Database Music

A History, Technology, and Aesthetics of the Database in Music Composition

by

Federico Nicolás Cámara Halac

A dissertation submitted in partial fulfillment

of the requirements for the degree of

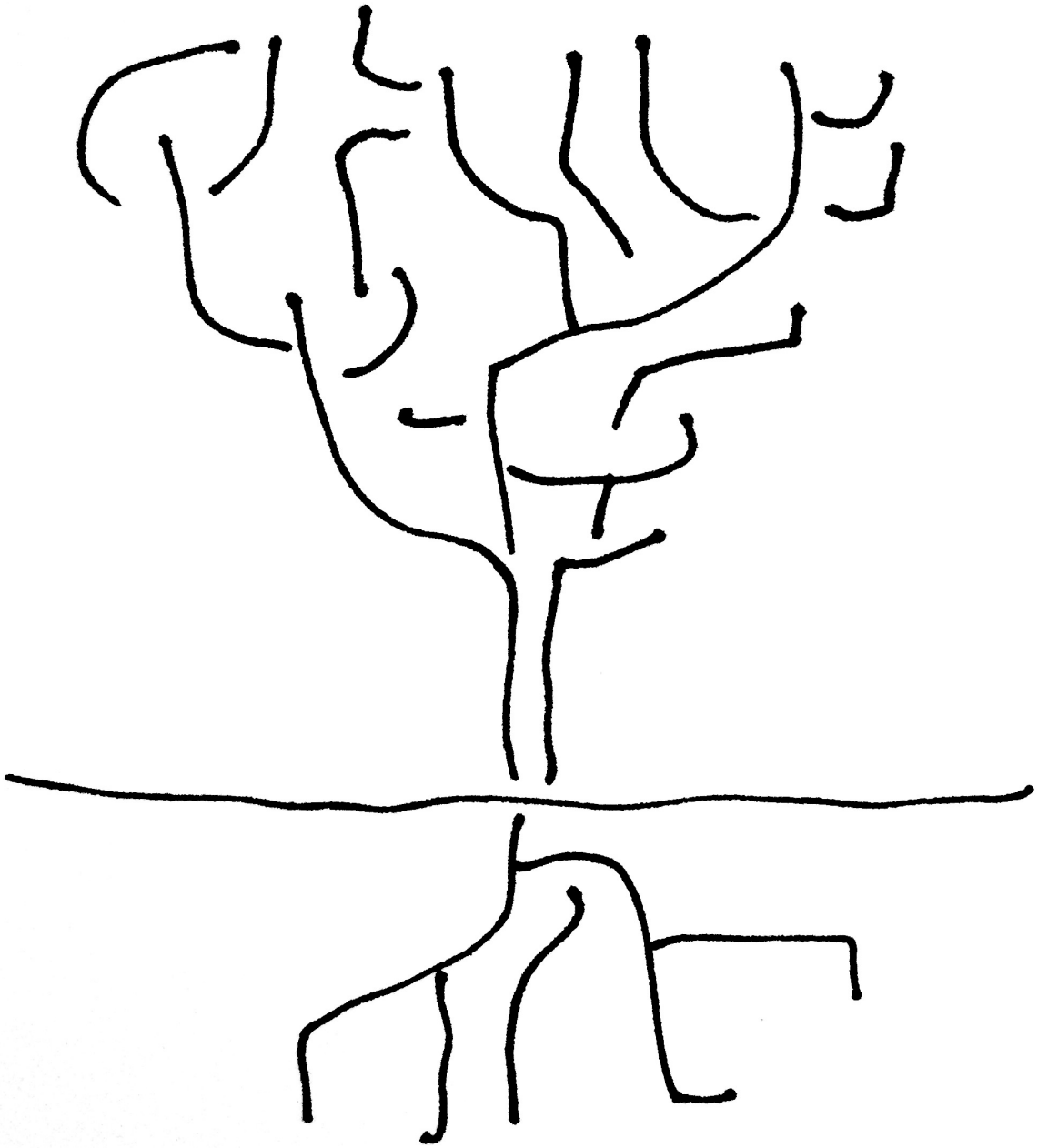
Doctor of Philosophy

Department of Music

New York University

May, 2019

Jaime Oliver La Rosa



Dedication

For my mother and father, who have always taught me to never give up with my research, even during the most difficult times. Also to my advisor, Jaime Oliver La Rosa, without his help and continuous guidance, this would have never been possible. For Elizabeth Hoffman and Judy Klein, who always believed in me, and whose words and music I bring everywhere. Finally to Aye, whose love I cannot even begin to describe.

Acknowledgements

I would like to thank my advisor, Jaime Oliver La Rosa, for his role in inspiring this project, as well as his commitment to research, clarity, and academic rigor. I am also indebted to committee members Martin Daughtry and Elizabeth Hoffman, for their ongoing guidance and support even at the very early stages of this project, and William Brent and Robert Rowe, whose insightful, thought-provoking input made this dissertation come to fruition. I am also everlastingly grateful to Judy Klein, for always being available to listen and share her listening. As well as to Aye, for her endless support and her helping me maintain hope in developing this project. I would also like to thank my parents, Ana and Hector, who inspired and nurtured my interest in music from a young age, and my sister Flor and my brother Joaquin who were always with me, next to every word. Finally, this dissertation could not have been possible without the support and help of my friends, some of which I would like to mention by name because they affected directly certain aspects of this text. I would like to thank Matias Delgadino and Lucia Simonelli, for their continuous layers of abstraction; Matias Borg Oviedo, for those endless conversations; Ioannis Angelakis, for his glass sculptures.

