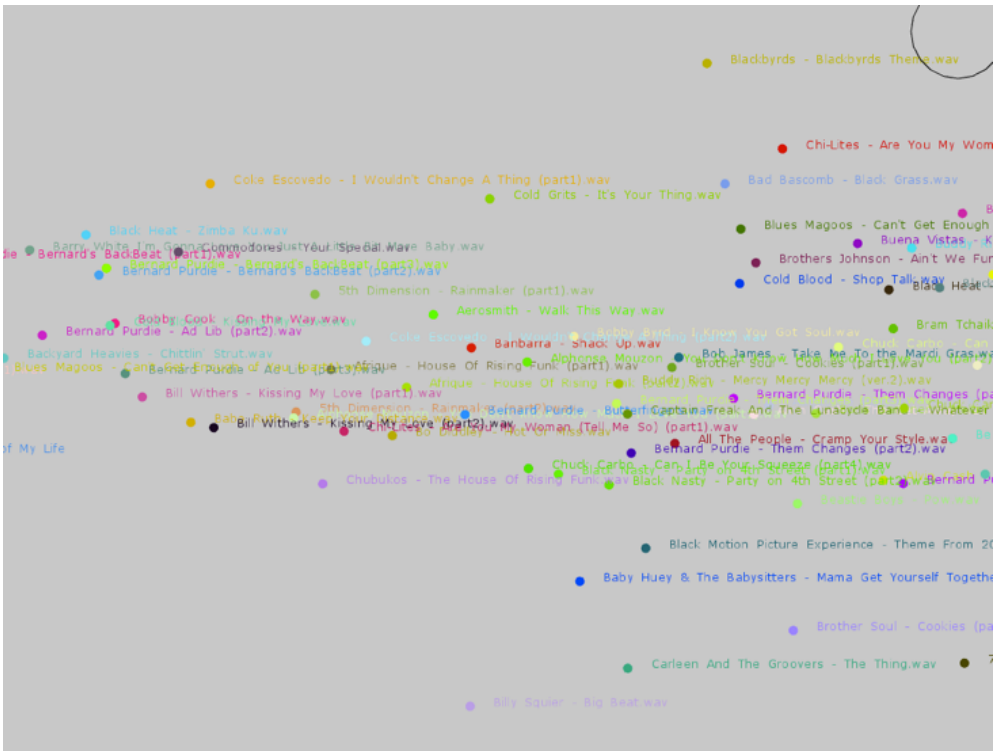



**Figure 6.3:** RhythmCAT First Prototype. Entire waveforms are represented in space using PCA reduction of the MFCC. A small mouse radar scans the space and onsets are played at random from the files in proximity.

### 6.2.3 A Prototypical Rhythmic Concatenative Synthesiser Interface

#### 6.2.3.1 First Iteration

Our first attempt at a prototype interface that was capable of performing rudimentary rhythm-driven concatenative synthesis was naturally extremely crude and experimental (Figure 6.4) but a valuable learning experience and necessary stepping stone.

Developed in openFrameworks[85], the application used MFCC features computed on the entire audio file, then positioned the sound in space (along with its filename string for some reason which now evades me!) according to PCA reduction of the features. To explore the sounds we adapted the radar style navigation metaphor introduced in CataRT, but this immediately raised the question of how to trigger the sounds. This early incarnation simply selected an onset chosen at random from within radar proximity to the mouse target and triggered at each beat according to a 1/16 grid subdivision of the tempo.

[85] http://openframeworks.cc/


**Figure 6.4:** RhythmCAT Second Prototype, implemented as a VST plugin

#### 6.2.3.2 Second Iteration

The next iteration was largely concerned with implementation. We reasoned that the typical dance music producer would not use software like this unless it was integrated into their existing workflow of tools, meaning we needed to consider how interface it with a DAW like Ableton. So, we spent some time developing a streaming real-time plugin using Virtual Studio Technology (VST) that could synchronise with the tempo information received from a VST-host enabled DAW, and also allow the user to select different beat subdivisions.

While this version was more streamlined and featured a slicker interface built with the aid of JUCE, fundamentally it performed the same functionality. The points in space referred to complete sound files and a mouse radar filtered the points by proximity. Targeting could be controlled by external systems via the OSC or automatically and algorithmically using Boids (Reynolds, 1987), a flocking algorithm that has been used in other systems for controlling spatialisation and granulation parameters in electroacoustic works.(Kim-Boyle, 2006; Wilson, 2008; Barreiro, 2010).

But these are arbitrary and deferential extensions that still just scrub the surface features of the the visual space. Using the tool as a rhythm generation tool reveals the

deficiency of the mouse radar exploratory method. A typical drum pattern is made up of complementary sounds in different timbres and hence occupy different zones of the timbre space at different points in time. This is conceivably better served by the multi circle model of (Frisson et al., 2010)'s AudioGarden, which could be used to place three circles in the timbre space corresponding to the kick, hat and snare, as an example.

Most critically, these prototypes did not yet integrate the possibility of an audio target could intelligently define where the system should search for units to select. We will will now discover how these two outstanding concerns are resolved by adapting a graph model, inspired by the modular metaphor, that shows the spatial connection and temporal evolution between consecutive units selected for concatenation, in the breadth and span of a timbre space, given a target sound specification.

## 6.3 RhythmCAT

Building on all the topics we have been discussing in this dissertation, RhythmCAT was conceived as "a user-friendly system for generating rhythmic loops that model the timbre and rhythm of an initial target loop" (Ó Nuanáin et al., 2017b). This system was first introduced at New Interfaces for Musical Expression (NIME) in 2016 (Ó Nuanáin et al., 2016b), subsequent papers dealt with its evaluation (Ó Nuanáin et al., 2016c,a) culminating in a summary journal article for the Computer Music Journal in 2017 (Ó Nuanáin et al., 2017b).

### 6.3.1 Developing the System

In this section, we will describe our implementation of the RhythmCAT system, beginning with an explanation of the musical analysis stages of onset detection, segmentation and feature extraction. This is followed by an examination of the interactive user interface and the pattern generation process. Figure 6.5 gives a diagrammatic overview of these important stages, which can be briefly summarised as:

1. Sound Input
2. Onset Detection & Segmentation
3. Audio Feature Extraction
4. Storage & Data Representation
5. Pattern Synthesis
6. Real-time Audio Output

The system is developed in C++ using the JUCE framework [86], the Essentia

[86] https://juce.com/

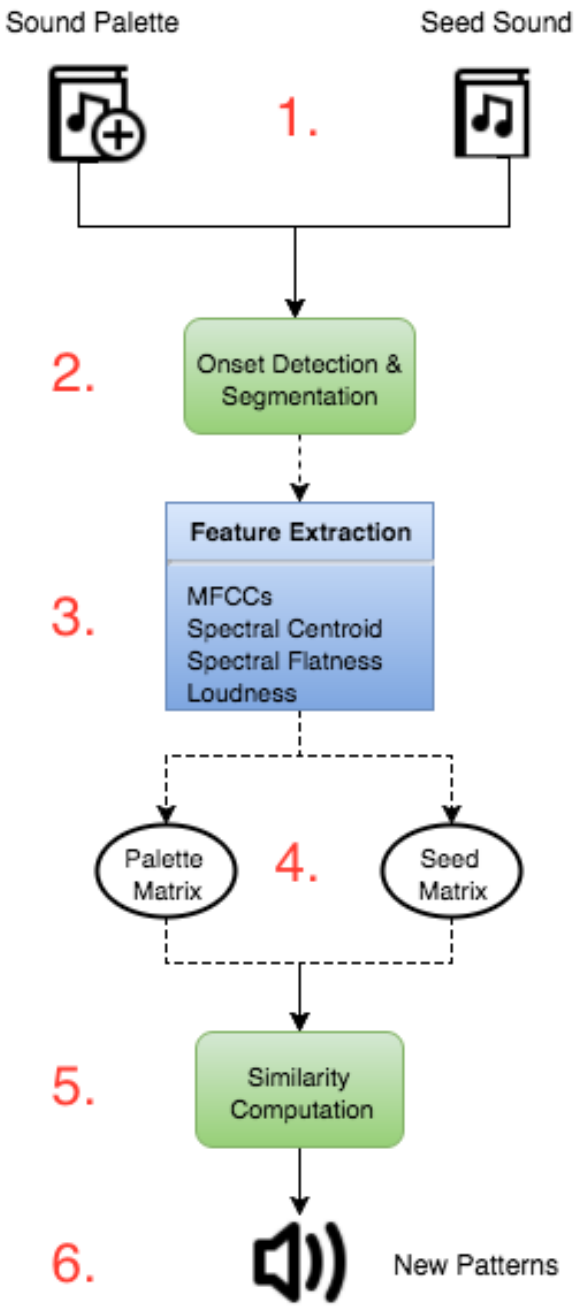



**Figure 6.5:** Block Diagram of Functionality in RhythmCAT System

musical analysis library (Bogdanov et al., 2013) and the OpenCV computer vision library (Bradski, 2000) (for matrix operations an PCA).

#### 6.3.1.1 Sound Input

The first stage in building a concatenative music system generally involves gathering a database of sounds to select from during the synthesis procedure. This database can be manually assembled but in many musical cases the starting point is some user-provided audio that may range in length from individual notes to phrases to complete audio tracks.

The two inputs to the system are the Sound Palette and the Seed Sound. The sound palette refers to the pool of sound files we want to use as the sample library for generating our new sounds. The seed sound refers to the short loop that we wish to use as the similarity target for generating those sounds. The final output sound is a short (one to two bars) loop of concatenated audio that is rendered in real time to the audio host.

#### 6.3.1.2 Onset Detection and Segmentation

In cases where the sounds destined for the sound palette exceed note or unit length, the audio needs to be split into its constituent units using onset detection and segmentation.

We covered onset detection methods at length in Chapter 5, and we showed how how the most accurate methods come at a cost of increased computational complexity, especially when involving the use of state of the art deep learning techniques. What's more, the CNN algorithm is only available in the Python Madmom library, which excludes it from being a possible method in our compiled and monolithic plugin at the present time. A port of Superflux is available in Essentia, but for best results it requires considerable tweaking of the parameters that depends heavily on the program material to be analysed, meaning it was quite hard to find a generalised configuration to cover a wide range of possible input sounds to the instrument.

To this end we rely on the standard 'OnsetRate' method that at least returned acceptable results when concentrated on the percussive datasets used in the evaluation (Table 5.2). As a reminder this invoked two methods based on analysing signal spectra from frame to frame (at a rate of around 11 ms). The first method involves estimating the high-frequency content in each frame (Masri & Bateman, 1996) while the second method involves estimating the differences of phase and magnitude between each frame (Bello & Daudet, 2005).

The onset detection process produces a list of onset times for each audio file, which we use to segment into new audio files corresponding to unit sounds for our concatenative database.

#### 6.3.1.3 Audio Feature Extraction

As stressed, successful feature extraction is a tradeoff between choosing the richest set of features capable of describing the signal succinctly at the expense of storage and computational complexity. This is more important than ever in the design of a real-time instrument such as here. Whereas the PyConcat environment offered powerful selection and combination of a wide set of descriptors, RhythmCAT has a smaller subset that is more manageable complexity wise. These features are as follows, along with their intended perceptual label.

- **Power** - computed using Stevens' Law to estimate "Loudness".
- **Spectral Centroid** - for estimating "Brightness".
- **Spectral Flatness** - for estimating "Harmonicity".
- ***FCC [2-13]** - for abstracting "Timbre". At the moment we use regular MFCC as our implementation of BFCC has not been fully optimised and is slightly slower.

#### 6.3.1.4 Pattern Synthesis and User Interaction

Further on we will describe a bit more on how the seed or target audio signal is actually received from the VST host, but in terms of analysis on that seed signal, the process is the same as before: onset detection and segmentation followed by feature extraction.

The resulting feature vectors are stored in two matrices: the palette matrix and the target matrix. The palette matrix stores the feature vectors of each unit of sound extracted from the sound palette and similarly the target matrix stores feature vectors of units of sound extracted from the seed loop.

#### 6.3.1.5 Pattern Synthesis

This section details the visible, aural and interactive elements of the system as they pertain to the user. Figure 6.6 gives a glimpse of the user interface in a typical pattern generation scenario.

**Workflow** The layout of the interface was the result of a number of iterations of testing with users who, while praising the novelty and sonic value of the instrument, sometimes expressed difficulty understanding the operation of the system. One of the main challenges faced was how best to present to the user the general workflow in a simple and concise manner. It was decided to represent the flow of the various operations of the software emphatically by using a simple set of icons and arrows, as is visible in Figure 6.6.

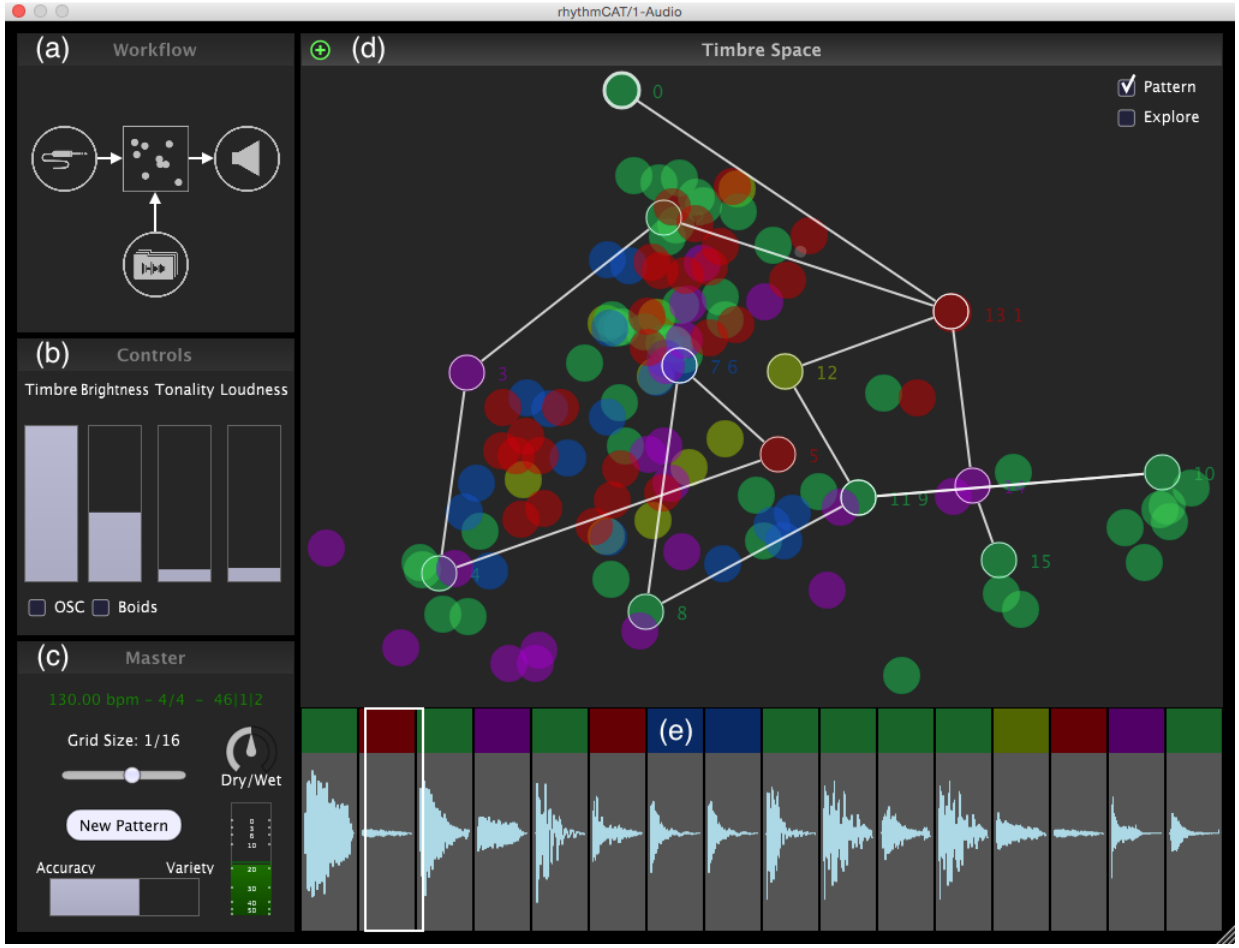



**Figure 6.6:** Main User Interface Showing Panels A-E with Onset Graph

The icons indicate the four main logical operations that the user is likely to do, and opens up the related dialogs, namely:

- The Palette Dialog - as indicated by the folder icon
- The Seed Dialog - as indicated by the jack cable icon
- The Sonic Dialog - as indicated by the square feature space icon
- The Output Dialog - as indicated by the speaker icon

**Sound Palette** The user loads a selection of audio files, or folders containing audio files, which are analysed to create the sound palette as has previously been discussed. Next, dimensionality reduction is performed on each feature vector of the units in the sound palette using Principal Component Analysis (PCA). Two PCA components are retained and scaled to the visible area of the interface to serve as coordinates for placing a circular representation of the sound in two-dimensional space. These visual representations, along with their associated audio content, we term sound objects. They are clearly visible in main timbre space window in the figure.

**Seed Input** Seed audio is captured and analysed by recording directly from the input audio of the track on which the instrument resides in the audio host. Using the real-time tempo and bar/beat information provided by the host, the recorder will wait until the next bar starts to begin capture, and will only capture complete bars of audio. This audio is analysed as before but with one exception. Since the goal of the instrument is to integrate with an existing session and generate looped material, we make the assumption that the incoming audio is quantised and matches the tempo of the session. Thus, onset detection is not performed on the seed input; rather, segmentation takes place at the points in time determined by the grid size (lower left of the screen).

An important aspect to note: since the instrument is fundamentally real-time in its operation, we need to be careful about performing potentially time consuming operations such as feature extraction when the audio system is running. Thus, we perform the audio recording stage and feature extraction process on separate threads so the main audio playback thread is uninterrupted. This is separate to yet another thread that handles elements of the user interface.

**Sonic Parameters** Clicking on the square sonic icon in the centre of the workflow component opens up the set of sliders shown in (Figure 6.6 -B) that allows us to adjust the weights of the features in the system. Adjusting these weightings has effects in terms of the pattern generation process but also in the visualisation. Presenting their technical names (centroid, flatness and MFCCs) would be confusing for the general

user, thus we have relabelled them with what we consider their most descriptive subjective labelling. With the pattern generation process, these weights directly affect the features when performing similarity computation and unit selection, as we will see in the next section. Depending on the source and target material, different combinations of feature weightings produce noticeably different results. Informally we have experienced good results using MFCCs alone for example, as well as combinations of the flatness and centroid. In terms of visualisation, when the weights are changed, dimensionality reduction is re-initiated and hence positioning of the sound objects in the timbre space changes. Manipulating these parameters can help disperse and rearrange the sound objects for clearer interaction and exploration by the user in addition to affecting the pattern generation process.

Once the palette and seed matrices have been populated, a similarity matrix between the palette and seed matrix is created. Using the feature weightings from the parameter sliders, a sorted matrix of weighted Euclidean distances between each onset in the target matrix and each unit sound in the palette matrix is computed as per Eq. 5.21 in Chapter 5

**Unit Selection** In Chapter 5 we summarised the various unit selections ranging from the basic linear search procedure to Viterbi style decoding. We then proposed two novel unit selection methods that expand Viterbi decoding further to produce a $k$-Best list of potential sequences for exploring variations.

While these methods provide increasingly richer and more interconnected syntheses of a target sound, the complexity increases and the performance suffers, as we saw in the analysis of the two $k$-Best methods. Evidently this poses problems for real-time scenarios, as Coleman has noticed:

> "*None of the systems cited in this section use advance planning by minimizing transition costs (in contrast to some of the systems [...], i.e. Caterpillar, Musaicing, or Audio Analogies). This is likely due to its high computational cost. Instead, more immediate selection methods are used.*"
>
> (Coleman, 2015)

Which reiterates an earlier remark by Schwarz with regards to CataRT:

> "*Because of the real-time orientation of CataRT, we cannot use the globally optimal path-search style unit selection based on a Viterbi algorithm as in Caterpillar, neither do we consider concatenation quality, for the moment. Instead, the selection is based on finding the units closest to the current position x in the descriptor space, in a geometric sense...*"
>
> (Schwarz, 2006)

So this likely rules out the Viterbi decoding methods we have described, even though we should emphasise that our tool is not strictly real-time - in the sense that it computes complete loops based on the analysis of one bar in a given tempo in order to capture a temporal context - thus we would qualify it as *quasi* real-time. Regardless, for performance, in RhythmCAT we only consider the target cost when concatenating output sequences, thus making its unit selection scheme a variation on linear search. While we do sacrifice the continuity factor that is inherent in Viterbi decoding, when considered, it might not hold the same sway for rhythm purposes as it would for synthesising voice or solo instrument performance. Given that a kick and snare have, or should have, very different spectral and timbre profiles, efforts to constrain the slope of features between them should really only concentrate on ensuring some consistency of energy[87].

The unit selection algorithm is quite straightforward, and to allow for flexible exploration of variations instead of returning the nearest neighbour sequence we allow any mixing and matching of units from the corpus sorted by similarity to the targets. For each unit $i$ in the segmented target sequence (e.g. 16-step) and each corpus unit $j$ (typically many more), the target unit cost $C_i, j$ is calculated by the weighted Euclidean distance of each feature $k$ as per usual.