Preface

Dataloquy (you don't need to ~~write~~ this) (The initial title that explains how databases are everywhere) The name of the database. (Now think of the data, and the base, and how these relate) These words must point to two things placed one inside the other. (Is it base in data or is it data in base?) The base (of the data). A basement, a basis, a basic foundation for data. The house where data resides. (Is it the base or the data that is economical? Or both?) The addresses in which they are located. (Clearly, you are talking about pointers) The discretized space that guards data. (Guardians of space? This starts to look like a bad sci-fi thing...) Data as in the plurality of datum, and database as the plain (*planicie* in spanish) or the lattice upon which the address space is laid for data. (Data under house arrest) Datum as in bit, as the zero or the one, and nothing in between (Are you sure there is nothing in the middle?) Data as in bytes, and the eight bits that follow it around (like ducklings without mom [*pata* in spanish]) Data as in data types, the many names of the binary words representing the values of almost all numbers (and this 'almost' is still more than enough (Some would disagree)). Data as in data structures and their unions, symbols, lists, tables, arrays, sequences, dictionaries, simultaneously pointing to their interfaces and their implementations and assemblage (The assembly is in order) Data as in files *fichiers* in french, *archivos* in spanish, and their kilobytes and megabytes (inside directories and folders, etc.) Data as in data flow and data streams (are there data fountains?) Data you translate from slot to slot, transmit from client to server to client, transduce to and fro with `adc` and `dac`, transcode from format to format (transgress from torrent to torrent) Data as in dataset for your algorithms to test,

to improve, to fit, to make them more efficient, to teach them the right tendencies, to drive your models data-driven (You are driving me crazy) Data as in data banks (also its transactions and currency) Data as in data corpus (Oh, so it has body?) Data as in database (finally), basing gigabytes with models meant for system management and warehouses, repositories, terabytes, their mining, and their subsequent data clouds, clusters, spacing out into the (in)famous big data leap from bit to big.

Abstract

The aim of this dissertation is to outline a framework to discuss the aesthetic agency of the database in music composition. I place my dissertation in relation to existing scholarship, artists, and developers working in the fields of music composition, computer science, affect, and ontology, with emphasis on the ubiquity of databases and on the need to reflect on their practice, particularly in relation to databasing and music composition. There is a database everywhere, anytime, always already affecting our lives; it is an agent in our aesthetic and political lives just as much as we are agents in its composition and performance. Database music lives in between computers and sound. My argument is that in order to conceptualize the agency of the database in music composition, we need to trace the history of the practice, in both its technical and its artistic use, so as to find nodes of action that have an effect on the resulting aesthetics. Therefore, this dissertation is composed of two main sections.

In the first part, I trace a history of database practices from three points of view. The first is from new media theory, emphasizing the intersection between the database and the body. The second is from the history of the database in computer science, giving a panoramic view of the tools and concepts behind database systems, models, structures. The third is from their use in sound practices, describing different approaches to databasing from the fields of music information retrieval, sonification, and computer music.

In the second part, I discuss the agency of the database under the broader concepts of sound, self, and community. These three axes are addressed in four sections, each with a different

perspective. First I focus on listening and present the database as a resonant subject in a networked relation and community with the human. Second, I focus on memory, comparing human memory and writing with digital information storing, thus relating databasing and composition with memory, archives, and their spectrality. Third, I analyze the performativity of databasing, understanding the database as gendered, in its temporality, repetition, and in its contingent appearance as style, skin, and timbre. In the last section, I revise the notion of music work, reflecting on the consequences of the anarchic and the inoperative in the community of database music.

As an appendix, I present an open-source library for making and performing a multimodal database that combines computer vision and timbre analysis algorithms to generate a database of image and audio descriptors suitable for automated navigation, querying, and live audio-visual performance.

Contents

Dedication iv

Acknowledgements v

Preface vi

Abstract viii

List of Figures xv

List of Tables xvii

Introduction 1

I Database Art 5

1 The Database In New Media Theory 6

1.1 Database As Form 6

1.2 A Semiotic Trap 7

1.3 Digital Convergence 10

1.4 Bodiless Information 12

1.5 Embodying Databasing 14
1.6 Filtering And Framing 15
1.7 Interlude: An Embodied Database 17
1.8 Closing Remarks 22

2 Databasing And The History Of Databases **24**
2.1 Databasing: The Performance Of The Database 24
2.1.1 Data types and structures 25
2.1.2 Temporality of Databasing 26
2.1.3 Databasing and Writing 27
2.1.4 The Von Neumann Architecture 30
2.2 A Database Tree 31
2.3 The Realm Of Data Structures 34
2.4 A Brief History Of Database Models 36
2.4.1 Hierarchical 38
2.4.2 Network 39
2.4.3 Relational 40
2.4.4 Non-Relational 41
2.4.5 Graph 42
2.4.6 Object 42
2.4.7 Semi-structured 43
2.4.8 Pure Data as Database System 44

3 Databasing Sound: Applications Of Databases In Sound **47**
3.1 Music Information Retrieval 48
3.2 Sonification 52
3.2.1 Parameter mapping 54

3.2.2 Model-based sonification . . . 55
3.2.3 Artistic sonification . . . 55
3.2.4 Sonification Installations . . . 57
3.2.5 Sonification Software . . . 59
3.3 Computer Music . . . 60
3.3.1 Hierarchical environments . . . 61
3.3.2 Music Notation Software . . . 67
3.3.3 Enter Objects . . . 70
3.4 Intersections . . . 77
3.4.1 Corpus-based Approaches . . . 77
3.4.2 Querying Methods . . . 80
3.4.3 Traversing Methods . . . 82
3.4.4 Resource Sharing . . . 85
3.4.5 Closing Remarks . . . 88

II Database Aesthetics 91

4 Listening Databases 94

4.1 Interlude: I Am Sitting In A Room... . . . 94
4.2 The Resonance Of A Return . . . 95
4.3 Resonant Network . . . 99
4.4 The Unworking Network . . . 103

5 Databases And Memory 111

5.1 The Effraction Of The Trace . . . 111
5.2 The Archontic Principle . . . 116

5.3 The Spectral Database 121

6 Performativity Of Databases **126**

6.1 Gendered Database 126

6.2 Towards The Limits 130

6.3 Contingencies Of Style 131

6.4 A Specter Of Authority 137

7 Rethinking Composition **141**

7.1 Interlude: Hyperbolic Reactions 141

7.2 Working Composition 145

7.3 The Composer As Navigator 149

7.4 The Database As Performer 153

7.5 The Severed Object Of Music 160

7.6 Anarchy And The Unwork 165

7.7 [Wip] Work In Progress 170

Afterword **173**

Appendix **177**

8 Multimodal Database: mmdb **177**

8.1 Overview 177

8.2 Steps 180

8.2.1 Download 180

8.2.2 Preprocess 181

8.2.3 Analyze Images 181

8.2.4 Sorter . . . 182
8.2.5 Color Sounds . . . 182
8.2.6 Analyze Sounds . . . 184
8.2.7 Live Query . . . 186
8.3 Extra . . . 186
8.3.1 Reader / Visualizer . . . 186
8.3.2 Image Query (non-live) . . . 187
8.4 Dependencies . . . 187
8.4.1 Externals . . . 187
8.4.2 Abstractions . . . 188