These unit costs are stored in similarity matrix $M$. Next we create a matrix $M'$ of the indices of the ascendingly sorted elements of $M$. Finally, a concatenated sequence can be generated by returning a vector of indices $I$ from this sorted matrix and playing back the associated sound file. To retrieve the closest sequence $V_0$ one would only need to return the first row.

Returning sequence vectors as rows of a (sorted) matrix limits the number of possible sequences to the matrix size. This can be extended if we define a similarity threshold $T$ and return a random index between 0 and $j - T$ for each step $i$ in the new sequence.

**Pattern Generation and Interaction** In our description of the early prototypes we highlighted the challenge of coming up with an effective visualisation and interaction paradigm that could show the map of sounds in timbre space as well as the temporal evolution and connection of a rhythmic-centric concatenated sequence that makes its way through that timbre space in accordance to some target. The interaction metaphor we propose to tackle this is to represent a rhythm as a path through that graph or network, with edges connecting each sound in space as appropriate. This obviously borrows heavily from the Modular metaphor explained at the chart of the chapter and (Roma & Herrera, 2010) uses a similar graph grammar style metaphor for navigating

[87] In the absence of any magical timbre feature analysis techniques that we know of that can be used to isolate and ensure unit selection comes from similar timbre of just the room profile, recording chain or album and not the instrument timbre, or in other words, only retaining those features that contribute to the 'album effect'(Mandel & Ellis, 2005a; Kim et al., 2006b)

the Freesound[88] library. More recently, Font & Bandiera (2017) proposes a similar tool for the same task that uses the t-SNE reduction method to arrange samples.

Facilitating the efficient UI visualisation and interaction of sequences of units in space is enabled with the use of a linked list data structure to contain the necessary information of each unit. Linked lists, as explained in any introductory text on data structures are an efficient way of ordering data in a linear fashion and is an apt container for chaining the series of nodes in our graph model.

To create a pattern when the user hits the "New Pattern" button (Figure 6.6 - C), a new linked list of objects we term sound connections is formed, representing a traversal through connected sound objects in the timbre space. The length of the linked list is determined by the grid size specified by the user, thus if the user specifies a grid size of 1/16 for example, a one bar sequence of 16th notes will be generated. The exact procedure whereby we generate a list is detailed in Algorithm 1. The variance parameter affects the threshold of similarity by which onsets are chosen. With 0 variance, the most similar sequence is always returned. This variance parameter is adjustable from the Accuracy/Variety slider in the lower left corner of the instrument (Figure 6.6 - C).

**Algorithm 1** Get Onset List for Concatenative Sequence

```
for n in GridSize do
    R = Random number 0 < Variance
    I = Index from Row R of Similarity Matrix
    S = New SoundConnection
    S->SoundUnit = SoundUnit(I)
    Add S to Linked List
return LinkedList
```

In the main timbre space interface (Figure 6.6 - D), a visual graph is generated in the timbre space by traversing the linked list and drawing line edges connecting each sound object pointed to by the sound connection in the linked list. In this case a loop of 16 onsets has been generated, with the onset numbers indicated beside the associated sound object for each onset in the sequence. An animated trace shows the units triggered in time with each beat of the tempo source from the VST host. Of course the most compelling aspect of this is that the user is free to manipulate these sound connections to mutate these patterns by touching or clicking on the sound connection and dragging to another sound object. Multiple sound connections assigned to an individual sound object can be group selected by double tapping then dragging, useful for changing all instances of a hi-hat or a snare for example

On the audio side, every time there is a new beat, the linked list is traversed and if a sound connection's onset number matches the current beat the corresponding

[88] https://freesound.org/

sound unit is played back. One addition that occurred after some user experiments (see Chapter 7) with the prototype is the linear waveform representation of the newly generated sequence Figure 6.6 - E). Users felt the combination of the 2D interface with the traditional waveform representation made the sequences easier to parse visually as well as being able to manipulate the internal arrangement of sequence itself once generated.

# Chapter 7

# Evaluation

## 7.1 Introduction

One of the primary difficulties faced with designing instruments for creative and compositional tasks remains the elaboration of an appropriate evaluation methodology. Indeed, this is a trending challenge facing many researchers (Barbosa et al., 2015), and numerous papers address this directly with various proposals for methodological frameworks, some drawing from the related field of HCI (Hsu & Sosnick, 2009; Kiefer et al., 2008; Hiraga et al., 2004). More generally the evaluation of computer composition systems has also been the subject of much discussion in the literature. As we alluded to in Chapter 3 a frequent benchmark for evaluating algorithmic music systems is a type of Turing test where the success criterion is determined by the inability of a human listener to discern between human and computer-generated music. As (Abdulla, 2002) notes however, these kind of tests can be problematic for two reasons. Firstly, it makes the assumption that the music generated by the algorithm is intended to sound like music produced by humans, rather than something to be treated differently. Secondly it ignores other facets of the system that imperatively need evaluation, such as the interface and the experience. Pachet & Roy (2015) also finds issue with simplistic Turing test approaches to music evaluation. He repeats, for instance, the view that unlike the traditional Turing test which evaluated the ability to synthesis believable natural language, no such "common-sense" knowledge exists for aspects of music.

As we were researching many of the existing works in Table 4.1 we were struck by the absence of discussion regarding evaluation in most of the accompanying articles. Some of the articles provided descriptions of use cases (Cardle et al., 2003) or at least provided links to audio examples (Sturm, 2004). Notably, many of the articles (Simon et al., 2005), (Hackbarth, 2011) consistently made references to the role of "user", but only one of those actually conducted a user experiment (Aucouturier & Pachet, 2005). By no means is this intended to criticise the excellent work presented by these authors. Rather it is intended to highlight that although evaluation is not al-

ways an essential part of such experiments - especially in "one-off" designs for single users such as the author as composer - it is an underexplored aspect that could benefit from some contribution.

We can identify two key characteristics of our research that can inform what kind of evaluation can be carried out. Firstly it's a retrieval system, and can be analysed to determine its ability to retrieve relevant items correctly. Secondly it is a system that involves users, or more precisely musical users. How do we evaluate this crucial facet?

Coleman has identified and addressed the lack of subjective evaluation factors in concatenative synthesis (Coleman, 2015). In his doctoral thesis he devotes a chapter to a listening survey conducted to determine the quality of a number of different algorithmic components of the system under consideration. He asks the listeners to consider how well the harmony and timbre of the concatenated sequences are retained. In Chapter 3 we conducted a similar-style listening survey to determine the ability of a genetic algorithm to create symbolic rhythmic patterns that also mimic a target input pattern. Evaluation strategies need to be tailored specifically for systems, but if the system is intended to retrieve items according to some similarity metric, and the material is musical, then a listening survey should be critical. Furthermore, and echoing Coleman's work, we would emphasise that whatever the application of a concatenative system, the evaluation of the timbral quality is essential.

We conducted extensive evaluation of our own system, both quantitatively and qualitatively. In the quantitative portion, we set out to investigate two key aspects. First, if we consider the system as a retrieval task that aims to return similar items, how accurate and predictable is the algorithm and its associated distance metric? Second, how does this objective retrieval accuracy correspond to the perceptual response of the human listener to the retrieved items? Also, rather than fit the subjective factor into some sort of haphazard Turing test box, can we determine simply if the listener likes what comes out?

The qualitative evaluation consisted of interactive, informal interviews with intended users — mostly active music producers but also music researchers and students as they used the software. We gathered their responses and impressions and grouped them according to thematic analysis techniques. As alluded to in the introduction, both the quantitative evaluation and the qualitative evaluation have been previously reported in separate publications, but we include summaries of each here for reference.

## 7.2 System Evaluation

We describe here the qualitative portion of the evaluation, first by introducing the experimental setup, then presenting and comparing the results of the algorithm's retrieval accuracy with the listener survey.

### 7.2.1 Evaluation and Experimental Design

Given the rhythmic characteristics of the system we formulated an experiment that evaluated its ability to generate new loops based on acoustic drum sounds. We created a small dataset (Appendix C) of 10 breakbeats ranging from 75 BPM to 142BPM and truncated them to single bar loops. As we introduced in Chapter 2 breakbeats are short drum solo sections from funk music records in the 1970s and exploited frequently as sample sources for hip-hop and electronic music. Chapter 4 also showed that computational use of breakbeats has been of interest to the scientific community, as evident in work by Ravelli et al. (2007), Hockman & Davies (2015) and Collins (2006a).

Each of these loops was then used as a seed loop for the system with the sound palette derived from the remaining 9 breakbeats. Four variations were then generated with 4 different distances to the target. These distances correspond to indices into the similarity matrix we alluded to in Section 2, which we normalise by dividing the index by the size of the table. The normalised distances then chosen were at 0.0 (the closest to the target), 1.0 (the furthest from the target) and two random distances in ranges 0.0 - 0.5 and 0.5 - 1.0.

Repeating this procedure 10 times for each target loop in the collection for each of the distance categories, we produced a total of 40 generated files to be compared with the target loop. Each step in the loop was labelled in terms of its drum content, for example the first step might have a kick and a hi-hat. Each segmented onset (a total of 126 audio samples) in the palette was similarly labelled with its corresponding drum sounds producing a total of 169 labelled sounds. The labellings we used were K = Kick, S = Snare, HH = Hi-hat, C = Cymbal and finally X when the content wasn't clear. Figure 7.1a shows the distribution of onsets by type in the full corpus of segmented units. Another useful statistic is highlighted in Figure Figure 7.1b, which plots the distribution of onsets for each step in the 16 step sequence for the predominant kick, snare and hi-hat for the 10 target loops. Natural trends are evident in these graphs, namely the concentration of the kick on the 1st beat, snares on the 2nd and 4th beat and hi-hats on off beats.

#### 7.2.1.1 Retrieval Evaluation

The aim of the experiment was first to determine how well the system was able to retrieve similarly labelled "units" for each 1/16th step in the seed loop. To evaluate the ability of the algorithm to retrieve correctly labelled sounds in the generated loops we defined the accuracy $A$ by Eq. 7.1, based on a similar approach presented by Thompson et al. (2014). We make a simplification that corresponding HH and X and C labels also yield a match based on our observation that their noisy qualities are very similar, and some of the target loops used did not have onsets sounding at each 1/16th step.

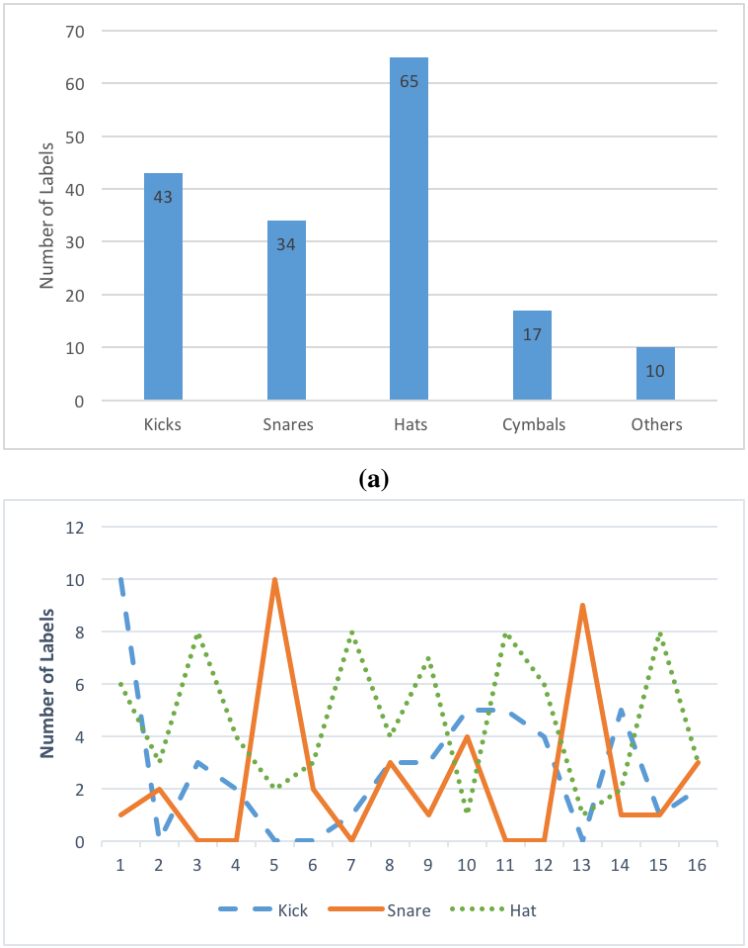
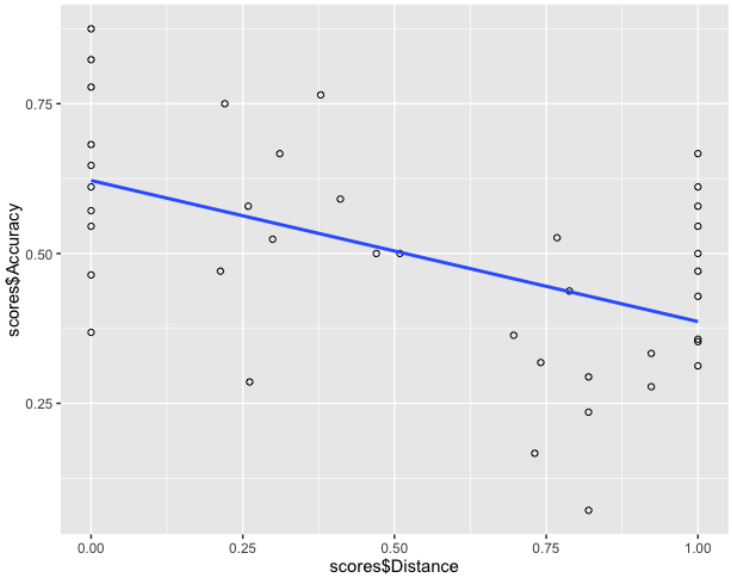



**Figure 7.1:** Mean and standard deviation of fitness versus Likert scores for (a) each stimulus (b) measure and string type

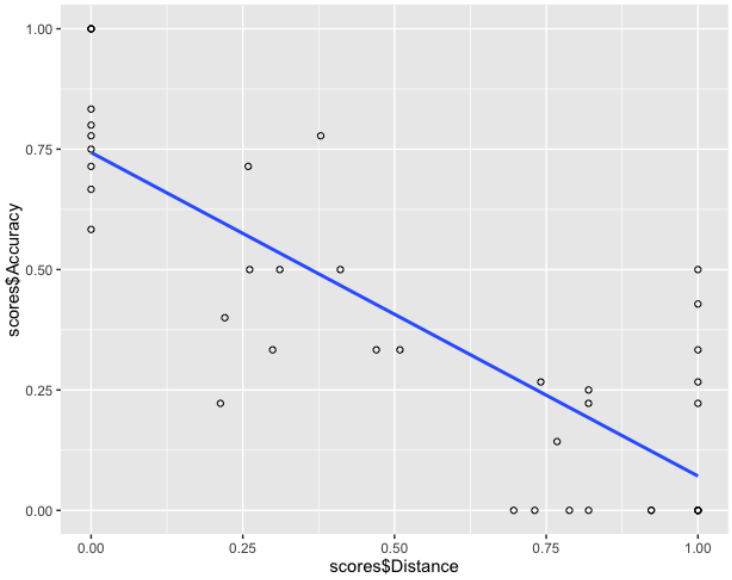
$$A = \frac{number\ of\ correctly\ retrieved\ labels}{total\ number\ of\ labels\ in\ target} \tag{7.1}$$

**Retrieval Results** By studying the Pearson's correlation between the retrieval ratings and the distance, we can confirm the tendency of smaller distances to produce more similar patterns by observing the moderate negative correlation ($\rho = -0.516, p = 0.001$) between increased distance and the accuracy ratings (Figure 7.2a).

An interesting observation is that when we isolate the retrieval accuracy ratings to kick and snare we see this correlation increase sharply to ($\rho = -0.826, p = 0.001$), as can be seen in Figure 7.2b.




**(a)**




**(b)**

**Figure 7.2:** Scatterplot and Regression Line of Retrieval Accuracy for Distance for (a) all Drum Sounds (b) kick and snare isolated

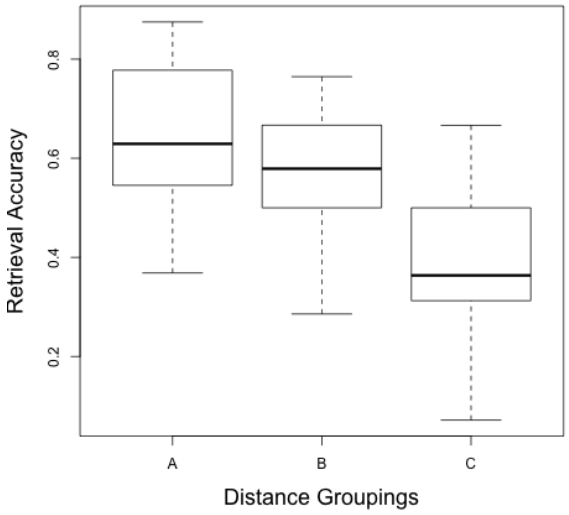



**Figure 7.3:** Central Tendencies of Retrieval Ratings for the Similarity/Distance Categories

Delving into the data further, we can identify 3 different categorical groupings that demonstrate predictable trends in terms of the central tendencies and descending retrieval accuracy (Figure 7.3). We label these categories A, B and C with the breakdown of the number of patterns and their corresponding distance ranges as follows:

- *A* - 10 Patterns - 0.0
- *B* - 9 Patterns - [0.2 - 0.5]
- *C* - 21 Patterns - [0.5 - 1.0]

**Listener Evaluation** The retrieval accuracy gives the objective ratings of the system's capability for retrieving correctly labelled items. This information may not be consistent with the human listener's perceptual impression of similarity, nor does it give any indication whether the retrieved items are musically acceptable or pleasing. To assess the human experience of the sonic output and to compare with the objective ratings of the system, we conducted a listening survey which will be described here.

To directly compare and contrast with the retrieval evaluation the same 40 loops generated by the system and used in the retrieval analysis were transferred to the listening survey. The survey itself was web-based using a modified version of the listening survey tool[89] used in Chapter 3 and took roughly 15-20 minutes to complete.

[89] SeeAppendix B

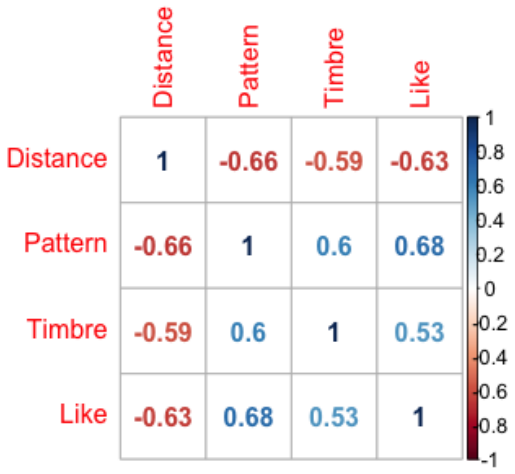



**Figure 7.4:** Correlations Between Distance and Subjective Ratings of Pattern Similarity, Timbre Similarity and Liking

Participants were presented with the seed pattern and the generated patterns and could listen as many times as they liked. Using a 5 point Likert scale the participants were then asked to rate their agreement with the following statements:

- Is the rhythmic pattern similar?
- Is the timbre similar?
- Do you like the loop?

Twenty one participants took part in total, drawn from researchers at the authors' institute of UPF as well as friends and colleagues with an interest in music. Twenty out of the 21 participants declared they were able to play a musical instrument. Ten of the 21 participants specified they played a percussion instrument and 9 reported they could read notation. In the instructions we provided brief explanations of the key terms and audio examples demonstrating contrasting rhythmic patterns and timbres.

**Listener Evaluation Results** The survey data was analysed using Spearman's rank correlation on the mode of the participants' responses to each loop stimulus with the associated distance of that loop. We identified a moderate to strong negative correlation of -0.66, -0.59, -0.63 respectively for each of the pattern, timbre and "likeness" aspects ($p < 0.01$ in all instances). This can be observed in the red values in the correlation matrix presented in Figure 7.4.

It should be evident that the subjective listening data conforms quite well to the findings of the objective retrieval rating. Increased distance resulted in decreased retrieval accuracy which in turn corresponded to a decrease in listener ratings for qualities pertaining to pattern similarity and impression of timbral similarity in the sounds themselves. Furthermore, it was revealed that the aesthetical judgement of the generated loops, encapsulated by the "likeness" factor, also followed the trend set out by the objective algorithm. We were curious to establish whether any particular subject did not conform to this preference for similar loops, but examining the individual correlation coefficients revealed all to be negative (all participants preferred more similar sounding patterns).

## 7.3 User Experience Evaluation

In February 2016 we conducted several days of intensive interactive user discussions with a prototype of our system. The interviews took place at Universitat Pompeu Fabra in Barcelona and at Native Instruments Headquarters in Berlin. These interviews complement the more stringent evaluation of the system in terms of perceptual output and retrieval performance we just saw in the previous section and published in (Ó Nuanáin et al., 2016c). This evaluation found that the retrieval ability to of the system returned greater instances of correctly labelled polyphonic drum sounds with sequences that were more similar to the target using the metric and algorithm previously described. An accompanying listener evaluation revealed that participants' ratings correlated with our metrics in terms of similarity of pattern and timbre to the target. Additionally, the listener ratings found a correlation between closer patterns and subjective preference.