Glossary **189**

Acronyms **202**

Bibliography **231**

List of Figures

1.1 Syntagm and Paradigm reversal 9
1.2 Manovich's cultural algorithm 22

2.1 A Database Tree 32
2.2 Hierarchical Model 38
2.3 Network Model 39
2.4 Relational Model 41
2.5 Object Model 42
2.6 Semi-structured Model 43

3.1 Database performance and interdisciplinary feedback. 47
3.2 Diagram of database performance in Music Information Retrieval (MIR) practices. 49
3.3 Diagram of database performance in sonification practices. 52
3.4 Diagram of database performance in computer music practices. 61
3.5 Generality vs. Strength 66
3.6 A real-time version of MUSIC-11. 70
3.7 Intersection space. 89

4.1 Community as work 104
4.2 Community as unwork 105
4.3 Spandex fibers under an optical microscope 107

6.1 Lorenz Attractor 134

8.1 Diagram representing each step 179

List of Tables

2.1 Database model development timeline with examples. 46

8.1 Description of the ‘entries’ text file. This file is a plain text file that contains one entry (rows) per image file holding the data described above, one value per field (columns) 182

8.2 Description of the JavaScript Object Notation (JSON) data files. These files are JSON files containing one object per image file holding the data mentioned above. 183

8.3 Description of the ‘colorwords’ JSON file holding one object accessed with the key `data` which has an array of objects as described above, one object per color name. 183

8.4 Matching Matrix: image and sound descriptor equivalencies. 187

Introduction

This dissertation begins with a word, database, and a proposition: Is there something we can call database music? This sudden jump from noun to adjective comes not without its audible retaliation, and I make no attempt to muffle it. The reader would perhaps find it useful to know that my condition of composer has guided my research, and made me jump too quickly at the opportunity to make some sound in this initial gesture. Nevertheless, there is a sound and we can listen to it.

In the literature that I have mined (for this dissertation is not only a text, but the sedimented layers of text that I initially traversed with keywords in database queries, such as "`database AND music`"), the database has many histories and many names attached to it. I make no attempt to cover all of these, but I admit that I have (foolishly) tried: the subtitle does begin *"a history…."* However, this is 'a' history and not 'the' history and thus there are gaps and missing parts to which this text is inevitably bound. It is also the 'a' that accounts for the path that I begin delineating in the digital house (or database) that is this dissertation.

The two nodes that compose the focus of its text, *database and music*, have each their own historical, technical, and aesthetic idiosyncrasies. The primary goal of this dissertation (if I dare to say that it has one) is thus to find where database and music intersect. The '`AND`' in our query had indeed much less results than the `OR`, which meant my quest was already promising some reduction. For instance, I decided that the search would only pertain to situations in which computers were involved ("with, perhaps, the exception of…" 4.1). Needless to say, not only was my search dramatically expanded through the plethora of database applications, the history of their systematization, and the ongoing struggle between models, it also opened up the programming world, and with it the history of programming languages, and of the computer itself. Despite this broadening of the field, the conjugation 'database music' narrowed down to computer based music practices, leaving out music made without (and also music made before) the emergence of computers.

Upon this abysmal enterprise, I did as any musician would and started listening for the

sound of databases. I realized that this is not just the sound of your computer reading from its hard-drive: it is the sound made with its software. At this point, a network (and this is one of the key terms throughout this text) of sonic software had begun to appear. What this network pointed to, besides the key to open door number one, was a certain silence that much of the literature relating to databases in art continues to abide to: a sonic silence (a silence regarding sonic practices). This relates to the *"history, technology..."* part of the subtitle. In addressing this silence, I do not attempt to invalidate previous approaches to theorizing database art (Manovich, 2001; Vesna, 2007). On the contrary, these texts have shed light on the key concepts with which I have traversed the sonic software I present, the types of programming decisions I discuss, the various disciplines I place at the intersection of computers and music, and the plurality of shapes that have appeared in relation to databases: door number two. Through these authors, I introduce notions of embodiment, virtuality, and framing drawn from posthumanism (Hayles, 1999) and new media theory (Hansen, 2004), in order to contextualize the role of the database within the practices of Music Information Retrieval (MIR), Sonification, and Computer Music. Each one of these disciplines has its own history and it is evidenced by the many conference proceedings and journals that I have also mined, as well as the various authors (most of them composers, and most of them programmers) that I refer to. In between these two, that is, in between my exploration of the database in new media theory, and its corresponding exploration within sound practices, I introduce the more technical evolution of the database, in order to develop a secondary concept that speaks of the performativity of databases: *databasing*. I use this non-existing gerund to refer to all the actions that need to take place around databases, whether these are made by humans or not, and referring to these users as *databasers*.

After having reached this point, in which the intersection of the database and music was covered in terms of its facticity, as the evidence of a motion, the trace of the database, I could not help but noticing door number three. I attempted to connect database practices to theories of listening, memory, and networks outline a framework for discussing the aesthetic agency of

database music. I began listening to a certain sound of networks, a certain resonance. This door refers to the next part of the subtitle "...aesthetics of the database..." and opened up to the complex world of sound, flipping this text horizontally: `music AND database`. (Bang!) The meaninglessness of this reversal, at least semantically or even in terms of a database query, became precisely the point. The sound of databases that I delineate through the first two doors now reached a point where it faced a difference. I enter first with Nancy's (2007) ontology of sound, with which I understand the networks of music software as resonant networks. With this move, the database sounds as an actor (Latour, 1990) that reconfigures the way we think communities (Nancy, 1991). The implications of this association led me to distinguish databases, memory, and archives, and find within their connections a spectral form of authority (Derrida, 1978; Derrida & Prenowitz, 1995). The final move extends towards the activity of the (human) body, and the relationship between database, gender, and performance (Butler, 1988). These three nodes (resonance, spectrality, and performance) encompass an aesthetics of the database music that I plan only to suggest throughout this text, but that I will leave, to a certain extent, unexplained. [1]

Up to this point, my argument might seem to have arrived to an end: I contextualize and define database practices (chapter 1); I develop a technical overview of the performativity of the database into what I call databasing (chapter 2); I review the existing literature with emphasis on sound practices (chapter 3); I conceptualize sound in terms of resonance, networks, and community (chapter 4); I delineate the differences between database, memory, and archive, in order to present the spectrality of databases (chapter 5); I develop databasing in terms of performance, gender, and style in relation to databases (chapter 6). However, there is yet one more leap that the more