The profiles of the users for the qualitative evaluation could be divided into roughly four categories. In Barcelona, they were mostly researchers on one side, and students with a background in Sound and Music Computing on the other side. Nearly all of these participants worked with digital music in some form or other. In Berlin the users were drawn from Red Bull Music Academy (RBMA)[90] associated artists based in the city, as well as employees of Native Instruments. The RBMA was initiated in 1998 with the aim of gathering young music producers and DJs at worldwide events for the purposes of lectures, workshops and performances. RBMA collaborated in the GiantSteps project (Knees et al., 2015) to provide access to these upcoming artists for research interaction and evaluation. These Berlin participants included producers, promoters, DJs, musicologists, product designers, engineers and graphic designers. United by the influence of the city, nearly everyone identified themselves as producers of techno and/or house, whether it be full-time or as a hobby.

With each participant we explained briefly the instrument and guided them through the process of generating sounds with the instrument. Mostly, the participants

[90] http://www.redbullmusicacademy.com

**Figure 7.5:** Testing Station Setup in at RBMA in Berlin

were eager to start playing with the instrument as soon as possible, which we were more than happy to oblige. Test stations were set up throughout the venue with a laptop, monitor and headphones as shown in Figure 7.5.

While the interviews themselves were kept informal we at least tried to steer the individual sessions with some common questions or themes in order to elicit conversation. These included statements such as:

- Did the overall concept make sense to you?
- Was the interface intuitive? What elements were confusing?
- Would you use this system to make music?
- Would you use this system in production scenarios, live performance or both?
- What did you like, what didn't you like?
- What improvements would you make?
- What features would you like to see?

#### 7.3.0.1 Positive Reactions and Outcomes

Before delving into the specifics, we will first highlight the overall extremely positive feedback received from the participants. The word cloud in Figure 7.6 shows a culmination of some of the frequent positive descriptions participants attached to the system during the course of the tests.


**Figure 7.6:** Wordcloud of most frequent positive descriptions

Some of the more detailed positive remarks give further insights into exactly why the system appealed to them:

> *"It's an excellent tool for making small changes in real time. The interface for me is excellent. This two dimensional arrangement of the different sounds and its situation by familiarity, it's also really good for making these changes."*

> *"I'm really interested in more visual, more graphic interface. Also the fact that you can come up with new patterns just by the push of a button is always great."*

> *"It's inspiring because this mix makes something interesting still, but also I have the feeling I can steal it."*

> *"The unbelievable thing is that it can create something which is so accurate. I wouldn't believe that it's capable of doing such a thing."*

Many of the producers we spoke to reflected a particular trend in dance music at the moment for working with hardware and modular systems. This is often borne out of a desire to break out of the typical computer or digital audio workstation workflow and seek another path for inspiration.

*"I just jam for quite a while and then try to build something that I like and then bring it to computer and then add stuff from computer. You have to jam out really. The biggest issues come with recording."*

The most encouraging outcome from our studies conducted with these users was that the "interesting" and "different" design of our system offers these discouraged users a way "back in" for composing with the computer once again. This was suggested by comments such as:

*"I think something I've been looking to do in terms of experimentation and generating ideas melodically, is looking to go a bit more modular, use some modular stuff. To me, this is a digital form of it."*

*"Yeah. I use a lot of hardware, but if I'd use ... a few disco breaks or something or funk breaks that would be kind of nice, totally."*

#### 7.3.0.2 Thematic Analysis

We will now touch upon some of the common recurring themes that arose during the course of the interviews, and describe our own interpretations and plans to address them in future.

**Usage Scenarios** With respect to specific use cases, users provided some interesting scenarios where they could see the tool being used in their own interest. Numerous users were curious as to the ability to record and analyse live input such as instrumental performance or beatboxing for example.

*"This is great! Ah, but wait. Does it mean I could like beat box really badly some idea that I have... and then bring my samples, my favourite kits and then it will just work?"*

Live performance input was not something we had previously considered but is theoretically possible since the host would handle the capture of input audio. It would however require continuous re-analysis and computation of the similarity matrices which could be computationally costly. Still, other users have also expressed a desire for the possibility to continuously analyse incoming audio so it will be investigated.

Quite a number of users weren't interested in using the targeting capability of the synthesis engine whatsoever, and wondered if it was possible to start building patterns from scratch by selecting the onsets manually one by one. For example, referring to the fact that the dry signal is the original and the wet signal is the concatenated output:

*"I've got this fully on wet straight away, which tells you the direction I'd be going with it."*

*"...you just want to drag in 100 different songs and you just want to explore without having this connection to the original group. Just want to explore and create sound with it."*

This is entirely possible; in fact, we had another "exploration" mode previously that gave the option to scan and audition the timbre space with a circular mouse radar that triggered the enclosed sounds, CataRT style. The motivation for this was to allow users to explore and audition the timbre space freely to identify regions of interest before proceeding to build their patterns. Merging this auditioning ability to create a sequence of patterns from zero would also stem the frustration of many of the users who wanted to create sounds straight away without capturing target input.

**Traditional Navigation** A very early outcome of the user testing was the realisation that although users were more than open to this new way of dealing with their sound, they still wanted a link to the familiar - the 2D waveform/timeline paradigm they are so used to dealing with in existing tools such as DAWs.

*"It's a bit hard to figure out which sixteenth you are looking for, because you are so used to seeing this as a step grid."*

*"It's kind of good to see a different interface and not always follow the same timeline,..., But it could just be mirrored in a timeline"*

*"You have a waveform or something... Then I know, okay, this is the position that I'm at."*

*"Is there also a waveform place to put the visualisation? People are so used to having that kind of thing."*

Initially this timeline was not something we had intended on offering; after all isn't it these paradigms we're seeking to break away from? After hearing these comments we realised that it was a useful option for the user and was one of the first items we implemented subsequently, as can be seen from Figure 6.6.

**Shaping the Sounds** Other than generating the sequences and rearranging individual units in the sequence, the synthesiser offers no additional ways to modify the output sound (discounting the ability to mix between the target sequence and the generated sequence). Many users agreed it would be useful to be able to manipulate these individual sounds sonically somehow. Most crucially they desired the option to

be able to control the envelopes of the individual units via drawable Attack and Decay parameters.

> *"... an attack and decay just to sort of tighten it up a little bit. Get rid of some of the rough edges of the onsets and offsets."*

> *"Yeah, the thing is if you listen to it now, there's kind of a rhythm going, but it would be great if you could increase the decay of the snare for example. Which if it's prototype, you can't expect to have all those functions there immediately, but in an end product, I think it would be a necessity."*

**State and Feedback** One of the consistent items that concern users of generative systems is the notion of state. On a superficial level state can refer to an effective preset management system that stores their efforts, for as one participant notes "you're always afraid of losing something.". Users are terrified of losing their progress once they've entered a pleasing "state", although this is a much bigger concern in probabilistic systems that produce - but then may never reproduce - "happy accidents".

At present our system has no state, and this was something frequently remarked upon and something we are actively considering. Users expressed a desire for complex state operations. Comparing two generated sequences visually and sonically for example, and being able to mix or find an interpolation between the two of them somehow. One artist wondered whether would it be possible to extend the graph visualisation technique to a space of "patterns". In this manner a series of stored patterns would be plotted in 2D space one after each other, and could be explored and sequenced in a more high-level arrangement or score-type interaction with the instrument. As he explained:

> *"Even with this if it wasn't actually blending or interpolating between points, but just so you could save the state in a composition screen of dots and you could just jump between."*

This was actually inspired by the artist's own experiences with another experimental but commercial tool for music production: Audiomulch[33]. It includes a unique feature known as the MetaSurface, which allows navigation and interpolation of multiple parameter states by manipulating a colourful visual cluster space.

Also related to issue of state was the ability to initiate feedback, i.e. continuously assigning the concatenated output to the input, which once again can be done manually by recording to the host and re-recording in. This could be facilitated in the tool itself, but would require having the option of removing the matching sounds from the corpus itself to encourage diversity of sound.

**Parameterisation and Visualisation** One of the recurring difficulties that faced participants was our presentation of the parameters. As we explained briefly in the implementation section, the user is able to control the influence of the four features in concatenation algorithm and the PCA visualisation. We relabelled the features from their objective names to what we considered a subjective equivalent that a lay user may understand. MFCCs are labelled as "Timbre", spectral centroid as "Brightness", spectral flatness as "harmonicity" with Loudness unchanged.

Unfortunately, users expressed confusion at their purpose and were unable to interpret their effect on neither the arrangement of the units of sound in space or their resulting effect on the patterns generated. For instance:

*"The problem is I'm a little bit lost already."*

*"you have four parameters, and you don't know which thing is what"*

*"I would prefer not to have too much controls."*

Presenting this additional complexity was a perhaps naive inclusion on our part. Clearly the typical user is content with the overall output from the system and would rather not delve into these specifics. However, at least in terms of the visualisation there is a "sweet spot" for feature weightings in the arrangement of the units of sound in the timbre space and this is why the controls were made available. The weightings can vary greatly depending on the corpus, though in our experience MFCC alone often provide the best separation and clustering.

The challenge will be to find the best approach to arranging the units in sound with the best separation and shielding these parameters from the user. Potentially, a way forward could be to remove these sliders and replace them with a number of options including an "advanced" mode with the ability to select specific parameters for the axes like CataRT in addition to "automatic" arrangement presets made possible using dimensionality reduction techniques. An area for study would be to gather many different sound sets and try various combinations of feature selections and weightings to find the best visual separation. At present PCA is used for dimension reduction but as we reported in Chapter 6 there also exists MDS and t-SNE which we have yet to offer to the user. In our informal experiments with the latter we have noticed two immediate drawbacks in its applicability for real-time or at least 'production ready' usage. Firstly, a successful projection, encompassing good separation with minimal overlapping of clusters, is heavily dependent on tweaking the parameters of the system to the data provided. Secondly, the algorithm is complex enough that it needs to pass through a number of iterations before it finally settles on a projection with minimal threshold error, thus there is a delay in the time it takes for it to project (however we have observed that this makes a rather nice dynamic animation when its processing routine is placed in the interface update loop!).

# Chapter 8

# Conclusions and Future Work

In this dissertation we set out to examine computational strategies aimed at a very well-defined objective: the generation of rhythm-centric *loop*s that are symptomatic of modern dance music composition. Dance music is an unfolding musical style and an emerging sociocultural culture and phenomenon that is still very much in its infancy. Only recently has it begun to be treated with the legitimate discourse it deserves in academic narrative and forums. Equally, computational music analysis and research have been slow to respond to its needs for investigation. We hope that we have raised the profile of both these aspects in addition to several contributions to general computation musical analysis and synthesis which we summarise here.

## 8.1 Contributions

### 8.1.1 Tools for Symbolic Rhythmic Algorithmic Composition

The literature abounds with esoteric systems for algorithm composition in the symbolic domain. The unique contribution we offer to this field a highly-specialised genetic algorithm that addresses perceptual aspects of rhythm in addition to documented listener evaluation of its applicability.

### 8.1.2 Evaluation and Expansion of Cepstrum-Based Timbre Analysis

MFCCs are frequently assumed to be the *de facto* descriptor for analysing timbre in computer music and MIR literature. In choosing appropriate features for concatenative synthesis tasks, we challenged this assumption and provided experimental evidence that other methods should be given serious consideration. This was achieved through an implementation of the BFCC method along with two contrasting datasets to confirm its accuracy. While our newly implemented BFCC extractor improves over the existing MFCC one, our experiment also demonstrated that the GFCC is the best performing overall. In the spirit of reproducibility we provide all the data used and the associated scripts for other researchers to validate.

### 8.1.3 *k*-Best Unit Selection

Unit selection is a vital procedure in concatenative synthesis of sounds from a corpus with a target sound as reference. Viterbi decoding of HMM-style representations of a concatenative synthesis task provide the means of maximising the similarity of candidate units from the corpus to each unit in the target sound, while ensuring continuity and stability of consecutive candidate units placed in sequence. The drawback of the Viterbi algorithm is that it only outputs one optimised sequence which is not conducive to creative musical application.

To this end, we proposed the application of two algorithms from speech processing and graph theory that have hitherto not been investigated in musical concatenative synthesis tasks. These are known as the Parallel LVA and *k*-Shortest paths respectively and we provide reference implementations of both as well as demonstrations of their usage in *k*-Best unit selection of multiple candidates.

### 8.1.4 A New User-Focussed Tool for Rhythmic Concatenative Synthesis

As well as contributing improvements to the state of the art in the research of concatenative synthesis we also provided a practical realisation of concatenative synthesis methods that addresses the unique requirements of the typical dance music producer. We demonstrated how state of the art methods can be refined and condensed into a 'production ready' system, along with the proposal of a novel graph-based interaction metaphor that can cope with the specificities of rhythm.

### 8.1.5 An Evaluation Methodology for Concatenative Synthesis

The final contribution of our thesis concerned the evaluation of concatenative synthesis systems, which we maintain is still under-examined in its academic dissemination. The evaluation conducted in Chapter 7 sought to address this by providing a formal methodological framework that appraised two vital perspectives:

1. Quantitative:

   *a*) The objective retrieval of returning labelled units in the correct sequence according to a reference loop.

   *b*) The corresponding subjective response to the sequences given the reference loop's pattern and timbre.

2. Qualitative: Overall impressions and thematic analysis of the users' experience with using the instrument and interface.

## 8.2 Future Work

Throughout the thesis we have alluded to a myriad avenues that constitutes valuable work, but here we summarise three of the most outstanding areas as we see them at the present.

### 8.2.1 Optimisation and Evaluation of *k*-Best Unit Selection

We are really excited with the novelty and possibilities presented with *k*-Best strategies for unit selection in concatenative synthesis. It is an area of speech and music processing that is under-explored and seldom reported in the literature. Of course the major issue we identified is poor performance and scalability for large corpora. We propose two strategies for resolving this that can constituted a future work:

1. Implementation of the Serial Decoder, which is purported to be more efficient than the Parallel Decoder.

2. Pre-pruning of units to eliminate unrelated units to the current context of the target sequence.

### 8.2.2 Vertical Concatenative Synthesis

In Chapter 5 we briefly outlined the possibilities of *vertical*-oriented concatenative synthesis that considers supplanting units of sound on top of each other in addition to their horizontal concatenation. Of particular interest here is the role that recent improvements in source separation of signals that can also perform *vertical* segmentation of units into constituent elements in the frequency domain for later combination.

### 8.2.3 Feature Implementation and Improvement of RhythmCAT

The qualitative evaluation of RhythmCAT provided in the previous chapter raised a whole range of interesting possibilities regarding improvements to RhythmCAT. Some of these include:

- Live capture of loops from external instruments and sources.

- Envelope editing and effects processing to transform selected units sonically.

- High-level visualisation and interpolation of *patterns* rather than the lower-level visualisation of individual units and onsets.

### 8.2.4 A Final Reflection

A doctoral dissertation is often likened to an apprenticeship, and I can safely conclude that after four-odd years foraging in the depths of academic wilderness I have come out the other side with an immense appreciation of just how much I have achieved, *yet*, also the slow realisation that I have, perhaps, just scratched the surface.

As computer music practitioners we are among the lucky few to be working on what we truly love, but sometimes we lose sight of the crotchets and quavers as we sift through the ones and zeros. One of my proudest outcomes with this work is that I have discovered some new ways of making music and tools that can help achieve them. This I look forward to most of all.

# Appendix A

# Publications by the Author

### Peer-reviewed journals

- **Ó Nuanáin, C.**, Herrera, P., & Jordà, S. (2017). Rhythmic Concatenative Synthesis for Electronic Music: Techniques, Implementation, and Evaluation. *Computer Music Journal*, 41(2), pp. 21–37.
- Vogl, R., Leimeister, M., **Ó Nuanáin, C.**, Jord, S., Hlatky, M., & Knees, P. (2016). An Intelligent Interface for Drum Pattern Variation and Comparative Evaluation of Algorithms. *Journal of the Audio Engineering Society*, 64(7), pp. 503–513.

### Full articles in peer-reviewed conferences

- **Ó Nuanáin, C.**, Jordà, S., & Herrera, P. (2017). k -Best Hidden Markov Model Decoding for Unit Selection in Concatenative Sound Synthesis. In *Proceedings of the International Symposium on Computer Music Multidisciplinary Research (CMMR)*. Porto, Portugal.
- **Ó Nuanáin, C.**, Herrera, P., & Jordà, S. (2016). An Evaluation Framework and Case Study for Rhythmic Concatenative Synthesis. In Proceedings of the International Society for Music Information Retrieval Conference (ISMIR). New York, USA.
- **Ó Nuanáin, C.**, Jordà, S., & Herrera, P. (2016). An Interactive Software Instrument for Real-time Rhythmic Concatenative Synthesis. In Proceedings of the International Conference on New Interfaces for Musical Expression (NIME). Brisbane, Australia.
- **Ó Nuanáin, C.**, Jordà, S., & Herrera, P. (2016). Towards User-Tailored Creative Applications of Concatenative Synthesis in Electronic Dance Music. In *Proceedings of the International Workshop on Musical Metacreation (MUME)*. Paris, France.

- **Ó Nuanáin, C.**, Herrera, P., & Jordà, S. (2015). Target-Based Rhythmic Pattern Generation and Variation with Genetic Algorithms. In *Proceedings of the Sound and Music Computing Conference (SMC)*. Maynooth, Ireland.
- **Ó Nuanáin, C.**, & Sullivan, L. O. (2014). Real-time Algorithmic Composition with a Tabletop Musical Interface - A First Prototype and Performance. In *Proceedings of Audio Mostly: A Conference on Interaction With Sound*. Aalborg, Denmark.

### Other contributions to conferences

- Faraldo, Á., **Ó Nuanáin, C.**, Gómez, D., Herrera, P., & Jordà, S. (2015). Making Electronic Music with Expert Musical Agents. In *Late-Breaking Demo Session of the International Society for Music Information Retrieval Conference (ISMIR)*. Málaga, Spain.
- **Ó Nuanáin, C.**, Hermant, M., Faraldo, Á., & Gómez, D. (2015). The EEEAR: Building a Real-Time MIR-Based Instrument from a Hack. In *Late-Breaking Demo Session of the International Society for Music Information Retrieval Conference (ISMIR)*. Málaga, Spain.

# Appendix B

# Listening Survey Web Tool

Over the course of this doctoral thesis a need arose for a flexible system that could perform comparative listening surveys and evaluation with ease. As a consequence, a web platform was developed that is easy to setup, customise, configure and deploy etc. The main advantage of this is that surveys can be conducted remotely by the user without the requirement of supervision and physical presence. The tool and its source code is freely available online [91], and has since been used in other researchers' works (Marti, 2015; Jade, 2016).

An initial landing page (Figure B.1) provides instructions for the participants along with an estimation of the time and the number of questions. The participant then fills out personal details that are gathered anonymously - no names or contact details are recorded - followed by some descriptive questions pertinent to the survey being conducted. For instance, in the case of our rhythm surveys we were interested to know the musical training of each participant and whether that was concentrated in more percussive practice.

The questionnaire commences (Figure B.2) with a series of randomised questions consisting of a target sound to be listened to and several other generated sounds that necessitate scrutiny. Repeated listening is possible before making a final decision on the Likert score for each question.

To make the tool as easy as possible to deploy, database dependency is avoided. Subsequent to each participant's concluded survey session, a CSV file of the completed questionnaire results are simply mailed to the researcher's email address.

[91] https://github.com/carthach/listener_survey_tool

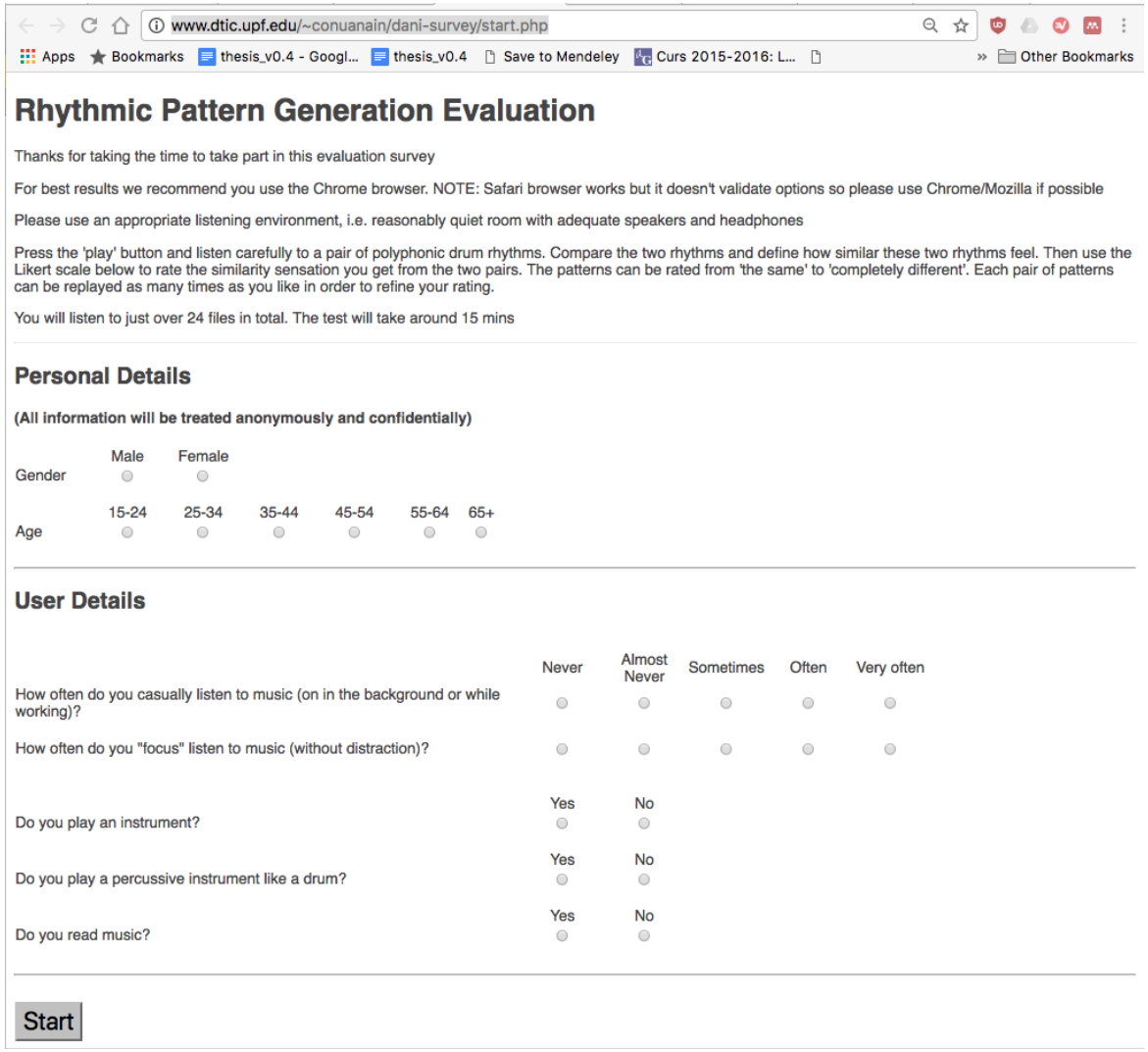



**Figure B.1:** Web Survey Landing Page

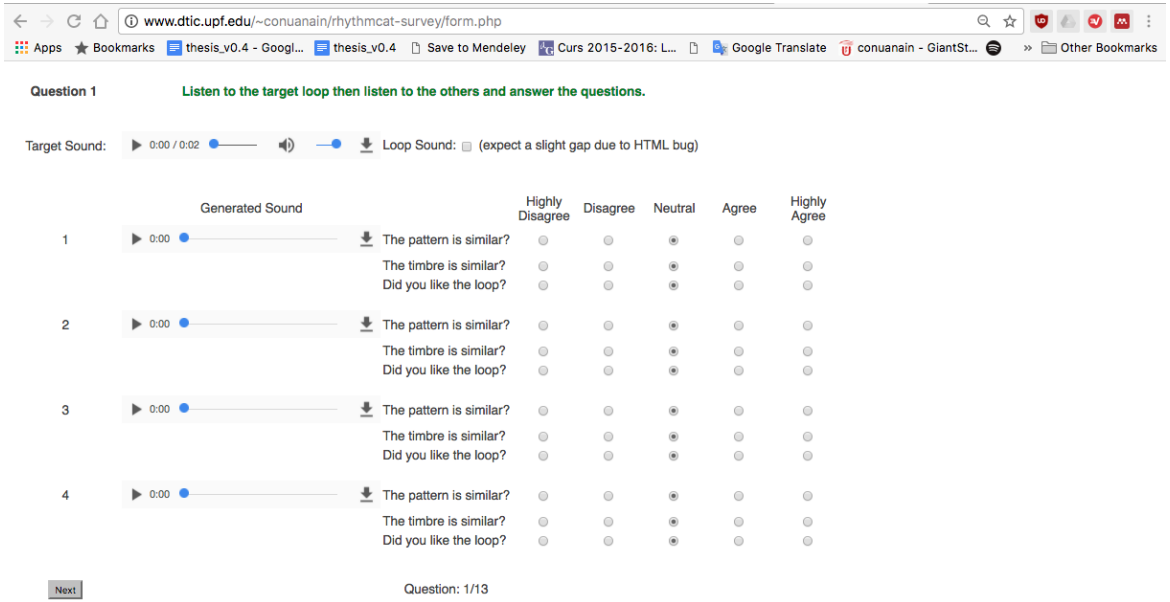



**Figure B.2:** Web Survey Question Page

# Appendix C
# Resources

## Code and Tools

### Implementation of Methods

Implementation of the methods presented in this thesis are made publicly available for research purposes. We also indicate (in squared brackets) the language in which the implementation is available. Other frameworks and libraries are also listed.

- SimpleGA [C++]
- RhythmGA [C++]
- GenDrum [Max/MSP and Max for Live]
- kViterbi [Python]
- PyConcat [Python]
- RhythmCAT [C++ (JUCE)]

### Tools

- Essentia audio analysis library (http://essentia.upf.edu/)
- Librosa audio analysis library (https://github.com/librosa/librosa)
- Madmom audio analysis library (https://github.com/CPJKU/madmom)
- Scikit-learn machine learning library (http://scikit-learn.org/stable/)
- OpenCV computer vision library (https://opencv.org/)
- Boost Graph Library (http://boost.org/)
- JUCE cross platform C++ framework (http://juce.com/)

## Corpora and Datasets

Corpora and datasets used throughout the thesis are summarised here and are categorised by those contributed by the authors as well as those that were made available by other researchers.

### Author Contributed

- 'Orchestral Timbre' collection used in the classification experiment (Section 5.3.4). Sound files, extracted features, classifier models and IPython notebooks can be downloaded here:
  http://github.com/carthach/phd_thesis/tree/master/classification/drums/data
- 'Drum Timbre' collection used in the classification experiment (Section 5.3.4). Sound files, extracted features, classifier models and IPython notebooks can be downloaded here:
  github.com/carthach/phd_thesis/tree/master/classification/orchestra/data
- 'Breakbeat' collection used in the evaluation of RhythmCAT (Section 7.2.1). Sound files and annotations can be downloaded here:
  http://www.github.com/carthach/thesis/experiments/evaluation/data

### Public Datasets

- ENST-Drums: Provided by Télécom ParisTech (Gillet & Gaël, 2006), contains an audio-visual annotated dataset of performances by 3 professional drummers using multi-track studio recording in a variety of configurations (single hits, phrases and accompaniments) and a variety of styles including jazz and rock. Summary and ordering details are available here:
  http://perso.telecom-paristech.fr/~grichard/ENST-drums/
- Modal (Musical Onset Dataset And Library): Hand annotated, contains 501 onsets across 71 files containing a mix of mostly monophonic events (Glover et al., 2011)
  Datset can be downloaded from:
  https://github.com/johnglover/modal
- JKU: Contains 321 files with 27,774 total onsets. Compiled from a number of different sources by Sebastian Böck for evaluating his SuperFlux algorithm (Böck & Widmer, 2013). The algorithm was developed especially to handle vibrato, and accordingly the dataset contains a large number of samples that purposely addressing vibrato, such as samples from opera or classical technique strings.
  Annotations and some audio can be downloaded from here:
  https://github.com/CPJKU/onset_db

# Appendix D

# The PyConcat Library

## D.1 Description

The PyConcat[92] library is a Python environment for conducting research-oriented concatenative synthesis of signals. It uses feature extraction routines from Essentia and Librosa as well as distance measures from scipy and novel unit selection via *k*-Best Viterbi.

The basic pipeline for performing concatenative synthesis with this tool is as follows.

1. **Segmentation**
   - Framewise in the time domain or using FFT/IFFT resynthesis
   - Onsts
   - Beats
2. **Feature Analysis**
   - Temporal: loudness, attack
   - Spectral: flatness, centroid
   - Timbral: BFCCs, MFCCs, GFCCs
   - Musical: f0, HPCPs
3. **Unit Selection**
   - Linear Search
   - *k*-D Tree
   - Viterbi
   - *k*-Best Viterbi

[92] Not to be confused with https://github.com/thomasballinger/pyconcat a script for concatenating Python sripts. I really need to come up with more original names for my software!

## D.2 API

A concatenative synthesis task is performed in stages in accordance with the previous pipeline. Separate statements segment the units and compute features before finally combining a sequence with unit selection. This also allows the flexible separation of unit scale segmentation and features that allow different specifications for target and corpus unit.

The code snippet below shows an example of extracting units from the target at the beat unit scale while concatenating the top 10 sequences at the onset scale using Viterbi decoding and time-stretching.

```
e = Extractor.Extractor()

#Find all the corpus files in the path
corpusFiles = extractor.getListOfWavFiles(corpusPath)

#Extract features and units
tFeatures, tUnits, tUnitTimes = e.analyseFile(targetFile,
                                              scale="beat")

cFeatures, cUnits, cUnitTimes = e.analyseFiles(corpusFiles,
                                              writeOnsets=True,
                                              scale="onsets")

#Get list of candidate sequences with k-Best Viterbi Decoding
sequences = unitSelection.unitSelection(tFeatures,
                                        cFeatures,
                                        method="kViterbiParallel",
                                        normalise="MinMax",
                                        topK=10)

# Concatenate the sequences with time stretching
audio = e.concatOnsets(sequences,
                       corpusUnits,
                       targetUnits,
                       stretchUnits=True)

e.writeAudio(audio, "out.wav")
```

## D.3 Links

- Documentation - http://pyconcat.readthedocs.io/en/latest/source/PyConcat.html
- Library - https://github.com/carthach/pyconcat

# Appendix E
# Related Applications

## E.1 The Eear - Enhanced DJ Assistant

The Eear[93] was prototyped by members of The GiantSteps team over a 24-hour period at the Sónar Music Hack Day in 2014 and subsequently won the MusicBricks category for best hack. A more refined version of the system was developed for mobile and desktop and selected for presentation at the Music Tech Fest in Slovenia, as well as the late breaking demo session at ISMIR 2015, in Málaga (Ó Nuanáin et al., 2015). A lot of the feature extraction and segmentation logic that was described in this thesis was also used to develop this system.

The Eear comprises a mobile application (Figure E.1 - left) that uses the microphone of the device to capture live input from any source and perform harmonic analysis (using a rolling average of the HPCP vector) in order to estimate the current chord or key. This key and scale information is sent via OSC to a VST plugin (Figure E.1 - right) that allows MPC style sampling of related sounds filtered by that very same tonal information - thus it selects units that are always 'in key' with the live input captured by the mobile.

## E.2 Atomix

For the 2017 edition of the Sónar Innovation Challenge[94], members of the MTG team proposed and mentored a project that challenged selected participants to prototype a system that somehow combined the spirit of timbre space driven concatenative synthesis with state of the art source separation technology.

While the team eventually devised their own tools and did not implement any source separation, the concept art by Olivia Andrieux (Figure E.2) shows how it could be realised visually with RhythmCAT-inspired visualisation and interaction for each track of stream of audio.

[93] http://carthach.github.io/eear/
[94] http://sic.upf.edu/challenges/


**Figure E.1:** The Snitch, mobile app (left), plugin interface (right)

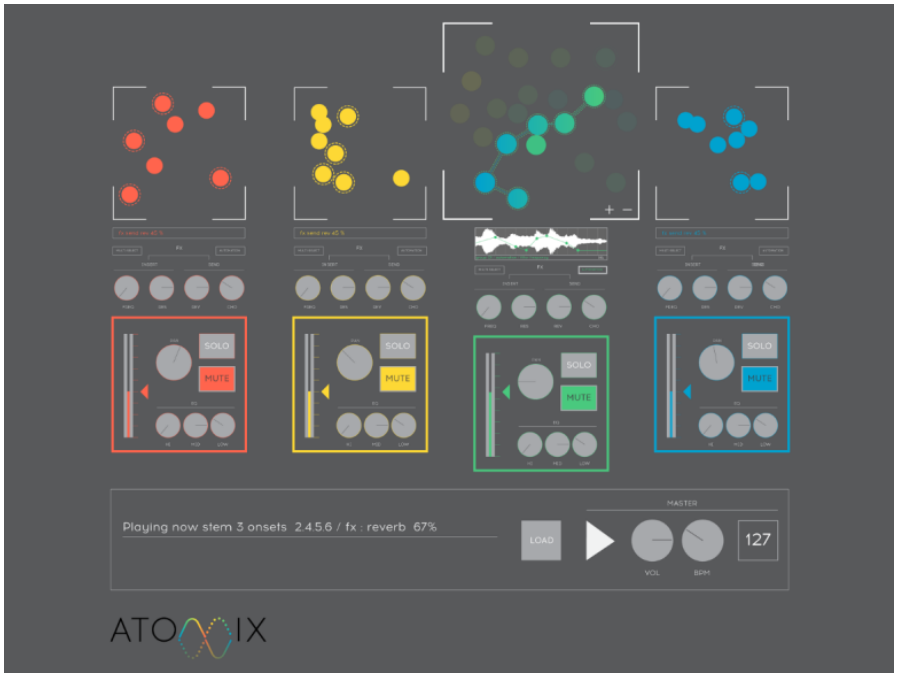



**Figure E.2:** Atomix User Interface. Image from Centre for Digital Music at Queen Mary.

# Appendix F

# Glossary

## F.1 Acronyms

| | |
|---|---|
| $k$-NN | $k$-Nearest Neighbours |
| ACMC | Australasian Computer Music Conference |
| AI | Artificial Intelligence |
| ANN | Approximate Nearest Neighbour |
| ANN | Artificial Neural Network |
| ANOVA | Analysis of Variance |
| BFCC | Bark-Frequency Cepstral Coefficient |
| BPM | Beats Per Minute |
| CNN | Convolutional Neural Network |
| DAG | Directed Acyclic Graph |
| DAW | Digital Audio Workstation |
| dB | deciBel |
| DCT | Discrete Cosine Transform |
| EBU | European Broadcasting Union |
| EDM | Electronic Dance Music |
| EMC | Elementary Markov Constraints |
| EMI | Experiments in Musical Intelligence |
| ENST | École Nationale Supérieure des Telecommunications, nowadays Télécom ParisTech |
| ERB | Equivalent Rectangular Bandwidth |
| f0 | Fundamental Frequency |
| FFT | Fast Fourier Transform |
| FM | Frequency Modulation |
| GA | Genetic Algorithm |
| GFCC | Gamma-Frequency Cepstral Coefficient |
| GTCC | Gammatone Cepstrum Coefficients (alternative acronym to GFCC) |
| GTTM | General Theory of Tonal Music |

| | |
|---|---|
| HCI | Human Computer Interaction |
| HFC | High Frequency Content |
| HMM | Hidden Markov Model |
| HPCP | Harmonic Pitch-Class Profiles |
| HTK | Hidden Markov Model Toolkit |
| IDM | Intelligent Dance Music |
| IFFT | Inverse Fast Fourier Transform |
| IIR | Infinite Impulse Response |
| IOI | Inter-Onset Interval |
| IRCAM | Institute for Research and Coordination in Acoustics/Music |
| JKU | Johannes Kepler University |
| LISP | LISt Processor |
| LVA | List Viterbi decoding Algorithm |
| MCM | Markov Chain Monte Carlo |
| MDS | Multi-Dimensional Scaling |
| MFCC | Mel-Frequency Cepstral Coefficient |
| MIDI | Musical Instrument Digital Interface |
| MIR | music information retrieval |
| MIREX | Music Information Retrieval Evaluation eXchange |
| MPC | Music Production Controller |
| MTG | Music Technology Group (at UPF) |
| NIME | New Interfaces for Musical Expression |
| OSC | Open Sound Control |
| PCA | Principal Component Analysis |
| PCoA | Principal Coordinate Analysis |
| PCP | Pitch-Class Profiles |
| Pd | Pure Data |
| PoS | Part of Speech |
| RBMA | Red Bull Music Academy |
| RMS | Root Mean Square |
| SMC | Sound and Music Computing |
| SNE | Stochastic Neighbour Embedding |
| STEIM | Studio for Electro-Instrumental Music |
| STFT | Short-Time Fourier Transform |
| SVM | Support Vector Machines |
| t-SNE | t-Distributed Stochastic Neighbour Embedding |
| TLU | Threshold Logic Unit |
| TTS | Text-to-Speech |
| TUBS | Time Unit Box System |
| UI | User Interface |
| UPF | Universitat Pompeu Fabra |
| VST | Virtual Studio Technology |
| WAV | Wave Audio Format |

| | |
|---|---|
| WDR | West German Radio Studios |
| WNBD | Weighted Note-to-Beat Distance |

# Bibliography


Abdulla, W. H. (2002). Auditory Based Feature Vectors for Speech Recognition Systems. *Advances in Communications and Software Technologies*, pp. 231–236. [Cited on pages 100 and 151.]

Agostini, G., Longari, M., & Pollastri, E. (2003). Musical instrument timbres classification with spectral features. *Eurasip Journal on Applied Signal Processing*, *2003*(1), 5–14. [Cited on page 106.]

Agres, K., Forth, J., & Wiggins, G. A. (2016). Evaluation of musical creativity and musical metacreation systems. *Computers in Entertainment*, *14*(3), 1–33. [Cited on page 88.]

Albiez, S. & Pattie, D. (2010). *Kraftwerk: music non-stop*. Bloomsbury Publishing USA. [Cited on page 11.]

Alonso, M., Richard, G., & David, B. (2007). Tempo estimation for audio recordings. *Journal of New Music Research*, *36*(1), 17–25. [Cited on page 82.]

Alwakeel, R. (2009). IDM as a "Minor" Literature: The Treatment of Cultural and Musical Norms by "Intelligent Dance Music". *Dancecult: Journal of Electronic Dance Music Culture*, *1*(1), 1–21. [Cited on page 14.]

Ames, C. (1990). Statistics and compositional balance. *Perspectives of new music*, *28*(1), 80–111. [Cited on page 19.]

An, S. S., James, D. L., & Marschner, S. (2012). Motion-driven concatenative synthesis of cloth sounds. *ACM Transactions on Graphics*, *31*(4), 1–10. [Cited on page 73.]

ANSI (1994). Psychoacoustic Terminology: Timbre. *American National Standards Institute*. [Cited on page 89.]

Antonopoulos, I., Pikrakis, A., & Theodoridis, S. (2007). Self-similarity analysis applied on tempo induction from music recordings. *Journal of New Music Research*, *36*(1), 27–38. [Cited on page 81.]

Apel, W. (1969). *Harvard dictionary of music*. Harvard University Press. [Cited on page 49.]

Appel, A. (2002). *Jazz modernism: from Ellington and Armstrong to Matisse and Joyce*. Knopf. [Cited on page 57.]

Arar, R. & Kapur, A. (2013). A History of Sequencers: Interfaces for Organizing Pattern-Based Music. *Proceedings of the Sound and Music Computing Conference*, *2*, 383–388. [Cited on page 27.]

Arewa, O. B. (1979). From JC Bach to hip hop: Musical borrowing, copyright and cultural context. *Hofstra Law Review*, *7*(2), 243–258. [Cited on page 60.]

Ariza, C. (2009). The Interrogator as Critic : The Turing Test and the Evaluation of Generative Music Systems. *Computer Music Journal*, *33*(2), 48–70. [Cited on page 17.]

Arom, S. (2004). *African polyphony and polyrhythm: musical structure and methodology*. Cambridge university press. [Cited on page 50.]

Assayag, G., Rueda, C., Laurson, M., Agon, C., & Delerue, O. (2006). Computer-assisted composition at IRCAM: From PatchWork to OpenMusic. *Computer*, *23*(3). [Cited on page 23.]

Atkinson, R. (2007). Ecology of sound: The sonic order of urban space. *Urban Studies*, *44*(10), 1905–1917. [Cited on page 10.]

Aucouturier, J.-J. & Pachet, F. (2005). Ringomatic: A Real-Time Interactive Drummer Using Constraint-Satisfaction and Drum Sound Descriptors. *Proceedings of the International Conference on Music Information Retrieval*, pp. 412–419. [Cited on pages 67, 70, and 151.]

Balestri, M., Pacchiotti, A., Quazza, S., Salza, P. L., & Sandri, S. (1999). Choose the best to modify the least: a new generation concatenative synthesis system. In *Sixth European Conference on Speech Communication and Technology*. [Cited on page 62.]

Barbieri, G., Pachet, F., Roy, P., & Esposti, M. D. (2012). Markov constraints for generating lyrics with style. In *Proceedings of the 20th European Conference on Artificial Intelligence*, pp. 115–120. IOS Press. [Cited on page 21.]

Barbosa, J., Malloch, J., Wanderley, M. M., & Huot, S. (2015). What does " Evaluation " mean for the NIME community? *NIME 2015 - 15th International Conference on New Interfaces for Musical Expression*, p. 6. [Cited on page 151.]