[1] The reasons behind this artistic licence within the present academic text might perhaps come suggested from my condition of composer. However, this intentional suspension of my argument towards database music is grounded on the creative potentials that would otherwise be limited by an exhaustive account of the aesthetics in question. In other words, by advancing this threefold approach (resonance, spectrality, and performance) I suggest only one —rather fragile— way in which we can understand aesthetics within database music; the moment this aesthetics is defined to the fullest extent of its possibilities, it is when our creative duty as databasers becomes anchored to the framework of its definition. Thus, by providing the reader with only some aspects of this suspended or interrupted framework I suggest the reader to continue to explore these and other uncharted territories that database music proposes, and to traverse further thresholds that are yet unknown to this composer.

adventurous reader might take with the final chapter of this dissertation. In chapter 7, I bring the discussion of database and music to the latter part of the subtitle *"in music composition."* With this chapter (a fourth door) I engage with the work of music composition. That is to say, with the history of the database in mind, I rethink the activity of composing (Vaggione, 2001), the role of the composer (G. E. Lewis, 1999), and the operativity of the music work (Cascone, 2000). The reader will be warned that this last chapter has no conclusions, let alone answers. Neither has it proper questions. It can be held as an attempt to incite, if anything, a provocation before the question.

I have already warned the reader about the aesthetic impulse of a composer writing a dissertation, but that should not discourage neither academic rigor, nor literary thirst. I have thus included an exhaustive bibliography, interludes, a transcription, and a postlude (work in progress), together with graphs, tables, a list of acronyms, a glossary, and some snippets of pseudocode and sometimes working code. In the hope that the content of this dissertation will open the discussion about the role of databases beyond their technical implementation and into their roles in musical aesthetics; and in the hope that you continue reading these p{age,ixel}s, I will (simply) ~~draft/end~~ these introductory remarks.[2]

[2]This dissertation was made using LaTeX and the Sublime Text editor. Different versions (html, pdf, and docx) and the source code can be accessed here: https://fdch.github.io/database_music

Part I

Database Art

Chapter 1

The Database In New Media Theory

1.1 Database As Form

> The world appears to us as an endless and unstructured collection of images, texts, and other data records, it is only appropriate that we will be moved to model it as a database —but it is also appropriate that we would want to develop the poetics, aesthetics, and ethics of this database. (Manovich, 2001, p. 219)

To point to the origin of the database as it is known today is not an easy task. Certainly, databases are closely related to the history of computers, but they also relate to the history of lists. The common link between these two is the fact that they are written —on a memory-card, on a page—, which would take its history to the origins of the written word.... However, there is a point where the history of storage takes an operational turn. At this point, the 'word' becomes a type of data, and data begins to bloom exponentially, impulsing faster and more efficient storage and retrieval technologies. Database systems were modelled hand-in-hand with computer languages and architectures from the late 1950s until the present day, when they continue to be developed for almost all aspects of the business world.

In the artworld of the 1990s, the increasing availability of personal desktop computers — with software suites, programming languages, and compilers— resulted in the emergence of new

media art. Lev Manovich (ibid.) was the first media historian to argue that the database became the center of the creative process in the computer age. The database had become the content and the form of the artwork in *The Language of New Media*. Furthermore, Manovich recognized that the artwork itself had become an interface to a database; an interface whose variability allowed the same content to appear in individualized narratives. Thus, he claimed that narrative and meaning in new media art had been reconfigured differently. Narrative became the trajectory through the database (ibid., p. 227), and meaning became tethered to the internal arrangement of data.[1] Therefore, for Manovich, the "ontology of the world as seen by a computer" (ibid., p. 223) was the symbiotic relationship between algorithms and data structures. As a consequence of the use of databases in art, the architecture of the computer was transferred to culture at large (ibid., p. 235). Manovich's 'database as symbolic form' thus became a technologically determined shadow that haunted much of new media.

1.2 A Semiotic Trap

In order to reveal the extent to which the presence of the database has a radical effect on narrative, however, Manovich reverses the semiotic theory of syntagm and paradigm that governed much of the 20th century (ibid., p. 231). Manovich describes the paradigm as a relation subjected to substitution, and the syntagm as a relation subjected to combination. For example, from the entire set of words in a language (the paradigm) a speaker constructs a speech (the syntagm): the paradigm is implicit (absent) and the syntagm is explicit (present). The relation between these two planes (of the paradigmatic and the syntagmatic) is established by the dependence of the latter on the former: "the two planes are linked in such a way that the syntagm cannot 'progress' except by calling successively on new units taken from the associative plane [i.e., the paradigm]" (Barthes, Lavers, &

[1] Weinbren (2007) writes that a database "does not present data: it contains data. The data must always be in an arrangement... that gives the data its meaning" (ibid., pp. 67–9).

Smith, 1968, p. 59). Barthes gave several examples with different 'systems,' one of which was the food system. Put simply, the dish (as a set of choices with which to make a dish) is the paradigm and what you are eating is the syntagm (ibid., p. 63). However, when one looks at the restaurant's 'menu', one can glance at both planes simultaneously: "[the menu] actualizes both planes: the horizontal reading of the entrées, for instance, corresponds to the [paradigm], the vertical reading of the menu corresponds to the syntagm" (ibid., p. 63).

A software menu would come to represent both planes as well: the paradigm is the set of all possible actions the user might make within the specific context of the menu; the syntagm is the actual sequence of clicks that the user makes. Manovich points to a reversal of these planes (See Figure 1.1). Given the concrete presence of the database (the options on the software menu), and given the hyperlinked quality of the user interface (those options are clickable links), the database becomes explicit (present) and the sequence of clicks becomes implicit (absent, 'dematerialised'). Since Barthes' project was focused mostly on the distinction between speech (as syntagm) and language (as paradigm), this is how the database was first understood in art. On the one hand, narrative (speech) is the syntagm: it is the trajectory through the navigational space of a database, and since this narrative is abstracted, its concreteness comes achieved by the interface, and interface and narrative depend on each other. The result is an interlocking of narrative and interface that results on the conception of the interface-as-artwork. On the other hand, the database is the paradigm (language), since it represents the set of elements to be selected by the user.