Barreiro, D. L. (2010). Considerations on the handling of space in multichannel electroacoustic works. *Organised Sound*, *15*(3), 290–296. [Cited on page 139.]

Bartók, B. (1993). Hungarian Folk Music. In *Béla Bartók Essays*, pp. 3–4. [Cited on page 56.]

Bates, E. (2009). The Composition and Performance of Spatial Music. *PhD Thesis*, (August). [Cited on page 56.]

Battier, M. (2007). What the GRM brought to music: from musique concrète to acousmatic music. *Organised Sound*, *12*(03), 189–202. [Cited on page 56.]

Bello, J. & Daudet, L. (2005). A tutorial on onset detection in music signals. *IEEE Transactions on Audio, Speech, and Language Processing*, *13*(5), 1035–1047. [Cited on pages 75 and 142.]

Bello, J. P., Duxbury, C., Davies, M., & Sandler, M. (2004). On the use of phase and energy for musical onset detection in the complex domain. *IEEE Signal Processing Letters*, *11*(6), 553–556. [Cited on page 74.]

Benward, B. & Saker, M. (2008). *Music: In Theory and Practise*. McGraw-Hill, eighth edn. [Cited on page 2.]

Bernardes, G. (2010). Style Emulation of Drum Patterns by Means of Evolutionary Methods and Statistical Analysis. *Proceedings of the Sound and Music Computing Conference*, pp. 1–4. [Cited on page 35.]

Bernardes, G., Guedes, C., & Pennycook, B. (2013). EarGram : An Application for Interactive Exploration of Concatenative Sound Synthesis in Pure Data. *From Sounds to Music and Emotions*, pp. 110–129. [Cited on pages 62, 64, 70, 79, 86, and 132.]

Bernstein, L. (1977). *The Unanswered Question: Six Talks at Harvard*, vol. 63. Cambridge. [Cited on page 22.]

Beutnagel, M., Conkie, A., Schroeter, J., Stylianou, Y., & Syrdal, A. (1999). The AT&T next-gen TTS system. In *Joint meeting of ASA, EAA, and DAGA*, vol. 1, pp. 18–24. Berlin, Germany. [Cited on page 62.]

Biles, J. A. (1994). GenJam : A Genetic Algorithm for Generating Jazz Solos. In *International Computer Music Conference*, pp. 131 – 137. [Cited on pages 20 and 26.]

Biles, J. A. (2001). Autonomous GenJam: eliminating the fitness bottleneck by eliminating fitness. In *Proceedings of the 2001 Genetic and Evolutionary Computation Conference Workshop Program*. San Francisco, USA. [Cited on page 26.]

Biles, J. A. (2002). GenJam: Evolution of a jazz improviser. *Creative evolutionary systems*, *168*, 2. [Cited on page 26.]

Biles, J. A. (2003). GenJam in perspective: a tentative taxonomy for GA music and art systems. *Leonardo*, *36*(1), 43–45. [Cited on page 26.]

Biles, J. A. (2007). Improvizing with genetic algorithms: GenJam. In *Evolutionary Computer Music*, pp. 137–169. Springer. [Cited on page 26.]

Bilmes, J. (1993). Timing is of the essence: Perceptual and computational techniques for representing, learning, and reproducing expressive timing in percussive rhythm. *Sciences-New York*. [Cited on page 33.]

Bird, S. (2016). NLTK : The natural language toolkit NLTK : The Natural Language Toolkit. *Proceedings of the COLING/ACL on Interactive presentation sessions*, (March), 69–72. [Cited on page 115.]

Black, A. W. & Campbell, N. (1995). Optimising selection of units from speech databases for concatenative synthesis. In *Proc. Eurospeech*, vol. 2, pp. 581–584. [Cited on page 63.]

Black, A. W. & Taylor, P. (1994). CHATR: a generic speech synthesis system. In *Proceedings of the 15th conference on Computational linguistics-Volume 2*, pp. 983–986. Association for Computational Linguistics. [Cited on page 62.]

Blashill, P. (2002). Six Machines That Changed the Music World. *Wired Magazine*, *10*. [Cited on page 9.]

Böck, S., Korzeniowski, F., Schlüter, J., Krebs, F., & Widmer, G. (2016). madmom: a new Python Audio and Music Signal Processing Library. [Cited on pages 76 and 97.]

Böck, S., Krebs, F., & Schedl, M. (2012). Evaluating the Online Capabilities of Onset Detection Methods. *ISMIR*, pp. 1–6. [Cited on page 75.]

Böck, S. & Widmer, G. (2013). Maximum filter vibrato suppression for onset detection. *Proc. of the 16th Int. Conference on Digital Audio Effects*, pp. 1–7. [Cited on pages 75, 76, and 176.]

Boden, M. A. (1998). Creativity and artificial intelligence. *Artificial Intelligence*, *103*(1), 347–356. [Cited on page 17.]

Boden, M. A. (2009). Computer models of creativity. *AI Magazine*, *30*(3), 23. [Cited on page 17.]

Boden, M. A. & Edmonds, E. A. (2009). What is generative art? *Digital Creativity*, *20*(1-2), 21–46. [Cited on page 17.]

Bogdanov, D., Wack, N., Gómez, E., Gulati, S., Herrera, P., Mayor, O., Roma, G., Salamon, J., Zapata, J., & Serra, X. (2013). ESSENTIA: an Audio Analysis Library for Music Information Retrieval. *International Society for Music Information Retrieval Conference (ISMIR 2013)*, pp. 493–498. [Cited on pages 76, 97, and 142.]

Boone, H. N. & Boone, D. A. (2012). Analyzing Likert Data Likert-Type Versus Likert Scales. *Journal of Extension*, *50*(2). [Cited on page 43.]

Bradshaw, M. & Xenakis, I. (1973). Formalized Music: Thought and Mathematics in Composition. [Cited on pages 18 and 57.]

Bradski, G. (2000). The OpenCV Library. *Dr Dobbs Journal of Software Tools*, *25*, 120–125. [Cited on page 142.]

Bregman, A. S. (1994). *Auditory scene analysis: The perceptual organization of sound*. MIT press. [Cited on pages 84 and 89.]

Brent, W. (2009a). Cepstral analysis tools for percussive timbre identification. *Proceedings of the 3rd International Pure Data Convention*. [Cited on page 97.]

Brent, W. (2009b). Perceptually based pitch scales in cepstral techniques percussive timbre identification. In *Proceedings of the International Computer Conference*, pp. 3–6. Montreal, Canada. [Cited on page 97.]

Brent, W. (2010). A timbre analysis and classification toolkit for Pure Data. In *International Computer Music Conference*, pp. 2–7. [Cited on pages 64, 97, and 132.]

Brooks, F., Hopkins, A., Neumann, P., & Wright, W. (1993). *An experiment in musical composition*. Cambridge: MIT. [Cited on page 21.]

Brossier, P. (2006). *Automatic annotation of musical audio for interactive applications*. Ph.D. thesis, Queen Mary, University of London. [Cited on pages 81, 97, and 102.]

Brossier, P., Bello, J. P., & Plumbley, M. D. (2004). Real-time temporal segmentation of note objects in music signals. *Proceedings of the 2004 International Computer Music Conference*. [Cited on page 88.]

Brown, D. G. & Golod, D. (2010). Decoding HMMs using the k best paths: algorithms and applications. *BMC bioinformatics*, *11 Suppl 1*, S28. [Cited on page 121.]

Bullock, J. (2007). LibXtract: A Lightweight Library for Audio Feature Extraction. *Proc. International Computer Music Conference*, pp. 3–6. [Cited on page 97.]

Burns & Helen, K. (1994). *The history and development of algorithms in music composition, 1957–1993*. Ball State University. [Cited on page 3.]

Butler, M. J. M. J. (2006). *Unlocking the groove : rhythm, meter, and musical design in electronic dance music*. Indiana University Press. [Cited on pages 2 and 9.]

Cai, W., Li, Q., & Guan, X. (2011). Automatic singer identification based on auditory features. *Proceedings - 2011 7th International Conference on Natural Computation, ICNC 2011*, *3*, 1624–1628. [Cited on page 100.]

Campbell, P. S. (1997). Music, the universal language: fact or fallacy? *International Journal of Music Education*, *os-29*(1), 32–39. [Cited on pages 1 and 22.]

Cardle, M., Brooks, S., & Robinson, P. (2003). Audio and User Directed Sound Synthesis. *Proceedings of the International Computer Music Conference (ICMC)*. [Cited on pages 67, 70, and 151.]

Cardoso, A., Veale, T., & Wiggins, G. (2009). Converging on the Divergent: The History (and Future) of the International Joint Workshops in Computational Creativity. *AI Magazine*, *30*(3), 15–22. [Cited on page 17.]

Carpentier, G. & Bresson, J. (2010). Interacting with symbol, sound, and feature spaces in Orchidée, a computer-aided orchestration environment. *Computer Music Journal*, *34*(1), 10–27. [Cited on page 84.]

Cascone, K. (2000). The aesthetics of failure:"Post-digital" tendencies in contemporary computer music. *Computer Music Journal*, *24*(4), 12–18. [Cited on page 57.]

Casey, M., Veltkamp, R., Goto, M., Leman, M., Rhodes, C., & Slaney, M. (2008). Content-Based Music Information Retrieval: Current Directions and Future Challenges. *Proceedings of the IEEE*, *96*. [Cited on page 55.]

Chang, C.-C. & Lin, C.-J. (2011). LIBSVM: a library for support vector machines. *ACM transactions on intelligent systems and technology (TIST)*, *2*(3), 27. [Cited on page 105.]

Chang, J. (2007). *Can't stop won't stop: A history of the hip-hop generation*. St. Martin's Press. [Cited on page 13.]

Chih-Wei Hsu, Chih-Chung Chang & Lin, C.-J. (2008). A Practical Guide to Support Vector Classification. *BJU international*, *101*(1), 1396–400. [Cited on page 105.]

Cho, T., Weiss, R. J., & Bello, J. P. (2010). Exploring Common Variations in State of the Art Chord Recognition Systems. *Sound and Music Computing*, *1*(January), 11–22. [Cited on page 116.]

Chomsky, N. (1957). *Syntactic Structures*, vol. 7. [Cited on page 22.]

Chudy, M. & Dixon, S. (2010). Towards Music Performer Recognition Using Timbre Features. In *International Conference of Students of Systematic Musicology*. Cambridge, UK. [Cited on page 100.]

Colannino, J. & Toussaint, G. (2005). An algorithm for computing the restriction scaffold assignment problem in computational biology. *Information Processing Letters*, *95*, 466–471. [Cited on pages 33 and 40.]

Coleman, G. (2015). *Descriptor Control of Sound Transformations and Mosaicing Synthesis*. Ph.D. thesis, Universitat Pompeu Fabra. [Cited on pages 84, 146, and 152.]

Coleman, G., Maestre, E., & Bonada, J. (2010). Augmenting Sound Mosaicing with Descriptor-Driven Transformation. *Proc. Digital Audio Effects (DAFx-10)*, pp. 1–4. [Cited on page 84.]

Collins, N. (2001). Algorithmic composition methods for breakbeat science. *Proceedings of Music Without Walls*. [Cited on page 59.]

Collins, N. (2002). Interactive evolution of breakbeat cut sequences. In *Proceedings of Cybersonica, Institute of Contemporary Arts, London*, Dahlstedt 2001. [Cited on page 59.]

Collins, N. (2006a). Towards a style-specific basis for computational beat tracking. [Cited on page 153.]

Collins, N. (2006b). *Towards autonomous agents for live computer music: Real-time machine listening and interactive music systems*. Ph.D. thesis, University of Cambridge. [Cited on page 59.]

Collins, N. (2007a). Audiovisual concatenative synthesis. *Proceedings of the International Computer Conference*, pp. 389–392. [Cited on page 110.]

Collins, N. (2008). The Analysis of Generative Music Programs. *Organised Sound*, *13*(03), 237. [Cited on page 3.]

Collins, N. (2010). *Introduction to computer music*. John Wiley & Sons. [Cited on page 89.]

Collins, N. & D'Escrivan, J. (2017). *The Cambridge companion to electronic music*. Cambridge University Press. [Cited on pages 21, 27, 56, and 57.]

Collins, N., Schedel, M., & Wilson, S. (2013). *Electronic Music*. Cambridge Introductions to Music. Cambridge University Press. [Cited on page 11.]

Collins, S. (2007b). Amen to that. *M/C Journal*, *10*(2). [Cited on page 14.]

Colton, S. & Wiggins, G. A. (2012). Computational creativity: The final frontier? *Frontiers in Artificial Intelligence and Applications*, *242*, 21–26. [Cited on page 17.]

Conkie, A. (1999). A robust unit selection system for speech synthesis. *The Journal of the Acoustical Society of America*, *105*(2), 978. [Cited on page 62.]

Cooper, M., Foote, J., Pampalk, E., & Tzanetakis, G. (2006a). Visualization in audio-based music information retrieval. *Computer Music Journal*, *30*(2), 42–62. [Cited on page 131.]

Cooper, M., Foote, J., Pampalk, E., & Tzanetakis, G. (2006b). Visualization in Audio-Based Music Information Retrieval. *Computer Music Journal*, *30*(2), 42–62. [Cited on page 134.]

Cooprider, N. D. & Burton, R. P. (2007). Extension of Star Coordinates into three dimensions. *Electronic Imaging 2007*, *6495*(801), 64950Q–64950Q. [Cited on page 132.]

Cope, D. (1987). Experiments in Music Intelligence (EMI). In *ICMC Proceedings*, pp. 174–181. [Cited on pages 17 and 23.]

Cope, D. (1991). *Computers and Musical Style*. A-R Editions. [Cited on pages 3, 17, and 23.]

Cope, D. (2000). New directions in music. [Cited on page 18.]

Cope, D. (2009). *Hidden structure: music analysis using computers*. AR Editions, Inc. [Cited on page 23.]

Cox, C. & Warner, D. (2004). *Audio culture: Readings in modern music*. A&C Black. [Cited on page 60.]

Cutler, C. (1994). Plunderphonia. *MusicWorks*, *60*. [Cited on page 60.]

Dannenberg, R. B. (2006). Concatenative Synthesis Using Score-Aligned Transcriptions Music Analysis and Segmentation. *International Computer Music Conference*, pp. 352–355. [Cited on page 110.]

Davies, M., Stark, A., Gouyon, F., & Goto, M. (2014a). Improvasher: A Real-Time Mashup System for Live Musical Input. *Proceedings of the International Conference on New Interfaces for Musical Expression*, pp. 541–544. [Cited on page 61.]

Davies, M. E. P., Hamel, P., Yoshii, K., & Goto, M. (2013). AutoMashUpper: An Automatic Multi-Song Mashup System. *Proceedings of the 14th International Society for Music Information Retrieval Conference, ISMIR 2013*, pp. 575—-580. [Cited on pages 61 and 124.]

Davies, M. E. P., Hamel, P., Yoshii, K., & Goto, M. (2014b). AutoMashUpper: Automatic creation of multi-song music mashups. *IEEE/ACM Transactions on Audio, Speech and Language Processing (TASLP)*, *22*(12), 1726–1737. [Cited on page 61.]

de Cheveigné, A. & Kawahara, H. (2002). YIN, a fundamental frequency estimator for speech and music. *The Journal of the Acoustical Society of America*, *111*(4), 1917–1930. [Cited on page 101.]

Degara, N., Rua, E. A., Pena, A., Torres-Guijarro, S., Davies, M. E. P., & Plumbley, M. D. (2012). Reliability-informed beat tracking of musical signals. *IEEE Transactions on Audio, Speech and Language Processing*, *20*(1), 278–289. [Cited on page 81.]

Deltorn, J.-M. (2017). Deep creations: Intellectual Property and the Automata. *Frontiers in Digital Humanities*, *4*, 3. [Cited on page 3.]

Deng, J. (2008). A Study on Feature Analysis for Musical Instrument Classification. *IEEE Transactions on Systems, Man and Cybernetics*, *38*(2), 429–438. [Cited on page 106.]

Díaz-Báñez, J. M., Farigu, G., Gómez, F., Rappaport, D., & Toussaint, G. T. (2004). El compás flamenco: a phylogenetic analysis. In *Proceedings of BRIDGES: Mathematical Connections in Art, Music and Science*. Kansas. [Cited on pages 32 and 39.]

Dixon, S. (2006). Onset detection revisited. *Proceedings of the 9th International Conference on . . .*, pp. 133–137. [Cited on pages 75 and 81.]

Dixon, S. (2007). Evaluation of the audio beat tracking system BeatRoot. *Journal of New Music Research*, *36*(1), 39–50. [Cited on page 82.]

Dixon, S., Gouyon, F., & Widmer, G. (2004). Towards characterisation of music via rhythmic patterns. *Proceedings of the 5th International Conference on Music Information Retrieval (ISMIR 2004)*, *5*(April), 509–516. [Cited on page 33.]

Donnelly, P. J. & Sheppard, J. W. (2016). Cross-Dataset Validation of Feature Sets in Musical Instrument Classification. *Proceedings - 15th IEEE International Conference on Data Mining Workshop, ICDMW 2015*, pp. 94–101. [Cited on page 103.]

Duignan, M., Noble, J., Barr, P., & Biddle, R. (2004). Metaphors for electronic music production in Reason and Live. *Computer Human Interaction*, (January), 111–120. [Cited on page 130.]

Duignan, M., Noble, J., & Biddle, R. (2005). A taxonomy of sequencer user-interfaces. In *International Computer Music Conference*. [Cited on page 130.]

Duignan, M., Noble, J., & Biddle, R. (2010). Abstraction and Activity in Computer-Mediated Music Production. *Computer Music Journal*, *34*(4), 22–33. [Cited on page 130.]

Dupont, S., Ravet, T., Picard-Limpens, C., & Frisson, C. (2013). Nonlinear dimensionality reduction approaches applied to music and textural sounds. In *Proceedings - IEEE International Conference on Multimedia and Expo*. [Cited on page 135.]

Duxbury, C., Sandler, M., & Davies, M. (2002). A hybrid approach to musical note onset detection. *5th Int. Conference on Digital Audio Effects (DAFx-02), Hamburg, Germany*, (November 2002), 33–38. [Cited on page 73.]

Eigenfeldt, A. (2006). Kinetic Engine: Toward an Intelligent Improvising Instrument. *Proceedings of the Sound and Music Computing Conference*, pp. 97–100. [Cited on page 35.]

Eigenfeldt, A. (2009). The evolution of evolutionary software: intelligent rhythm generation in Kinetic Engine. *Applications of Evolutionary Computing*. [Cited on page 20.]

Eigenfeldt, A. (2016). Exploring Moment-form in Generative Music. *Proceedings of the Sound and Music Computing Conference*, (August). [Cited on page 21.]

Eigenfeldt, A., Bown, O., Brown, A. R., & Gifford, T. (2016). Flexible Generation of Musical Form : Beyond Mere Generation. In *International Conference on Computational Creativity*. [Cited on page 21.]

Eigenfeldt, A. & Pasquier, P. (2013). Evolving structures for electronic dance music. *Proceeding of the fifteenth annual conference ...*, p. 319. [Cited on pages 21 and 49.]

Einbond, A. & Schwarz, D. (2010). Spatializing Timbre With Corpus-Based Concatenative Synthesis. In *International Computer Music Conference*. New York, USA. [Cited on page 110.]

Elliott, T. M., Hamilton, L. S., & Theunissen, F. E. (2013). Acoustic structure of the five perceptual dimensions of timbre in orchestral instrument tones. *The Journal of the Acoustical Society of America*, *133*(1), 389–404. [Cited on page 89.]

Ellis, D. P. W. (2005). PLP and RASTA (and MFCC, and inversion) in Matlab. [Cited on pages XXV, XXV, and 98.]

Ellis, D. P. W. (2007). Beat Tracking by Dynamic Programming. *Journal of New Music Research*, *36*(1), 51–60. [Cited on page 81.]

Ellis, D. P. W. (2009). Gammatone-like spectrograms. [Cited on page 99.]

Essl, K. (1992). Real Time Composition Library (RTC-lib). [Cited on page 18.]

Eyben, F., Böck, S., Schuller, B., & Graves, A. (2010). Universal Onset Detection with Bidirectional Long-Short Term Memory Neural Networks. In *International Society for Music Information Retrieval Conference*, pp. 589–594. Utrecht, Netherlands. [Cited on page 74.]

Faraldo, Á., Gómez, E., Jordà, S., & Herrera, P. (2016). Key Estimation in Electronic Dance Music. In *38th European Conference on Information Retrieval*, pp. 335–347. Springer-Verlag, Padua, Italy: Springer-Verlag. [Cited on page 102.]

Faraldo, Á., Jordà, S., & Herrera, P. (2017a). A Multi-Profile Method for Key Estimation in EDM. In *AES International Conference on Semantic Audio*. Erlangen, Germany. [Cited on page 102.]

Faraldo, Á., Jordà, S., & Herrera, P. (2017b). A Study of Tonal Practises In Electronic Dance Music. In *Ninth European Music Analysis Conference*. Strasbourg, France. [Cited on page 102.]

Fathima, R. & Raseena, P. E. (2013). Gammatone Cepstral Coefficient for Speaker Identification. *International Journal of Advanced Research in Electrical, Electronics and Instrumentation Engineering*, *4*(10), 795–798. [Cited on page 99.]