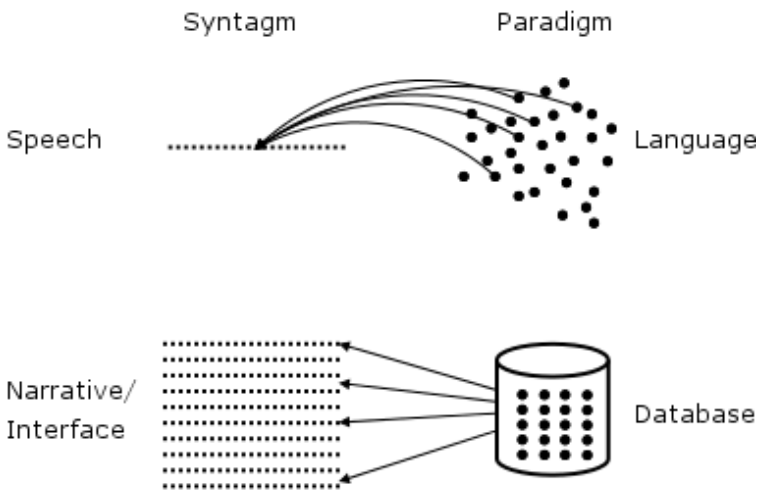


Figure 1.1: Syntagm and Paradigm reversal
Top: syntagm, paradigm, and their relation. Bottom: narrative, database, and their reversed relation.

For example, consider the case of the typical timeline-view of a video editor.[2] Normally, the user creates a session and *imports* files to working memory, creating a database of files —video files, in this case. Once this database is in working memory, the user places on a timeline the videos, cutting, and processing them at will, until an *export* or a *render* is made.[3] The timeline where the user places the videos is a visualization of the "set of links" to those files on disk, and not their actual data. Such a timeline is an editable graph that allows the user to place in time the pointers to the elements on the database. This is what Manovich means by "a set of links," because the user is not handling the files themselves —as would be the case with an analog video editor, where the user cuts and pastes the magnetic tape—, but the extremely abstract concept of memory pointers.

I consider this reversal to be valid as a shift from one-to-many to many-to-one (See Figure 1.1). The question of the materiality of the database and of the pointers depends on the materiality of data. Links or pointers have, for Manovich, a different (absent-like) status in relation to stored

[2]This example was used by Manovich in the late 1990s, and it is still valid today with most multimedia editing software.

[3]'Import,' 'export,' and 'render,' refer to processes that read from or write to the computer's disk.

memory itself. This is because of a distinction between pointers and data on the basis of their use: pointers are of a different nature since they do not store data directly. Instead, they refer to the address in memory where a specific stored data begins. However, the mutual binary condition of pointers and data, and the fact that they are both stored in the same memory, reveal Manovich's reversal to be somewhat misleading. Pointers are just another data type, however functionally different they may be. If one understands them as moving bodies, it follows that pointers are 'lighter' and travel much faster than other data types, which are 'heavier' and slower to move. However, data types are not moving bodies at all, and thinking of them as such interlocks us in a semiotic trap: accepting this reversal means accepting a certain materiality of data, which is different from a certain materiality of information.

1.3 Digital Convergence

> A mere 'byproduct' of pleasure, entertainment is a hangover from the media epoch: a function that caters to our (*soon to become obsolescent*) need for imaginary materialization through technology [which, in turn,] serves as a diversion to keep us ignorant of the operative level at which information, and hence reality, is programmed. [emphasis added] (Hansen, 2002, p. 59)

I find in Manovich a silent allegiance to german media theorist Friedrich Kittler's concept of digital convergence. Digital convergence entails that the bodily resonance of media becomes obsolete in the face of absolute digital information storage. Thusly, it turns the human into a "dependent variable" (ibid., p. 59). In the case of physical media, the human body was, for Kittler, directly shaped by media, and the limit of this 'shaping' was set by the bodily limits of perception. The body became a by-product of media. However, in the age of digital convergence, of an "absolute system of information" (ibid., p. 63), media remove this bodily limit of perception, making the human body a residual product. The body, then, becomes a residue of digital industries.

For example, the extents of this residual aspect of the human can be seen in writer Nor-

man Klein's considerations of the author (N. M. Klein, 2007). Following Manovich's interface-as-artwork, Klein argues that since the reader gets immersed in data, "[she] evolves pleasantly into the author" (ibid., p. 93). Because the reader participates in the narrative, the result is a reconfigured concept of shared authorship. However, Klein continutes "instead of an ending, the reader imagines herself about to start writing" (ibid., p. 93). This surprising twist in Klein's consideration adds another layer of complexity, namely, the categorical difference between 'writing' and 'not-yet-writing.' In Klein's sense, narrative constitutes a promise of authority that equally blurs the roles of the writer and of the reader. Most importantly, this blurred authority is seen as a reflection of control and subordination of the human. In this view, the potentiality of authority arising from the trajectory through the database belongs neither to the reader nor to the writer: it is appropriated by the database. The roles of the reader and the writer fade into each other and vanish, allowing the database to be a dominant middle term. In other words, human agency is absorbed into a shadow, making the database the sole agent to which the human is subjected. In Klein's own words, the human is a 'slave' to data, and as a consequence the human is economically 'colonized' and 'psychologically invaded' by the evolving force of computers, information, or technology in general (ibid., pp. 86–8). Authority converges, too, in the age of digital convergence.

Media theorist Mark Poster defines technological determinism as the "anxiety at the possibility of [the human mind's] diminution should these external [technological] objects rise up and threaten it" (Poster, 2011, p. X). In other words, the fear or anxiety that the human is ultimately subjected to the power of technology. Understanding new media as digital convergence leads to reading the 'new' in new media as the 'digital.' In reaction to the anxieties that this convergence brings, and from an embodied approach where databases have an aesthetic agency in resonance with the human, in what follows I propose to shift the focus from narrative (interface) to performance (databasing), and to reconfigure the shadow of the database as a hybrid skin exposing the human and the non.

1.4 Bodiless Information

> The disembodiment of information was not inevitable, any more than it is inevitable we continue to accept the idea that we are essentially informational patterns. (Hayles, 1999, p. 22)

Media theorist N. Katherine Hayles (ibid.) unearths the theoretical context of cybernetics, upon which the posthuman has been constructed throughout the 20th century. She identifies three waves of cybernetics, each governed by different concepts which helped build the undergirding structures of the technologically determined and disembodied literature in vogue in the 1990s.