Fels, S., Gadd, A., & Mulder, A. (2002). Mapping transparency through metaphor: towards more expressive musical instruments. *Organised Sound*, *7*(2), 109–126. [Cited on page 130.]

Fernández, J. D. & Vico, F. (2013). Ai methods in algorithmic composition: A comprehensive survey. *Journal of Artificial Intelligence Research*, *48*, 513–582. [Cited on pages 3, 17, 19, 20, and 117.]

Fitzgerald, D. (2004). Automatic drum transcription and source separation. *Dublin Institute of Technology*, p. 3. [Cited on page 85.]

Fletcher, H. & Munson, W. A. (1933). Loudness, Its Definition, Measurement and Calculation. *Journal of the Acoustical Society of America (JASA)*, *5*(2), 82. [Cited on page 86.]

Flexer, A. (2015). Improving visualization of high-dimensional music similarity spaces. *International Society for Music Information Retrieval Conference (ISMIR)*, pp. 547–553. [Cited on page 135.]

Font, F. & Bandiera, G. (2017). Freesound Explorer: Make Music While Discovering Freesound! *Web Audio Conference (WAC)*. [Cited on pages 135 and 148.]

Forman, M. & Neal, M. A. (2004). *That's the joint!: the hip hop studies reader*. Psychology Press. [Cited on page 58.]

Fox, J. & Carlile, J. (2005). SoniMime: movement sonification for real-time timbre shaping. *Proceedings of the 5th International Conference on New Interfaces for Musical Expression (NIME05)*, pp. 242–243. [Cited on page 100.]

Frane, A. V. (2017). Swing Rhythm in Classic Drum Breaks From Hip-Hop's Breakbeat Canon. *Music Perception: An Interdisciplinary Journal*, *34*(3), 291–302. [Cited on page 59.]

Frisson, C. (2015). *Designing interaction for browsing media collections (by similarity)*. Ph.D. thesis, Université de Mons. [Cited on pages 132 and 134.]

Frisson, C., Dupont, S., Yvart, W., Riche, N., Siebert, X., & Dutoit, T. (2014a). AudioMetro. *Proceedings of the 9th Audio Mostly on A Conference on Interaction With Sound - AM '14*, pp. 1–8. [Cited on page 135.]

Frisson, C., Picard, C., & Tardieu, D. (2010). Audiogarden : Towards a Usable Tool for Composite Audio Creation. *QPSR of the numediart research program*, *3*(2), 33–36. [Cited on pages 64, 66, 70, 79, and 140.]

Frisson, C., Yvart, W., Riche, N., Siebert, X., & Dutoit, T. (2014b). a Proximity Grid Optimization Method To Improve Audio Search for Sound Design. *15th International Society for Music Information Retrieval Conference ( ISMIR 2014 )*, (Ismir), 349–354. [Cited on page 135.]

Fujishima, T. (1999). Realtime Chord Recognition of Musical Sound: A System Using Common Lisp Music. [Cited on page 102.]

Gabrielsson, A. (1973a). Similarity ratings and dimension analyses of auditory rhythm patterns. 1. *Scandinavian Journal of Psychology*, *14*(1), 138–160. [Cited on page 134.]

Gabrielsson, A. (1973b). Similarity ratings and dimension analyses of auditory rhythm patterns. 2. *Scandinavian Journal of Psychology*, *14*(1), 161–176. [Cited on page 134.]

Garshol, L. M. (2003). BNF and EBNF: What are they and how do they work. *acedida pela última vez em*, *16*. [Cited on page 22.]

Gartland-Jones, A. & Copley, P. (2003). The suitability of genetic algorithms for musical composition. *Contemporary Music Review*, *22*(3), 43–55. [Cited on page 26.]

Gerhard, D. (2003). *Pitch extraction and fundamental frequency: History and current techniques*. [Cited on page 101.]

Ghedini, F., Pachet, F., & Roy, P. (2015). Creating music and texts with flow machines. *Multidisciplinary Contributions to the Science of Creative Thinking*, pp. 325–343. [Cited on page 3.]

Gilbert, J. (1997). Soundtrack for an uncivil society: rave culture, the criminal justice act and the politics of modernity. *New Formations*, pp. 5–22. [Cited on page 10.]

Gillet, O. & Gaël, R. (2006). Enst-drums: an extensive audio-visual database for drum signals processing. *In Proceedings of the 7th International Symposium on Music Information Retrieval (ISMIR 2006*, pp. 156–159. [Cited on pages 76 and 176.]

Glover, J., Lazzarini, V., & Timoney, J. (2011). Real-time detection of musical onsets with linear prediction and sinusoidal modeling. *EURASIP Journal on Advances in ...*, *2011*(1), 68. [Cited on pages 76 and 176.]

Goldberg, D. E. (1989). *Genetic algorithms in search optimization and machine learning*. Addison-Wesley Publishing Company. [Cited on page 24.]

Gómez, E. (2006). Tonal Description of Polyphonic Audio for Music Content Processing. *INFORMS Journal on Computing*, *18*(3), 294–304. [Cited on page 102.]

Gómez, E. & Herrera, P. (2004). Estimating The Tonality Of Polyphonic Audio Files: Cognitive Versus Machine Learning Modelling Strategies. *Ismir*, pp. 1–4. [Cited on page 102.]

Gómez, F., Melvin, A., Rappaport, D., & Toussaint, G. T. (2005). Mathematical measures of syncopation. *BRIDGES: Mathematical Connections in Art, Music and Science*. [Cited on pages 49 and 50.]

Gómez, F., Thul, E., & Toussaint, G. (2007). An experimental comparison of formal measures of rhythmic syncopation. *Proceedings of the International Computer Music Conference*, (January 2007), 101–104. [Cited on page 51.]

Gómez-Marín, D., Jordà, S., & Herrera, P. (2015). Pad and Sad: Two Awareness-Weighted Rhythmic Similarity Distances. *16th International Society for Music Information Retrieval Conference*. [Cited on page 53.]

Gómez-Marín, D., Jordà, S., & Herrera, P. (2016). Rhythm Spaces. *Proceeedings of the 4th International Workshop on Musical Metacreation*. [Cited on pages 20 and 134.]

Gómez-Marín, D., Jordà, S., & Herrera, P. (2017). Drum rhythm spaces : from global models to style-specifc maps. In *13th International Symposium on Computer Music Multidisciplinary Research*. Matosinhos, Portugal. [Cited on pages 49 and 134.]

Gore, G. (1997). The beat goes on: trance, dance and tribalism in rave culture. In *Dance in the City*, pp. 50–67. Springer. [Cited on page 10.]

Goto, M. (2001). An Audio-based Real-time Beat Tracking System for Music With or Without Drum-sounds. *Journal of New Music Research*, *30*(2), 159–171. [Cited on page 82.]

Greenberg, C. (1971). Collage. In *Art and Culture: Critical Essays*. Beacon Press. [Cited on page 55.]

Grekow, J. (2017). Audio features dedicated to the detection of arousal and valence in music recordings. *2017 IEEE International Conference on INnovations in Intelligent SysTems and Applications (INISTA)*, pp. 40–44. [Cited on page 100.]

Grill, T. (2004). C++ layer for Pure Data & Max/MSP externals. In *2nd International Linux Audio Conference*. [Cited on page 36.]

Grill, T. (2012). Constructing high-level perceptual audio descriptors for textural sounds. In *9th Sound and Music Computing Conference 2012*, p. 235. [Cited on page 135.]

Grzywczak, D. & Gwardys, G. (2014). Audio Features in Music Information Retrieval. *Active Media Technology*, *60*(4), 187–199. [Cited on page 74.]

Guastavino, C., Toussaint, G., Gómez, F., Marandola, F., & Absar, R. (2008). Rhythmic similarity in Flamenco music: Comparing psychological and mathematical measures. *Proceedings of the fourth Conference on Interdisciplinary Musicology*, p. 76. [Cited on page 34.]

Guéguen, L. (2005). Sarment: Python modules for HMM analysis and partitioning of sequences. *Bioinformatics*, *21*(16), 3427–3428. [Cited on page 115.]

Gunderson, P. A. (2004). Danger Mouse's Grey Album, mash-ups, and the age of composition. *Postmodern culture*, *15*(1). [Cited on page 61.]

Gurney, K. (1996). *Introduction to neural networks*, vol. 6. CRC press. [Cited on page 106.]

Gustafson, K. (1987). A new method for displaying speech rhythm, with illustrations from some Nordic languages. *Nordic Prosody IV*, pp. 105–114. [Cited on page 34.]

Hackbarth, B. (2011). Audioguide : A Framework for Creative Exploration of Concatenative Sound Synthesis. *IRCAM Research Report*. [Cited on pages 67, 70, 84, 85, and 151.]

Hackbarth, B., Schnell, N., Esling, P., & Schwarz, D. (2013). Composing morphology: Concatenative synthesis as an intuitive medium for prescribing sound in time. *Contemporary Music Review*, *32*(1), 49–59. [Cited on page 84.]

Hackeling, G. (2014). *Mastering Machine Learning with scikit-learn*. Packt Publishing Ltd. [Cited on page 134.]

Hagberg, A. A., Schult, D. A., & Swart, P. J. (2008). Exploring Network Structure, Dynamics, and Function using NetworkX. In G. Varoquaux, T. Vaught, & J. Millman (Eds.) *Proceedings of the 7th Python in Science Conference*, pp. 11–15. Pasadena, CA USA. [Cited on page 120.]

Hamel, P. & Eck, D. (2010). Learning Features from Music Audio with Deep Belief Networks. *International Society for Music Information Retrieval Conference (ISMIR)*, (Ismir), 339–344. [Cited on page 135.]

Harvey, J. (2000). Spectralism. *Contemporary music review*, *19*(3), 11–14. [Cited on page 90.]

Hatch, D. & Millward, S. (1987). *From blues to rock: an analytical history of pop music*. Manchester University Press. [Cited on page 2.]

Haworth, C. (2015). Participation Over Belonging : Analysing Microsound Using Digital Methods Introduction. *The Electroacoustic Music Studies Network Conference,*, (June), 1–10. [Cited on page 14.]

Helander, M. G. (2014). *Handbook of human-computer interaction*. Elsevier. [Cited on page 130.]

Herman, A. & Sloop, J. M. (1998). The politics of authenticity in postmodern rock culture: The case of Negativland and \textit{the letter U and the numeral 2}. *Critical Studies in Media Communication*, *15*(1), 1–20. [Cited on page 61.]

Herrera, P., Dehamel, A., & Gouyon, F. (2003). Automatic labeling of unpitched percussion sounds. In *Audio Engineering Society 114th Convention*. [Cited on pages 98, 105, and 106.]

Herrera, P., Yeterian, A., & Gouyon, F. (2002). Automatic classification of drum sounds: a comparison of feature selection methods and classification techniques. *International Conference on Music and Artificial Intelligence (ICMAI)*. [Cited on pages 105 and 106.]

Herrera-Boyer, P., Peeters, G., & Dubnov, S. (2003). Automatic Classification of Musical Instrument Sounds. *Journal of New Music Research*, *32*(1), 3–21. [Cited on pages 88, 105, and 106.]

Hiller, L. A. & Isaacson, L. M. (1979). *Experimental Music; Composition with an electronic computer*. Greenwood Publishing Group Inc. [Cited on page 18.]

Hiraga, R., Bresin, R., Hirata, K., & Katayose, H. (2004). Rencon 2004: Turing Test for Musical Expression. *Proceedings of the International Conference on New Interfaces for Musical Expression*, pp. 120–123. [Cited on page 151.]

Hoashi, K., Hamawaki, S., Ishizaki, H., Takishima, Y., Katto, J., Kddi, R., & Keiichiro Hoashi Shuhei Hamawaki, H. I. Y. T. J. K. (2009). Usability evaluation of visualization interfaces for content-based music retrieval systems. *International Society for Music Information Retrieval Conference (ISMIR'09)*, (Ismir), 207–212. [Cited on page 131.]

Hockman, J. (2007). Automatic Timbre Mutation of Drum Loops. [Cited on page 59.]

Hockman, J., Davies, M., & Fujinaga, I. (2012). One in the Jungle: Downbeat Detection in Hardcore, Jungle, and Drum and Bass. *ISMIR*, (Ismir), 169–174. [Cited on page 59.]

Hockman, J. A. & Davies, M. E. P. (2015). Computational Strategies for Breakbeat Classification and Resequencing in Hardcore, Jungle and Drum & Bass. In *Proc. of the 18th Int. Conference on Digital Audio Effects (DAFx-15)*, pp. 1–6. [Cited on pages 59 and 153.]

Hoffman, M., Cook, P. R., & Blei, D. M. (2009). Bayesian Spectral Matching: Turning Your MC Into MC Hammer via MCMC Sampling. *Proceedings of the 2009 International Computer Music Conference (ICMC)*, pp. 16–21. [Cited on page 84.]

Holland, J. H. (1975). Adaptation in natural and artificial systems. [Cited on page 24.]

Holm-Hudson, K. (1997). Quotation and Context: Sampling and John Oswald's Plunderphonics. *Leonardo Music Journal*, *7*(May), 17–25. [Cited on page 60.]

Holmes, T. (2008). *Electronic and Experimental Music*. [Cited on page 56.]

Hook, P. (2009). *The Hacienda: how not to run a club*. Simon and Schuster. [Cited on page 10.]

Horowitz, D. (2004). Generating Rhythms with Genetic Algorithms. In *The Twelfth National Conference on Artificial Intelligence*. [Cited on page 35.]

Hoskinson, R. & Pai, D. (2001). Manipulation and Resynthesis with Natural Grains. *Proceedings of the International Computer Music Conference (ICMC)*, pp. 338–341. [Cited on pages 67 and 70.]

Hoskinson, R. & Pai, D. K. (2007). Synthetic Soundscapes with Natural Grains. *Presence: Teleoperators & Virtual Environments*, *16*(1), 84–99. [Cited on page 67.]

Hove, M. J., Marie, C., Bruce, I. C., & Trainor, L. J. (2014). Superior time perception for lower musical pitch explains why bass-ranged instruments lay down musical rhythms. *Proceedings of the National Academy of Sciences*, *111*(28), 10383–10388. [Cited on page 49.]

Hsu, W. & Sosnick, M. (2009). Evaluating interactive music systems: An HCI approach. In *Proceedings of New Interfaces for Musical Expression*, pp. 25–28. [Cited on page 151.]

Hulshof, C., Siebert, X., & Melot, H. (2016). NeoMI : a New Environment for the Organization of Musical Instruments. In *International Workshop on Folk Music Analysis*. Dublin, Ireland. [Cited on page 103.]

Humphrey, E. J., Turnbull, D., & Collins, T. (2013). A brief review of creative MIR. *International Society for Music Information Retrieval*. [Cited on page 61.]

Hunt, A. J. & Black, A. W. (1996). Unit selection in a concatenative speech synthesis system using a large speech database. *1996 IEEE International Conference on Acoustics, Speech, and Signal Processing Conference Proceedings*, *1*, 373–376. [Cited on pages 62, 63, 111, and 112.]

ISO 226:2003 (2003). Acoustics—Normal Equal-Loudness Level Contours. *International Organization for Standardization; Geneva, Switzerland*. [Cited on page 86.]

ITU-R BS.1770-3 (2013). Algorithms to measure audio programme loudness and true-peak audio level. *International Telecommunications Union, Geneva Switzerland*, *3*, 1770–4. [Cited on page 87.]

Jacob, B. L. (1996). Algorithmic composition as a model of creativity. *Organised Sound*, *1*(3), S1355771896000222. [Cited on page 17.]

Jade, C. (2016). *Rhythm alternation using interval sets*. Ph.D. thesis, Universitat Pompeu Fabra. [Cited on page 171.]

Jathal, K. (2017). Real-Time Timbre Classification for Tabletop Hand Drumming. *Computer Music Journal*, *41*(2), 38–51. [Cited on page 97.]

Jehan, T. (2005). Creating music by listening. *Media Arts and Sciences*, *PhD*. [Cited on pages 33, 64, 70, 81, and 86.]

Johnson-Roberson, C. & Sudderth, E. (2017). Content-Based Genre Classification and Sample Recognition Using Topic Models. p. 8. [Cited on page 100.]

Jones, E., Oliphant, T., & Peterson, P. (2014). SciPy: Open Source Scientific Tools for Python. [Cited on page 111.]

Jorda, S. (1991). A Real-Time Midi Composer And Interactive Improviser By Means Of Feedback Systems. *International Computer Music Conference*, (January 1991). [Cited on page 21.]

Jordà, S., Gómez-Marín, D., Faraldo, Á., & Herrera, P. (2016). Drumming with style: From user needs to a working prototype. *Proceedings of the International Conference on New Interfaces for Musical Expression*, *16*, 365–370. [Cited on pages 20, 51, and 53.]

Jordà, S., Kaltenbrunner, M., Geiger, G., & Bencina, R. (2005). The reactable*. In *International Computer Music Conference*, pp. 2–5. Barcelona. [Cited on pages 18 and 131.]

Kaehler, A. & Bradski, G. (2016). *Learning OpenCV 3: Computer Vision in C++ with the OpenCV Library*. " O'Reilly Media, Inc.". [Cited on page 111.]

Kaliakatsos-Papakostas, M. a., Floros, A., & Vrahatis, M. N. (2013). evoDrummer: Deriving rhythmic patterns through interactive genetic algorithms. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 7834 LNCS, pp. 25–36. [Cited on page 35.]

Keith, M. (1991). From polychords to polya: adventures in musical combinatorics. [Cited on page 50.]

Kiefer, C., Collins, N., & Fitzpatrick, G. (2008). HCI Methodology For Evaluating Musical Controllers: A Case Study. *Proceedings of the 2008 International Conference on New Interfaces for Musical Expression (NIME-08)*, pp. 87–90. [Cited on page 151.]

Kim, H. G., Moreau, N., & Sikora, T. (2006a). *MPEG-7 audio and beyond: Audio content indexing and retrieval*. John Wiley & Sons. [Cited on pages 88, 93, and 94.]

Kim, Y. E., Williamson, D. S., & Pilli, S. (2006b). Towards Quantifying the" Album Effect" in Artist Identification. In *ISMIR*, pp. 393–394. [Cited on page 147.]

Kim-Boyle, D. (2006). Spectral and Granular Spatialization with Boids. *Proceedings of the 2006 International Computer Music Conference*, (Reynolds 1987), 139–142. [Cited on page 139.]

Klügel, N., Becker, T., & Groh, G. (2014). Designing Sound Collaboratively - Perceptually Motivated Audio Synthesis. In *New Interfaces for Musical Expression*, pp. 327–330. London, UK. [Cited on page 110.]

Knees, P., Andersen, K., Jordá, S., Hlatky, M., Bucci, A., Gaebele, W., & Kaurson, R. (2016). The GiantSteps Project: A Second-Year Intermediate Report. In *International Computer Music Conference*, pp. 363–365. [Cited on page 4.]

Knees, P., Faraldo, Á., Herrera, P., Vogl, R., Böck, S., Florian, H., & Le Goff, M. (2015). Two Data Sets for Tempo Estimation and Key Detection in Electronic Dance Music Annotated from User Corrections. In *Proceedings of the 16th International Society for Music Information Retrieval Conference (ISMIR)*. Malaga, Spain. [Cited on page 158.]

Kobayashi, R. (2003). Sound Clustering Synthesis Using Spectral Data. In *International Computer Music Conference*, vol. 0, pp. 1–3. [Cited on pages 70 and 73.]

Koetting, J. (1970). Analysis and notation of West African drum ensemble music. *Selected reports in ethnomusicology*, *1*(3), 115–146. [Cited on page 28.]

Korzeniowski, F., Sebastian, B., & Widmer, G. (2014). Probabilistic Extraction of Beat positions from a Beat Activation Function. *Proceedings of the 15th International Society for Music Information Retrieval Conference (ISMIR 2014)*, (Ismir), 513–518. [Cited on page 81.]

Koza, J. R. (1992). *Genetic programming: on the programming of computers by means of natural selection*, vol. 1. MIT press. [Cited on page 24.]

Krey, S. & Ligges, U. (2010). SVM based instrument and timbre classification. In *Classification as a Tool for Research*, pp. 759–766. Springer. [Cited on page 106.]

Krumhansl, C. L. (1989). Why is musical timbre so hard to understand ? [Cited on page 89.]

Lacoste, A. & Eck, D. (2007). A supervised classification algorithm for note onset detection. *Eurasip Journal on Advances in Signal Processing*, *2007*. [Cited on page 74.]

Lakkavalli, V. R., Arulmozhi, P., & Ramakrishnan, A. G. (2010). Continuity metric for unit selection based text-to-speech synthesis. *2010 International Conference on Signal Processing and Communications, SPCOM 2010*. [Cited on page 62.]

Lantz, B. (2013). Equidistance of Likert-Type Scales and Validation of Inferential Methods Using Experiments and Simulations. *Electronic Journal of Business Research Methods*, *11*(1), 16–28. [Cited on page 43.]

Laroche, J. (2003). Efficient Tempo and Beat Tracking in Audio Recordings. *Journal of the Audio Engineering Society*, *51*(4), 226–233. [Cited on page 73.]

Lartillot, O., Toiviainen, P., & Eerola, T. (2008). A matlab toolbox for music information retrieval. *Data analysis, machine learning and applications*, pp. 261–268. [Cited on page 97.]

Latartara, J. (2010). Laptop Composition at the Turn of the Millennium: Repetition and Noise in the Music of Oval, Merzbow, and Kid606. *Twentieth-Century Music*, *7*(01), 91–115. [Cited on page 57.]

Lazier, A. & Cook, P. R. (2003). Mosievius: Feature Driven Interactive Audio Mosaicing. In *Proc. of the 6th Int. Conference on Digital Audio Effects (DAFx-03)*, pp. 1–6. [Cited on page 70.]

Lee, C.-l., Lin, Y.-T., Yao, Z.-R., Lee, F.-Y., & Wu, J.-L. (2015). Automatic Mashup Creation By Considering Both Vertical And Horizontal Mashabilities. *Proc. International Society for Music Information Retrieval Conference (ISMIR)*, pp. 399–405. [Cited on page 61.]

Leimeister, M., Gaertner, D., & Dittmar, C. (2014). Rhythmic Classification of Electronic Dance Music. In *Audio Engineering Society Conference: 53rd International Conference: Semantic Audio*. [Cited on page 7.]

Lerch, A. (2012). *An introduction to audio content analysis: Applications in signal processing and music informatics.* [Cited on pages 74, 86, and 87.]

Lerdahl, F. & Jackendoff, R. (1985). *A generative theory of tonal music*. MIT press. [Cited on page 22.]

Levitin, D. J. (2007). *This is Your Brain on Music: The Science of a Human Obsession.* New York: Dutton/Penguin Books. [Cited on page 1.]

Lindemann, E. (2007). Music synthesis with reconstructive phrase modeling. *IEEE Signal Processing Magazine*, *24*(2), 80–91. [Cited on page 70.]

Liu, J. & Xie, L. (2010). SVM-based automatic classification of musical instruments. *2010 International Conference on Intelligent Computation Technology and Automation, ICICTA 2010*, *3*, 669–673. [Cited on page 106.]

Lochhead, J. I. & Auner, J. H. (2002). *Postmodern music/postmodern thought*. [Cited on page 56.]

Logan, B. (2000). Mel Frequency Cepstral Coefficients for Music Modeling. *International Symposium on Music Information Retrieval*, *28*, 11p. [Cited on pages 94 and 95.]

Lyons, J. (2015). Mel Frequency Cepstral Coefficient (MFCC) tutorial [Practical Cryptography]. [Cited on page 94.]

Lysloff, R. T. A. & Gay Jr, L. C. (2003). *Music and technoculture*. Wesleyan University Press. [Cited on page 61.]

Maestre, E., Hazan, A., Ramirez, R., & Perez, A. (2006). Using concatenative synthesis for expressive performance in jazz saxophone. *Proceedings of the International Computer Music Conference*, *2006*, 163–166. [Cited on page 110.]

Malmberg, V. (2010). *Iris: A Circular Polyrhythmic Music Sequencer*. Ph.D. thesis, Aalto University. [Cited on page 27.]

Mandel, M. I. & Ellis, D. (2005a). Song-Level Features and Support Vector Machines for Music Classification. In *ISMIR*, vol. 2005, pp. 594–599. [Cited on page 147.]

Mandel, M. I. & Ellis, D. P. W. (2005b). Song-level features and support vector machines for music classification. *Proceedings of the 6th International Symposium in Music Information Retrieval (ISMIR'05)*, (2004), 594–599. [Cited on page 105.]

Mangani, M., Baldizzone, R., & Nobile, G. (2006). Quotation in jazz improvisation: A database and some examples. In *Proceedings of the 9th International Conference on Music Perception and Cognition*, p. 286. [Cited on page 57.]

Manzo, V. J. & Kuhn, W. (2015). *Interactive composition: Strategies using Ableton live and max for live*. Oxford University Press, USA. [Cited on page 36.]

Marolt, M., Kavcic, A., & Privosnik, M. (2002). Neural networks for note onset detection in piano music. *Proceedings of ICMC 2002*, pp. 2–5. [Cited on page 74.]

Marti, U. (2015). *Expression Control of Singing Voice Synthesis: Modeling Pitch and Dynamics with Unit Selection and Statistical Approaches*. Ph.D. thesis, Universitat Pompeu Fabra. [Cited on page 171.]

Martin, D. (1999). Power play and party politics: The significance of raving. *The Journal of Popular Culture*, *32*(4), 77–99. [Cited on page 10.]

Martín, S. O. (2017). *Knowledge Extraction and Representation Learning for Music Recommendation and Classification*. Ph.D. thesis, Universitat Pompeu Fabra. [Cited on page 135.]

Masood, S., Gupta, S., & Khan, S. (2015). Novel approach for musical instrument identification using neural network. *2015 Annual IEEE India Conference (INDICON)*, pp. 1–5. [Cited on page 104.]

Masri, P. & Bateman, A. (1996). Improved modelling of attack transients in music analysis-resynthesis. *Proceedings of the International Computer Music Conference*, pp. 100–103. [Cited on page 142.]

McAdams, S., Beauchamp, J. W., & Meneguzzi, S. (1999). Discrimination of musical instrument sounds resynthesized with simplified spectrotemporal parameters. *The Journal of the Acoustical Society of America*, *105*(2 Pt 1), 882–897. [Cited on page 88.]

McAdams, S. & Bregman, A. S. (1979). Hearing Musical Streams. *Computer Music Journal*, *3*(4), 26–43. [Cited on page 89.]

McAdams, S., Winsberg, S., Donnadieu, S., De Soete, G., & Krimphoff, J. (1995). Perceptual scaling of synthesized musical timbres: Common dimensions, specificities, and latent subject classes. *Psychological Research*, *58*(3), 177–192. [Cited on page 88.]

McCormack, J. (2013). Aesthetics, art, evolution. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 7834 LNCS, pp. 1–12. [Cited on page 25.]

Mcfee, B., Raffel, C., Liang, D., Ellis, D. P. W., Mcvicar, M., Battenberg, E., & Nieto, O. (2015). librosa: Audio and Music Signal Analysis in Python. *PROC. OF THE 14th PYTHON IN SCIENCE CONF*, (Scipy), 1–7. [Cited on page 97.]

McGranahan, L. (2010). Bastards and booties: Production, copyright, and the mashup community. *Revista Transcultural de Música (TRANS)*, (14). [Cited on page 61.]

McKinney, M. F., Moelants, D., Davies, M. E., & Klapuri, A. (2007). Evaluation of audio beat tracking and music tempo extraction algorithms. *Journal of New Music Research*, *36*(1), 1–16. [Cited on page 81.]

McLeod, K. (2001). Genres, Subgenres, Sub-Subgenres and More: Musical and Social Differentiation Within Electronic/Dance Music Communities. *Journal of Popular Music Studies*, pp. 59–75. [Cited on pages 2 and 10.]

Merchant, H., Grahn, J., Trainor, L., Rohrmeier, M., & Fitch, W. T. (2015). Finding the beat: a neural perspective across humans and non-human primates. *Philosophical Transactions of the Royal Society B: Biological Sciences*, *370*(1664), 20140093–20140093. [Cited on page 48.]

Meroño-Peñuela, A., Meerwaldt, R., & Schlobach, S. (2017). SPARQL-DJ: The MIDI linked data mashup mixer for your next semantic party. *CEUR Workshop Proceedings*, *1963*, 1–4. [Cited on page 61.]

Metzer, D. (2003). *Quotation and cultural meaning in twentieth-century music*. [Cited on page 57.]

Middleton, R. (2009). 'Play It Again Sam': Some Notes on the Productivity of Repetition in Popular Music. *Popular Music*, *3*(1983), 235–270. [Cited on page 2.]

Miller, J. & Hammond, T. (2010). Wiiolin : a virtual instrument using the Wii remote. *Proceedings of the 2010 Conference on New Interfaces for Musical Expression (NIME 2010)*, (June), 15–18. [Cited on page 103.]

Milne, A. J. & Dean, R. T. (2015). Computational Creation and Morphing of Multilevel Rhythms by Control of Evenness. *Computer Music journal*, *40*(1), 35–53. [Cited on page 28.]

Milne, A. J., Herff, S. A., Bulger, D., Sethares, W. A., & Dean, R. T. (2016). XronoMorph: Algorithmic Generation of Perfectly Balanced and Well-Formed Rhythms. *Proceedings of the International Conference on New Interfaces for Musical Expression*, *16*, 388–393. [Cited on pages 28 and 30.]

Miranda, E. R. (1995). Granular Synthesis of Sounds by Means of a Cellular Automaton. *Leonardo*, *28*(4), 297–300. [Cited on page 57.]

Miranda, E. R. & Biles, A. J. (2007). *Evolutionary Computer Music*. Springer-Verlag New York, Inc. [Cited on pages 25 and 26.]

Miron, M. (2017). Monaural score-informed source separation for classical music using convolutional neural networks. (October). [Cited on page 85.]

Miron, M., Davies, M. E. P., & Gouyon, F. (2013). An open-source drum transcription system for Pure Data and Max MSP. *ICASSP, IEEE International Conference on Acoustics, Speech and Signal Processing - Proceedings*, pp. 221–225. [Cited on page 97.]

Miron, M., Janer Mestres, J., & Gómez Gutiérrez, E. (2017). Generating data to train convolutional neural networks for classical music source separation. In *Proceedings of the 14th Sound and Music Computing Conference*. Espoo, Finland. [Cited on page 85.]

Mitchell, M. (1998). *An introduction to genetic algorithms*. MIT press. [Cited on page 24.]

Mongillo, D. (2009). The Girl Talk Dilemma: Can Copyright Law Accommodate New Forms of Sample-Based Music. *Pittsburgh Journal of Technology Law and Policy*, *9*, 1. [Cited on page 61.]

Monteiro, A. & Manzolli, J. (2011). A Framework for Real-time Instrumental Sound Segmentation and Labeling. *Proceedings of IV International Conference of Pure data–Weimar*, (June 2014). [Cited on page 97.]

Morgan, N. & Legard, P. (2015). *Parametric composition : computer-assisted strategies for human performance*. Tonality Systems Press. [Cited on page 17.]

Muscutt, K. (2007). Composing with Algorithms: An Interview with David Cope. *Computer Music Journal*, *31*(3), 10–22. [Cited on page 23.]

Neill, B. (2002). Pleasure Beats: Rhythm and the Aesthetics of Current Electronic Music. *Leonardo*, *12*(2002), 3–6. [Cited on page 2.]

Neupert, M. & Gossmann, J. (2013). A remix instrument based on fragment feature-analysis. *Electronic Proceedings of the 2013 IEEE International Conference on Multimedia and Expo Workshops, ICMEW 2013*. [Cited on page 97.]

Nierhaus, G. (2009). *Algorithmic composition: Paradigms of automated music generation*. [Cited on pages 112 and 117.]

Nye, S. (2013). Minimal understandings: The Berlin decade, the minimal continuum, and debates on the legacy of German techno. *Journal of Popular Music Studies*, *25*(2), 154–184. [Cited on page 12.]

Nyman, M. (1999). *Experimental Music: Cage and Beyond (Music in the Twentieth Century)*. Cambridge University Press, 2nd edn. [Cited on page 2.]

Nzewi, M., Anyahuru, I., & Ohiaraumunna, T. (2008). *Musical sense and musical meaning: An indigenous African perception*. Rozenberg Publishers. [Cited on page 28.]

Ó Nuanáin, C., Hermant, M., Faraldo, Á., & Gómez, D. (2015). The EEEAR: Building a Real-Time MIR-Based Instrument from a Hack. In *Late-Breaking Demo Session of the International Society for Music Information Retrieval Conference (ISMIR)*. Málaga, Spain. [Cited on page 179.]

Ó Nuanáin, C., Herrera, P., & Jordà, S. (2016a). An Evaluation Framework and Case Study for Rhythmic Concatenative Synthesis. In *Proceedings of the 17th International Society for Music Information Retrieval Conference*. New York, USA. [Cited on page 140.]

Ó Nuanáin, C., Herrera, P., & Jordà, S. (2017a). Rhythmic Concatenative Synthesis for Electronic Music: Techniques, Implementation, and Evaluation. *Computer Music Journal*, *41*(2), 21–37. [Cited on page 110.]

Ó Nuanáin, C., Herrera, P., & Jordà, S. (2017b). Rhythmic Concatenative Synthesis for Electronic Music: Techniques, Implementation, and Evaluation. *Computer Music Journal*, *41*(2), 21–37. [Cited on page 140.]

Ó Nuanáin, C., Jordà, S., & Herrera, P. (2016b). An Interactive Software Instrument for Real-time Rhythmic Concatenative Synthesis. In *New Interfaces for Musical Expression*. Brisbane, Australia. [Cited on pages 110 and 140.]

Ó Nuanáin, C., Jordà, S., & Herrera, P. (2016c). Towards User-Tailored Creative Applications of Concatenative Synthesis in Electronic Dance Music. In *International Workshop on Musical Metacreation (MUME)*. Paris, France. [Cited on pages 140 and 158.]

Ó Nuanáin, C., Jordà, S., & Herrera, P. (2017c). k -Best Hidden Markov Model Decoding for Unit Selection in Concatenative Sound Synthesis. In *Proceedings of the 13th International Symposium on Computer Music Multidisciplinary Research*. Porto, Portugal. [Cited on pages 113 and 121.]

Ó Nuanáin, C. & Sullivan, L. O. (2014). Real-time Algorithmic Composition with a Tabletop Musical Interface - A First Prototype and Performance. In *AM '14 Proceedings of the 9th Audio Mostly: A Conference on Interaction With Sound*. Aalborg, Denmark. [Cited on pages 18 and 35.]

O'Connell, J. (2011). *Musical Mosaicing with High Level Descriptors*. Ph.D. thesis, Universitat Pompeu Fabra. [Cited on pages 67 and 70.]

Oliveira, J. L., Davies, M. E. P., Gouyon, F., & Reis, L. P. (2012). Beat Tracking for Multiple Applications: A Multi-Agent System Architecture With State Recovery. *Audio, Speech, and Language Processing, IEEE Transactions on*, *20*(10), 2696–2706. [Cited on page 82.]

Oliver, R. (2015). Breakbeat syncretism: The drum sample in African American popular music. *African American culture and society after Rodney King*, pp. 177–192. [Cited on page 59.]

Orio, N. (2006). Music Retrieval: A Tutorial and Review. *Foundations and Trends® in Information Retrieval*, *1*(1), 1–96. [Cited on page 102.]

Orio, N., Lemouton, S., & Schwarz, D. (2003). Score following: State of the art and new developments. *Proceedings of the Conference on New Interfaces for Musical Expression*, pp. 36–41. [Cited on page 116.]

Oswald, J. (1985). Plunderphonics, or Audio Piracy as a Compositional Prerogative. In *Wired Society Electro-Acoustic Conference*. Toronto. [Cited on page 60.]

Pachet, F. (2002). The Continuator: Musical Interaction With Style. In *Proceedings of the International Computer Music Conference*, pp. 333–341. [Cited on page 20.]

Pachet, F. (2008). The Future of Content is in Ourselves. *Computers in Entertainment (CIE)*, *6*(3), 1–20. [Cited on page 20.]

Pachet, F., Papadopoulos, A., & Roy, P. (2017). Sampling Variations of Sequences for Structured Music Generation. *Proceedings of the 18th International Society for Music Information Retrieval Conference (ISMIR)*. [Cited on page 3.]

Pachet, F. & Roy, P. (2011). Markov constraints: steerable generation of Markov sequences. *Constraints*, *16*(2), 148–172. [Cited on page 21.]

Pachet, F. & Roy, P. (2015). (Manufac) Turing Tests for Music. In *Proceedings of the 29th Conference on Artificial Intelligence (AAAI 2015), Workshop on "Beyond the Turing Test"*. Austin, Texas USA. [Cited on page 151.]

Pachet, F., Roy, P., Barbieri, G., & Paris, S. C. S. L. (2011). Finite-length Markov processes with constraints. *Proceedings of the Twenty-Second International Joint Conference on Artificial Intelligence*, *6*(1/3). [Cited on page 21.]

Pachet, F., Roy, P., & Ghedini, F. (2008). Creativity through Style Manipulation: the Flow Machines project. In *Marconi Institute for Creativity Conference*. Bologna, Italy. [Cited on page 20.]

Paiement, J.-F., Grandvalet, Y., Bengio, S., & Eck, D. (2007). A generative model for rhythms. *NIPS'2007 Music Brain Cognition Workshop*, pp. 1–8. [Cited on page 35.]

Pampalk, E., Flexer, A., & Widmer, G. (2005). Hierarchical Organization and Description of Music Collections at the Artist Level. In A. Rauber, S. Christodoulakis, & A. M. Tjoa (Eds.) *Research and Advanced Technology for Digital Libraries: 9th European Conference, ECDL 2005, Vienna, Austria, September 18-23, 2005. Proceedings*, pp. 37–48. Berlin, Heidelberg: Springer Berlin Heidelberg. [Cited on page 7.]

Panteli, M., Bogaards, N., & Honingh, A. (2014). Modeling Rhythm Similarity for Electronic Dance Music. In *15th International Society for Music Information Retrieval Conference (ISMIR 2014) MODELING*, Ismir, pp. 0–5. Taipei, Taiwan. [Cited on pages 49 and 51.]

Papadopoulos, G. & Wiggins, G. (1999). AI Methods for Algorithmic Composition : A Survey, a Critical View and Future Prospects. *AISB Symposium on Musical Creativity*, pp. 110–117. [Cited on page 19.]

Papadopoulos, H. & Peeters, G. (2007). Large-scale study of chord estimation algorithms based on chroma representation and HMM. *CBMI'2007 - 2007 International Workshop on Content-Based Multimedia Indexing, Proceedings*, pp. 53–60. [Cited on page 116.]

Parncutt, R. (1994). A perceptual model of pulse salience and metrical accent in musical rhythms. *Music Perception*. [Cited on page 33.]

Pasquier, P., Eigenfeldt, A., Bown, O., & Dubnov, S. (2017). An Introduction to Musical Metacreation. *Computers in Entertainment*, *14*(2), 1–14. [Cited on page 3.]

Patricio, L. & Dittmar, C. (2016). Towards Modeling and Decomposing Loop-Based Electronic Music. *Proc. 17th International Society for Music Information Retrieval Conference*, pp. 502–508. [Cited on page 2.]

Patterson, R., Nimmo-Smith, I., Holdsworth, J., & Rice, P. (1987). An Efficient Auditory Filterbank Based On The Gammatone Function. In *a meeting of the IOC Speech Group on Auditory Modelling at RSRE*, December, pp. 14–15. Malvern, UK. [Cited on page 99.]

Pease, A. & Colton, S. (2011). On impact and evaluation in computational creativity: a discussion of the turing test and an alternative proposal. *Proceedings of the AISB symposium on AI . . . ,*, p. 39. [Cited on page 18.]

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., & Duchesnay, É. (2012). Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research*, *12*, 2825–2830. [Cited on page 104.]

Peeters, G. (2004). A large set of audio features for sound description (similarity and classification) in the CUIDADO project. *CUIDADO IST Project Report*, *54*(0), 1–25. [Cited on pages 88 and 100.]

Pishdadian, F., Pardo, B., & Liutkus, A. (2017). A Multi-resolution approach to Common Fate-based audio separation. *ICASSP, IEEE International Conference on Acoustics, Speech and Signal Processing - Proceedings*, pp. 566–570. [Cited on page 103.]

Pollard, H. F. & Jansson, E. V. (1982). A tristimulus method for the specification of musical timbre. *Acustica*, *51*, 162–171. [Cited on page 100.]

Pope, R. (2011). Hooked on an affect: Detroit techno and dystopian digital culture. *Dancecult: Journal of Electronic Dance Music Culture*, *2*(1), 24–44. [Cited on page 11.]

Post, O. & Toussaint, G. (2011). The Edit Distance as a Measure of Perceived Rhythmic Similarity. *Empirical Musicology Review*, *6*(3), 164–179. [Cited on page 38.]

Puckette, M. (1997). Pure Data : another integrated computer music environment. *Proceedings, Second Intercollege Computer Music Concerts*, pp. 37–41. [Cited on page 36.]

Puckette, M. (2004). Low-dimensional parameter mapping using spectral envelopes. *Proceedings of the International Computer Music Conference*, *2004*, 406–408. [Cited on page 73.]

Puckette, M. (2006). *The Theory and Technique of Electronic Music*, vol. 11. World Scientific Publishing Co Inc. [Cited on pages 86 and 90.]

Rabiner, L. & Juang, B.-H. (1993). Fundamentals of Speech Recognition. [Cited on page 122.]

Rabiner, L. R. (1989). A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition. [Cited on pages 113 and 114.]

Raffel, C., Mcfee, B., Humphrey, E. J., Salamon, J., Nieto, O., Liang, D., & Ellis, D. P. W. (2014). mir_eval: A Transparent Implementation of Common MIR Metrics. *Proc. of the 15th International Society for Music Information Retrieval Conference*, pp. 367–372. [Cited on page 77.]