The foundational wave cybernetics (from 1945 to 1960) was built, among other concepts, on two main theories: Jon von Neumann's architecture of the digital computer (See 2.1) and Claude Shannon's theory of information. As "a probability function with no dimensions, no materiality, and no necessary connection with meaning" (ibid., p. 18), Shannon's formal definition of information within communication systems highlighted pattern over randomness (ibid., p. 33). Therefore, disembodied information became a signal to be encoded, decoded, and isolated from noise.

The word 'cybernetics' [steersman] thus synthesized three central aspects: information, communication, and control. Since the human was seen as an information processing entity, it was "essentially similar to intelligent machines" (ibid., p. 7). Therefore, the conceptualization of the feedback loop as a flow of information came to put at ease notions of human subordination, thus arriving at the governing concept of first wave cybernetics: *homeostasis*. In this sense, the "ability of living organisms to maintain steady states when they are buffeted by fickle environments" (ibid., p. 8), became a patch that simultaneously fixed computers as less-than-human, but also pointed to the anxiety of disembodied information that was growing underneath.

However, since the observer of the 'feedback loop' became part of the flow of the system, in the second wave (from 1960 to 1980), cybernetitians reconfigured homeostasis into *reflexivity*, that is, "the movement whereby that which has been used to generate a system [becomes] part of

the system it generates" (ibid., p. 8). This became also known as autopoiesis (i.e., self-generation), based on writings by Humberto Maturana and Francisco Varela. This second wave leaves the feedback loop behind, since it considers that "systems are informationally closed" (ibid., p. 10). This means that elements in the system do not see beyond their limits, and the only relation to the 'outside' environment is by the concept of a *trigger*. In this sense, disembodied information was buried deeply into the organization of the system, and the system itself appeared in the form of a cyborg.

Shifting from triggers to artificial intelligence signaled the third wave of cybernetics (from 1980 onwards), whose central concept was *virtuality*. Development of cellular automata, genetic algorithms, and principally, emergence, led to the formation of the posthuman, or an embodied virtuality. However, in Hayles view, the underlying premise of this 'posthuman' is that the human can be articulated by means of intelligent machines (ibid., pp. 17–8). In turn, reconfiguring the concepts of body, consciousness, and technology as inherent to (post-) human life, Hayles argues for the impossibility of artificial intelligence to serve as a proxy for the human. Hayles objective is, then, to dismantle cybernetics from its (relative) assumptions, questioning its major achievements over the years and thereby opening the field for new considerations of the body and its material environment within cybernetics, and by extension, of the body in new media:

> My dream is a version of the posthuman that embraces the possibilities of information technologies without being seduced by fantasies of unlimited power and disembodied immortality, that recognizes and celebrates finitude as a condition of human being, and that understands human life is embedded in a material world of great complexity, one on which we depend for our continued survival. (ibid., p. 5)

While her work is focused on the literary narratives that were built in parallel with cybernetics, she leaves incursions in new media theory for other media theorists. This is where Mark B. N. Hansen comes in.

1.5 Embodying Databasing

As I describe above, Manovich arrives at this notion of the interface-as-artwork by opposing database and narrative on the semiotic grounds of the reversal of the paradigm and syntagm. In turn, media theorist Mark B. N. Hansen (Hansen, 2004) notes that the interface-as-artwork constitutes a disembodied "image-interface" to information in which the process of information itself (in-formation; giving form) is overlooked. Hansen locates the source of this disembodied conception in Manovich's implicit —but nonetheless evident— premise of the overarching dominance of cinema in contemporary culture, which results in a "disturbing linearity [with] hints of technical determinism" (ibid., p. 36).

For example, Manovich argues that standardization processes originating from the Industrial Revolution have shaped how cinema is produced and received. Attuned to the perceptual limits of the body, the standardization of resolution can be seen (image dimensions, frames per second, and aspect ratio) and heard (audio bit depth, sampling rate, and number of channels). In this sense, the moviegoer and by extension, the listener became industrial by-products, determined by the massively produced electronic devices used for recording and playing. As I have described with Kittler's technological determinism, the devices driven by industrial forces, therefore shaped the body, and as an extension, the aesthetics of cinema.

For Manovich, due to the internal role of the database, the logic of new media is no longer that of the factory but that of the interface. Through the interface to a database, the user is given access to multiplicities of narrative, and thusly, to endless information. The user is granted the power of the database, making in Manovich's eyes the database an icon of postmodern art. In other words, on an aesthetic level, while mass-standardization and reproducibility of media —the "logic of the factory" (Manovich, 2001, p. 30)— shaped the form of cinema, post-industrial society and its logic of individual customization, shaped the database form. At the bodily level, cinema standardized perception of the passive body, and database individualizes experience. However, this

individualized experience still constitutes a technological 'shaping' of the body, a shaping that is exploded into every user quietly sitting behind the screen.

In opposition to this passivity of the body, Hansen describes images as something that emerges out of the complex relationship between the body and some sort of sensory stimulus. In radical disagreement with Manovich, Hansen considers that the image has become a process which gives form to information, and that this process needs to be understood in terms of the body as a filtering and creative agent in its construction. Drawing from Henri Bergson's theory of perception, and in resonance with cognitive science, Hansen defines the function of the body as a filtering apparatus. Under this conception, the body acts on and creates images by subtracting "from the universe of images" (Hansen, 2004, p. 3). Image creation is world creation, and it is not necessarily in contact with the reality that surrounds the body (or the reality of the body), but it is a result of the embodiment of a virtuality that is inherent to our senses. In other words, through this filtering activity, the body is empowered with "strongly creative capacities" (ibid., p. 4). The world is a virtuality that is constructed with our senses and our body. The world can only appear if it appears to the body. Therefore, instead of being a passive node, the body actively *in-forms* data as information (Hansen's word play). The databaser (database user) makes information out of data by precisely embodying the performative act that I call databasing.