Ravelli, E., Bello, J. P., & Sandler, M. (2007). Automatic rhythm modification of drum loops. *IEEE Signal Processing Letters*, *14*(4), 228–231. [Cited on page 153.]

Reich, S. & Hillier, P. (2011). *Writings on Music 1965-2000: 1965-2000*. [Cited on page 2.]

Reid, G. (2002). Synth Secrets: Practical Bass Drum Synthesis. *Sound on Sound, July*. [Cited on page 92.]

Reynolds, C. W. (1987). Flocks, herds and schools: A distributed behavioral model. *ACM SIGGRAPH computer graphics*, *21*(4), 25–34. [Cited on page 139.]

Reynolds, S. (2013). *Energy Flash: A Journey Through Rave Music and Dance Culture*. Faber & Faber. [Cited on pages 9, 11, and 13.]

Risset, J.-C. & Wessel, D. L. (1999). Exploration of Timbre by Analysis and Synthesis. *The Psychology of Music*, (Second Edition), 113–169. [Cited on page 92.]

Roads, C. (1985). Granular synthesis of sound. [Cited on page 57.]

Roads, C. (1988). Introduction to granular synthesis. *Computer Music Journal*, *12*(2), 11–13. [Cited on page 57.]

Roads, C. (1991). Asynchronous granular synthesis. In *Representations of musical signals*, pp. 143–186. MIT Press. [Cited on page 57.]

Roads, C. (1996). *The computer music tutorial*, vol. 32. MIT Press. [Cited on pages 3, 19, 21, 23, 57, 73, 74, 89, and 93.]

Roads, C. (2004). *Microsound*. [Cited on pages 57 and 73.]

Robindore, B. & Xenakis, I. (1996). Eskhate Ereuna : Limits of Extending the Musical Thought: Comments On and By Iannis Xenakis. *Computer Music Journal*, *20*(4), 11–16. [Cited on page 57.]

Rodgers, T. (2003). On the process and aesthetics of sampling in electronic music production. *Organised Sound*, *8*(03), 313–320. [Cited on page 59.]

Rogers, Y., Sharp, H., & Preece, J. (2011). *Interaction design: beyond human-computer interaction*. John Wiley & Sons. [Cited on page 130.]

Roma, G. & Herrera, P. (2010). Graph grammar representation for collaborative sample-based music creation. *Proceedings of the 5th Audio Mostly Conference*. [Cited on page 147.]

Romero, J. J., Machado, P., & Ashlock, D. (2008). The Art of Artificial Evolution: A Handbook on Evolutionary Art and Music. *Journal of Mathematics and the Arts*, *2*, 103–106. [Cited on page 25.]

Ross, A. (2007). *The rest is noise: Listening to the twentieth century*. Macmillan. [Cited on pages 2 and 90.]

Russ, M. (2004). *Sound Synthesis and Sampling*. Taylor & Francis, 2nd edn. [Cited on page 88.]

Russell, S. & Norvig, P. (2002). *Artificial Intelligence: A Modern Approach (2nd Edition)*. [Cited on pages 63, 106, 111, and 120.]

Savage, P. E., Brown, S., Sakai, E., & Currie, T. E. (2015). Statistical universals reveal the structures and functions of human music. *Proceedings of the National Academy of Sciences*, *112*(29), 8987–8992. [Cited on page 22.]

Schloss, A. W. (1985). On the Automatic Transcription of Percussive Music - From Acoustic Signal to High-Level Analysis. [Cited on page 74.]

Schlüter, J. & Böck, S. (2013). Musical Onset Detection with Convolutional Neural Networks. In *6th International Workshop on Machine Learning and Music*. Prague, Czech Republic. [Cited on pages 74 and 77.]

Schlüter, J. & Böck, S. (2014). Improved musical onset detection with convolutional neural networks. *IEEE International Conference on Acoustic, Speech and Signal Processing (ICASSP)*, pp. 7029–7033. [Cited on pages 74 and 77.]

Schlüter, R., Bezrukov, I., Wagner, H., & Ney, H. (2007). Gammatone Features and Feature Combination For Large Vocabulary Speech Recognition. In *Acoustics, Speech and Signal Processing, ICASSP 2007*, pp. 4–7. Hawaii, USA. [Cited on page 100.]

Schröder, M. (2001). Emotional speech synthesis: A review. In *Seventh European Conference on Speech Communication and Technology*. [Cited on page 62.]

Schubert, E., Wolfe, J., & Tarnopolsky, A. (2004). Spectral centroid and timbre in complex, multiple instrumental textures. In *Proceedings of the 8th International Conference on Music Perception and Cognition*, August 2004, pp. 654–657. Illinois, USA. [Cited on page 91.]

Schwarz, D. (2000). A system for data-driven concatenative sound synthesis. *Digital Audio Effects (DAFx)*, pp. 97–102. [Cited on pages 61 and 70.]

Schwarz, D. (2003). The Caterpillar System for Data-Driven Concateantive Sound Synthesis. In *Proceedings of the 6th Int. Conference on Digital Audio Effects (DAFx-03)*, pp. 1–6. [Cited on pages 63, 73, 111, 112, and 121.]

Schwarz, D. (2005). Current Research In Concatenative Sound Synthesis. In *Proceedings of the International Computer Music Conference*, pp. 9–12. [Cited on page 62.]

Schwarz, D. (2006). Concatenative Sound Synthesis: The Early Years. *Journal of New Music Research*, *35*(1), 3–22. [Cited on pages 62, 73, 109, 124, and 146.]

Schwarz, D. (2011). Distance Mapping for Corpus-Based Concatenative Synthesis. In *Sound and Music Computing Conference (SMC)*. Padova, Italy. [Cited on page 110.]

Schwarz, D., Beller, G., Verbrugghe, B., & Britton, S. (2006). Real-Time Corpus-Based Concatenative Synthesis with CataRT. *Proceedings of the 9th International Conference on Digital Audio Effects*, pp. 18–21. [Cited on pages 64, 70, 79, and 132.]

Schwarz, D. & Hackbarth, B. (2012). Navigating variation: composing for audio mosaicing. In *International Computer Music Conference*, pp. 1–4. [Cited on page 66.]

Schwarz, D., Schnell, N., & Gulluni, S. (2009). Scalability in Content-Based Navigation of Sound Databases. In *Proceedings of the International Computer Music Conference*, pp. 253–258. [Cited on page 110.]

Sequera, R. S. (2006). Timbrescape : a Musical Timbre and Structure Visualization Method Using Tristimulus Data. In *International Conference on Music Perception and Cognition*, March, pp. 352–356. [Cited on page 100.]

Seshadri, N. & Sundberg, C.-E. (1994). List Viterbi decoding algorithms with applications. *IEEE Transactions on Communications*, *42*(2/3/4), 313–323. [Cited on pages 118 and 119.]

Sethares, W. A. (2007). *Rhythm and transforms*. [Cited on pages 33, 38, and 82.]

Sewell, A. (2013). *A Typology of Sampling in Hip-Hop*. Ph.D. thesis, Indiana University. [Cited on page 59.]

Sewell, A. (2014). Paul's Boutique and Fear of a Black Planet: Digital Sampling and Musical Style in Hip Hop. *Journal of the Society for American Music*. [Cited on page 59.]

Shao, Y., Jin, Z., Wang, D., & Srinivasan, S. (2009). An auditory-based feature for robust speech recognition. *ICASSP, IEEE International Conference on Acoustics, Speech and Signal Processing - Proceedings*, (1), 4625–4628. [Cited on page 94.]

Sheh, A. & Ellis, D. P. W. (2003). Chord segmentation and recognition using EM-trained hidden markov models. *Proceedings of the International Conference on Music Information Retrieval (ISMIR)*, pp. 185–191. [Cited on page 116.]

Sicko, D. (2010). *Techno Rebels: The Renegades of Electronic Funk*. [Cited on pages 10 and 11.]

Siedenburg, K. & McAdams, S. (2017). Four distinctions for the auditory "wastebasket" of timbre. *Frontiers in Psychology*, *8*(OCT), 6–9. [Cited on page 89.]

Siek, J. G., Lee, L.-Q., & Lumsdaine, A. (2002). *The Boost Graph Library*. [Cited on page 40.]

Simon, I., Basu, S., Salesin, D., & Agrawala, M. (2005). Audio analogies: creating new music from an existing performance by concatenative synthesi. *Proceedings of the International Computer Music Conference*, *2005*, 65–72. [Cited on pages 70 and 151.]

Skovenborg, E. & Nielsen, S. H. (2004). Evaluation of Different Loudness Models with Music and Speech Material. In *Proceedings of the Audio Engineering Society Convention*. [Cited on page 87.]

Slaney, M. (1998). Auditory toolbox. *Interval Research Corporation, Tech. Rep*, *10*, 1998. [Cited on pages 98 and 99.]

Smith, J. B. L., Percival, G., Kato, J., Goto, M., & Fukayama, S. (2015). CrossSong Puzzle: Generating and Unscrambling Music Mashups with Real-time Interactivity. In *Sound and Music Computing Conference*. Maynooth, Ireland. [Cited on pages 61 and 124.]

Smith, L. I. (2002). A tutorial on Principal Components Analysis Introduction. *Statistics*, *51*, 52. [Cited on page 134.]

Smith, S. (2000). Compositional strategies of the hip-hop turntablist. *Organised Sound*, *5*(2), 75–79. [Cited on pages 13 and 59.]

Somerville, P. & Uitdenbogerd, A. L. (2008). Multitimbral musical instrument classification. *Proceedings - International Symposium on Computer Science and Its Applications, CSA 2008*, pp. 269–274. [Cited on page 105.]

Song, C., Pearce, M., & Harte, C. (2015). SYNPY: a python toolkit for syncopation modelling. *Proceedings of the 12th International Conference on Sound and Music Computing (SMC-15)*, pp. 295–300. [Cited on page 51.]

Song, C., Simpson, A. J. R., Harte, C. A., Pearce, M. T., & Sandler, M. B. (2013). Syncopation and the Score. *PloS one*, *8*(9), e74692. [Cited on page 51.]

Srinivas, M. & Patnaik, L. M. (1994). Genetic Algorithms: A Survey. *Computer*, *27*(6), 17–26. [Cited on page 24.]

Stark, A. M., Davies, M. E. P., & Plumbley, M. D. (2009). Real-Time Beat-Synchronous Analysis of Musical Audio. *Proc. of the 12th Int. Conference on Digital Audio Effects*, pp. 1–6. [Cited on page 81.]

Stevens, S. S. (1955). The Measurement of Loudness. *Journal of the Acoustical Society of America*, *27*(5), 815. [Cited on page 86.]

Stevens, S. S. (1975). *Psychophysics*. Transaction Publishers. [Cited on pages 87 and 88.]

Stevens, S. S., Volkmann, J., & Newman, E. B. (1937). A scale for the measurement of the psychological magnitude pitch. *The Journal of the Acoustical Society of America*, *8*(3), 185–190. [Cited on page 94.]

Stoll, T. (2013). CorpusDB: Software for Analysis, Storage, and Manipulation of Sound Corpora. In *InternationalWorkshop on Musical Metacreation (MuMe)*, Figure 1, pp. 108–113. [Cited on page 110.]

Sturm, B. L. (2004). Matconcat: An Application for Exploring Concatenative Sound Synthesis Using Matlab. In *7th International Conference On Digital Audio Effects (DAFx)*, pp. 323–326. [Cited on pages 64, 70, 86, and 151.]

Sturm, B. L. (2006). Adaptive Concatenative Sound Synthesis and Its Application to Micromontage Composition. *Computer Music Journal*, *30*(4), 46–66. [Cited on pages 62 and 110.]

Tamagawa, K. (1988). *Echoes from the East: The Javanese gamelan and its influence on the music of Claude Debussy*. Ph.D. thesis. [Cited on page 56.]

Tardieu, D. (2008). *Modèles d'instruments pour l'aide à l' orchestration*. Ph.D. thesis, Université Pierre Et Marie Curie. [Cited on page 84.]

Taube, H. (1991). Common Music: A music composition language in Common Lisp and CLOS. *Computer Music Journal*, *15*(2), 21–32. [Cited on page 23.]

Taube, H. (2004). Notes from the metalevel: introduction to algorithmic music composition. [Cited on page 23.]

Terasawa, H., Slaney, M., & Berger, J. (2006). Determining the Euclidean distance between two steady state sounds. *Proceedings of the 9th . . . .* [Cited on page 100.]

Théberge, P. (1997). *Any sound you can imagine: Making music/consuming technology*. Wesleyan University Press. [Cited on page 27.]

Thompson, L., Dixon, S., & Mauch, M. (2014). Drum Transcription via Classification of Bar-Level Rhythmic Patterns. In *International Society for Music Information Retrieval Conference*, pp. 187–192. [Cited on page 153.]

Thomson, P. (2004). Atoms and errors: towards a history and aesthetics of microsound. *Organised Sound*, *9*(02), 1–37. [Cited on pages 57 and 60.]

Tian, M. & Sandler, M. B. (2016). Music Structural Segmentation Across Genres with Gammatone Features. In *17th International Society for Music Information Retrieval Conference*. [Cited on page 100.]

Tihelka, D., Kala, J., & Matoušek, J. (2010). Enhancements of Viterbi search for fast unit selection synthesis. *Interspeech*, (September), 174–177. [Cited on page 62.]

Tindale, A. (2009). *Advancing the Art of Electronic Percussion*. Ph.D. thesis, University of Victoria. [Cited on page 27.]

Todd, P. M. & Werner, G. M. (1999). Frankensteinian methods for evolutionary music. *Musical networks: parallel distributed perception and performace*, pp. 313–340. [Cited on page 26.]

Tokui, N. (2008). Massh!: a web-based collective music mashup system. In *Proceedings of the 3rd international conference on Digital Interactive Media in Entertainment and Arts*, pp. 526–527. ACM. [Cited on page 61.]

Tomás, E. & Kaltenbrunner, M. (2014). Tangible Scores: Shaping the Inherent Instrument Score. *Proceedings of the International Conference on New Interfaces for Musical Expression*, pp. 609–614. [Cited on page 97.]

Toussaint, G. (2003). Classification and Phylogenetic Analysis of African Ternary Rhythm Timelines. In *BRIDGES: Mathematical Connections in Art, Music and Science*, pp. 25–36. Granada, Spain. [Cited on pages 28 and 32.]

Toussaint, G. (2004a). A Comparison of Rhythmic Similarity Measures. *ISMIR*, pp. 3–6. [Cited on pages 29 and 34.]

Toussaint, G. T. (2004b). A mathematical measure of preference in African rhythm. In *Papers Presented to the American Mathematical Society*, vol. 25, p. 248. [Cited on page 50.]

Toussaint, G. T. (2013). *The Geometry of Musical Rhythm: What Makes a " good" Rhythm Good?* CRC Press. [Cited on pages 27, 31, 33, and 49.]

Toussaint, G. T. & Oh, S. M. (2016). Measuring Musical Rhythm Similarity: Edit Distance versus Minimum-Weight Many-to-Many Matchings. In *Proceedings of the International Conference on Artificial Intelligence*, pp. 186–. Athens, Greece. [Cited on page 32.]

Truax, B. (1988). Real-time granular synthesis with a digital signal processor. *Computer Music Journal*, *12*(2), 14–26. [Cited on page 57.]

Truax, B. (2005). Music and science meet at the micro level: Time-frequency methods and granular synthesis. *Journal of the Acoustical Society of America*, *117*(4), 2415. [Cited on page 62.]

Valero, X. & Alias, F. (2012). Gammatone cepstral coefficients: Biologically inspired features for non-speech audio classification. *IEEE Transactions on Multimedia*, *14*(6), 1684–1689. [Cited on pages 99 and 100.]

Van Der Maaten, L. J. P. & Hinton, G. E. (2008). Visualizing high-dimensional data using t-sne. *Journal of Machine Learning Research*, *9*, 2579–2605. [Cited on page 135.]

Varèse, E. & Wen-chung, C. (1966). The Liberation of Sound. *Perspectives of New Music*, *5*(1), 11. [Cited on page 1.]

Veltkamp, R. C., Wiering, F., & Typke, R. (2008). Content based music retrieval. In *Encyclopedia of Multimedia*, pp. 97–98. Springer. [Cited on page 55.]

Vitos, B. (2014). Along the Lines of the Roland TB-303: Three Perversions of Acid Techno. *Dancecult: Journal of Electronic Dance Music Culture*, *6*(1). [Cited on page 10.]

Vogiatzoglou, I. (2016). *Application*. Ph.D. thesis, Aalborg University, Copenhagen. [Cited on page 97.]

Vogl, R. & Knees, P. (2017). An Intelligent Drum Machine for Electronic Dance Music Production and Performance. In *New Interfaces for Musical Expression*, pp. 251–256. Copenhagen. [Cited on page 35.]

Vogl, R., Leimeister, M., Ó Nuanáin, C., Jordà, S., Hlatky, M., & Knees, P. (2016). An Intelligent Interface for Drum Pattern Variation and Comparative Evaluation of Algorithms. *Journal of the Audio Engineering Society*, *64*(7), 503–513. [Cited on pages 35 and 53.]

Wang, J.-C., Wang, J.-F., He, K. W., & Hsu, C.-S. (2006). Environmental Sound Classification using Hybrid SVM/KNN Classifier and MPEG-7 Audio Low-Level Descriptor. *The 2006 IEEE International Joint Conference on Neural Network Proceedings*, pp. 1731–1735. [Cited on pages 85 and 105.]

Weihs, C. & Jannach, D. (2009). *Music Data Analysis: Foundations and Applications*. [Cited on page 90.]

Wessel, D. (1979). Timbre space as a musical control structure. *Computer Music Journal*, *3*(2), 45–52. [Cited on pages 132 and 134.]

Wessel, D. L. (1976). Perceptually based controls for additive synthesis. In *International Computer Music Conference, Massachusetts Institute of Technology*. [Cited on pages 132 and 134.]

Wiggins, G. A. (2006). Searching for computational creativity. *New Generation Computing*, *24*(3), 209–222. [Cited on page 17.]

Wiggins, G. A. (2008). Computer models of musical creativity: A review of computer models of musical creativity by David Cope. *Literary and Linguistic Computing*, *23*(1), 109–116. [Cited on page 23.]

Wiggins, G. A., Pearce, M. T., & Müllensiefen, D. (2012). Computational Modeling of Music Cognition and Musical Creativity. *The Oxford Handbook of Computer Music*, (January). [Cited on page 51.]

Wilson, S. (2008). Spatial Swarm Granulation. *Proceedings of the 2008 International Computer Music Conference*, pp. 4–7. [Cited on page 139.]

Winer, E. (2012). *The audio expert: everything you need to know about audio*. CRC Press. [Cited on page 92.]

Winkler, T. (2001). *Composing interactive music: techniques and ideas using Max*. MIT press. [Cited on page 36.]

Wirth, N., Wirth, N., Wirth, N., Informaticien, S., & Wirth, N. (1996). *Compiler construction*, vol. 1. Addison-Wesley Reading. [Cited on page 22.]

Xiang, P. (2002). A New Scheme for Real-Time Loop Music Production Based on Granular Similarity and Probability Control. *Proceedings of the 5th International Conference on Digital Audio Effects (DAFx-02)*, pp. 89–92. [Cited on pages 67 and 70.]

Xu, C., Maddage, N. C., Shao, X., Cao, F., & Tian, Q. (2003). Musical Gnere Classification Using Support Vector Machines. In *Acoustics, Speech, and Signal Processing, 2003 (ICASSP'03)*, pp. 429–432. [Cited on page 105.]

Yen, J. Y. (1971). Finding the K Shortest Loopless Paths in a Network. *Management Science*, *17*(11), 712–716. [Cited on page 120.]

Young, S., Evermann, G., Gales, M., Hain, T., Kershaw, D., Liu, X., Moore, G., Odell, J., Ollason, D., Povey, D., & Others (2002). The HTK book. *Cambridge University*, *3*, 175. [Cited on pages 62 and 98.]

Zapata, J. R., Davies, M. E. P., & Gómez, E. (2014). Multi-feature beat tracking. *IEEE Transactions on Audio, Speech and Language Processing*, *22*(4), 816–825. [Cited on pages 81 and 82.]

Zapata, J. R., Holzapfel, A., Davied, M. E., Oliveira, J. L., & Gouyon, F. (2012). Assigning a Confidence Threshold on Automatic Beat Annotation in Large Datasets. *Proc. of the Int. Conf. on Music Information Retrieval (ISMIR)*, (Ismir), 157–162. [Cited on page 82.]

Zhao, X. & Wang, D. (2013). Analyzing noise robustness of MFCC and GFCC features in speaker identification. *ICASSP, IEEE International Conference on Acoustics, Speech and Signal Processing - Proceedings*, pp. 7204–7208. [Cited on page 100.]

Zils, A. & Pachet, F. (2001). Musical mosaicing. *Digital Audio Effects (DAFx)*, pp. 1–6. [Cited on pages 61, 63, 70, and 112.]

Zils, A., Pachet, F., Delerue, O., & Gouyon, F. (2002). Automatic extraction of drum tracks from polyphonic music signals. In *Web Delivering of Music, 2002. WEDELMUSIC 2002. Proceedings. Second International Conference on*, pp. 179–183. IEEE. [Cited on page 49.]