1.6 Filtering And Framing

> The activity in the receiver's internal structure generates symbolic structures that serve to frame stimuli and thus to *in-form* information: this activity converts regularities in the flux of stimuli into *patterns* of information. (Hansen, 2002, p. 76)

The activity of framing, according to Hansen, must be differentiated from that of observation. In this way, "information remains meaningless in the absence of a (human) framer," (ibid., p. 77) and framing becomes a resonance of the (bodily) singularity of the receiver. Quoting MacKay's *Information, Mechanism, Meaning* (1969), the meaning of a message

> . . . can be fully represented only in terms of the full basic-symbol complex defined by all the elementary responses evoked. These may include visceral responses and hormonal secretions and what have you. . . an organism probably includes in its elementary conceptual alphabet (its catalogue of basic symbols) all the elementary internal acts of response to the environment which have acquired a sufficiently high probabilistic status, and not merely those for which verbal projections have been found. (ibid., p. 78)

It is with this conception of framing that Hansen describes precisely that information always requires a frame:

> . . . this framing function is ultimately correlated with the meaning-constituting and actualizing capacity of (human) embodiment. . . the digital image, precisely because it explodes the (cinematic) frame, can be said to expose the dependence of this frame (and all other media-supported or technically embodied frames) on the framing activity of the human organism. (ibid., pp. 89–90)

Therefore, in the context of Kittler's digital convergence, framing prevents the human from being rendered a dependent variable. On the contrary, the framing function of the human body allows the digital to become information. The frame, as Hansen describes, is the human body filtering images from the world, and creating a virtual image that gives form to data. The frame needs to happen as a relation, and thus, it is the temporal instantiation of a process. What would a human body without this framing and filtering capability look like? How would this temporality of the process of information be understood?

In the following interlude I take the concept of an absolute (human) memory to be at an intersection between disembodied theories of information and, precisely, the concept of an embodied memory. The aim is to introduce and differentiate between human and nonhuman in terms of memory and databases. The wonder, admiration, but also the fear and mystery that a notion of embodied memory awakens can speak for the uncanny feeling that occurs whenever databases are involved, and thus can speak for a certain agency of the database. I understand this feeling as what accounts for the aesthetic experience of database music.

1.7 Interlude: An Embodied Database

> I suspect, nevertheless, that he was not very capable of thought. To think is to forget differences... (Borges, 1942, p. 2)

The importance of memory —and forgetfulness— can be represented by Jorge Luis Borges's famous 1942 short story, *Funes, the memorious* (ibid.). Due to an unfortunate accident, the young Irineo Funes was —"blessed or cursed" as Hayles points out (Hayles, 1993, p. 156)— with an ability to "remember every sensation and thought in all its particularity and uniqueness " (ibid.). A blessing, since a capacity to remember with great detail is certainly a virtue and a useful resource for life in general; a curse, because he was unable to forget and, as a consequence, he was unable to think, to dream, to imagine. Throughout the years, he became condemned to absolute memory, and so to its consequence, insomnia:[4] he was secluded in a dark and enclosed space so as not to perceive the world. Hayles focuses on one aspect of the story, namely, the fact that Funes invented —and begun performing— the infinite task of naming all integers, that is, of giving a unique name —and sometimes, last name— to each number without any sequential reference. According to how Hayles describes it, by carrying out his number scheme, Funes epitomizes the impossibilities that disembodiment brings forth. As Hayles writes, "if embodiment could be articulated separately from the body ... it would be like Funes's numbers, *a froth of discrete utterances registering the continuous and infinite play of difference*" [emphasis added] (ibid., pp. 156–159). The point that Hayles touches upon can be seen as the limits and fragility of embodied memory, as well as the need to forget, in opposition to an embodiment 'outside' the body (disembodiment), that would require no need to forget. We will see how the difference between forgetting and erasing relates to the database. In that Manovichian world which "appears to us as an endless and unstructured collection of images, texts, and other data records" (Manovich, 2001, p. 219), this idea would be perfectly viable. Indeed, data banks have already been growing exponentially much

[4] In the prologue to *Ficciones*, Borges writes that this story is a long metaphor of insomnia: "Una larga metáfora del insomnio" (Oviedo, 2019).

in the same way as Funes' memory. This capability of accumulation without the need for erasure is enabled by the database structure inherent in computers. However, the distinction that Hayles presents is crucial: data is not information because information needs to be embodied, and with that embodiment comes the need to forget. In sum, on one hand, a disembodied data bank can have all the uniqueness and difference that is available by the sum of all cloud computing and storage to date; however, on the other hand, embodied memory is available by the human capacity to forget.

Matías Borg Oviedo (Oviedo, 2019) relates this incapacity for thought to the negation of narrativity itself, thus finding in the image of Funes a hyperbole for contemporary subjectivity (ibid., p. 5), where there is no room for narration, only data accumulation. In this sense, narrativity can be seen as that which resides in the threshold between knowledge and storage. I believe this distinction stems precisely from the difference between information and data. The process of information, of giving form, requires a certain temporality that is not that of the perceptually immediate and extremely operative zero-time of the Central Processing Unit (CPU). Within the zero-time of computer operations (within a millisecond) there simply is no *time* for narration, only for addition or increment. Narrative is temporal —happening as a historical process— and algorithms are atemporal —operating in an constant now. Since neither data structures nor algorithms operate outside the confines of the millisecond, they can't spare time to think and, likewise, they can't forget to count. Counting is all they do, so they cannot tell stories: the difference in the same spanish word *contar números* [to count numbers] and *contar historias* [to tell stories]; or, "if a German pun may be allowed: *zählen* (counting) instead of *erzählen* (narrating)" Ernst (2013, p. 128). Therefore, Funes' accumulative memory represents the overflow of the now, the totally blinding transparency of the world, and an absolute memory that precludes narration. Because he accumulates data of the world in its totality, he does not have time to think: "to think is to forget differences" reads the quote above. The only one in Borges' story who actually thinks is the narrator, to which we can add: Funes could not have told this story himself in first person narrative.[5] Thus, we can ask

[5]It is worth noting how Oviedo finds in Funes a premonitory 'antithesis of the writer' himself: the latter (blind)

ourselves if this hyperbolic 'light' of the Funesian 'absolute memory' is not a premonitory figure of the database.[6] Because of this antithetical condition between databases and narrative, Manovich proposed that the database became a form on its own, in opposition to narration. To the extent that 'database form' as a category that identifies art using databases, this can be considered accurate. However, considering this distinction between embodied and disembodied memory that resides in the ability to narrate, databases are inherently deprived of narration. Making art with databases is making them dance (see below). Therefore, to what extent is 'database music' in itself a contradiction if we consider 'music' to be a form of writing? I will leave this discussion for a later chapter (See 4).

This Funesian database can also be understood in relation to what Gayatri Chakravorty Spivak (Derrida & Spivak, 1976) writes about forgetfulness. She notes in Nietzsche the 'joyful' and 'affirmative' activity that constitutes forgetfulness as being twofold. On the one hand, this activity is a "limitation that protects the human being from the blinding light of an absolute historical memory," and on the other, it is "to avoid falling into the trap of 'historical knowledge"' (ibid., p. xxxi). The 'historical' here is an "unquestioned villain," which takes two forms: one "academic and preservative," the other "philosophical and destructive" (ibid., p. xxxi). For Nietzsche, as Spivak notes, forgetfulness is a choice that comes as a solution: an "antidote" to the "historical fever," or the "unhistorical," that is, "the power, the art of forgetting…" (ibid., p. xxxi). I propose an imaginary experiment that would add some noise to Borges' short story, however utterly fantastic his writing was. One thing that can be read from the story is that, in order to seclude himself from perceiving the world, or better, in order to forget the world altogether, Irineo stayed in the darkness of his room. This is how he cancelled light, a quite powerful stimulus if memory-space is to be optimized for the purpose of, say, getting some sleep. However, there is little to no mention of the sonic environment in which Funes was embedded —somewhere in the outskirts of the quiet

Borges could find himself immersed in a constant flow of narrativity (ibid.).

[6] In this sense, Funes is a bit like an Oracle —pun intended with the other Oracle Corporation (Oracle)—, with absolute knowledge of the past and the future, but with no time to think…

Uruguayan city of Fray Bentos. In fact, the only sonic references are focused on the narrator's perspective, referring to Funes' high-pitched and —due to his being in the darkness— acousmatic voice. To a certain extent, we might think of Funes' high-pitched (at least this is how the narrator heard it) voice as a hint to the highlighted overtones that link Borges' "long metaphor of insomnia" with "the 'laughter' of [Nietzsche's] Over-man [that] will not be a 'memorial or... guard of the... form of the house and the truth... He will dance, outside of the house, this... active forgetfulness" (ibid., p. xxxii). Nonetheless, Funes is deprived of this forgetfulness, and thus cannot go outside, let alone laugh or dance.[7] By locking himself inside a room he would have managed to attenuate sound waves coming in from outside. Notwithstanding his isolation —a house arrest—, sound waves are actually very difficult to cancel.

(It might be useful to compare Funes' attempt to filter out the world with John Cage's quest for silence. An interesting experiment would have been to have John Cage take Irineo to an anechoic chamber and ask him what he can remember then. From Cage's own experience, we can guess that Funes would effectively remember his own sounding body. Kim Cascone (2000) writes that "[Cage's] experience in an anechoic chamber at Harvard University prior to composing 4'33" shattered the belief that silence was obtainable and revealed that the state of 'nothing' was a condition filled with everything we filtered out" (ibid., p. 14). Imagine an 80 year-old Irineo in David Tudor's premiere at Maverick Concert Hall in Woodstock, NY, infinitely listening to 4'33")

It is very unlikely —but nonetheless possible— that Borges was aware of American acoustician Leo Beranek's research for the US Army during World War II, that is, when the first anechoic chamber was built.[8] Furthermore, even if he managed to isolate himself completely from the world by cancelling perception altogether, Funes would have been with his memories (he was not deprived of *anamnesis*, the ability to remember), which were not discrete, but continuous itera-

[7] The acousmatic quality of Funes' voice will not be touched here, but it is indeed a good point of departure for another time.

[8] https://en.wikipedia.org/wiki/Leo_Beranek

tions of the world he had accumulated over the years. What this means is that all the sounds he had listened to would be available to his imagination. As far as we can learn from the narrator, while *smell* is referenced to in the story, *sound* was nonetheless out of Funes' concerns. Therefore, one thing we can ask ourselves is how the world would sound for Irineo Funes? The task is not difficult to imagine: the world would be inscribed in poor Irineo's memory in such an infinitely continuous way that each fraction of wave oscillation would be different, unique, leaving no space for repetition of any kind. For example, one of Irineo's concerns was to reduce the amount of memories on a single day, which he downsized to about seventy thousand.... What would be Funes' sample rate? What frequencies could he be able to synthesize? All sounds (and all that can be registered) would be listened to completely, with every infinitesimal oscillation of a wave pointing to the most utterly complete scope of imaginable references. A complete state of listening. In fact, we might not be able to call it listening any more. Not even signal processing. An infinitesimal incorporation of sound is unthinkable. Within such total listening there would be no possibility for thought, no processing of any kind, and no synthesis: only infinite accumulation and storage. On the one hand, no matter how accurate our embodied listening might be, we are bound to miss some motion, some waves would pass through us and we would be busy forgetting to register them. On the other hand, if such a recording were humanly possible, thinking would cease to be so. This can be thought of as the intersection of the finite with the infinite: while Funes' nonhuman memory corresponds to a dynamics of the infinite, his human body is quite human. Funes is not deprived of this finitude, and existed (fantastically) on a ghostly liminality. This liminality grows more and more evidently throughout the story, hand in hand with the cumulative growth of the Funesian database, all the way until the end, in a sort of Moore's Law of data congestion, saturating completely in an utterly human pulmonary congestion (Oviedo, 2019).

1.8 Closing Remarks

Despite Manovich's technologically determined considerations of the database as form, he notes a fundamental aspect of the use of the database when he expresses that data need to be collected, generated, organized, created, etc: "Texts need to be written, photographs need to be taken, video and audio need to be recorded. Or they need to be digitized [and then] cleaned up, organized, and indexed" (Manovich, 2001, p. 224). In this sense, he begins to describe the actions that need to be performed around data, what I call databasing (See 2.1), which connotes the use of databases in terms of their performativity (See 2.1 and 6). He even goes further and proposes that this activity has become a "new cultural algorithm," (ibid., p. 225) (See Figure 1.2).

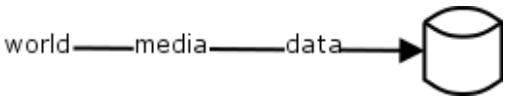


Figure 1.2: Manovich's cultural algorithm
The world is mediatized, stored in some media (film, tape), then digitized into data, then structured into a database. The result is the world represented by the database.

While Manovich calls for an "info-aesthetics" (ibid., p. 217), as well as a poetics, and ethics of the database, neither Manovich nor the following generation of media artists and theorists could carry out an exhaustive account of an aesthetics of the database. Several authors continue to abide by Manovich's claim that the aesthetics of the database, or the database as form, is a symptom of the uncritical use of database logic throughout the visual art world of the 1990s. It is in hindsight that his argument can be understood as grounded on the same disembodied constructions that prevent him from including human agency in his account. In any case, his contribution to the literature on the role of databases in new media have led us to this point of inflexion, in which we can consider different points of view regarding the topic of databases. My revision of this algorithm will come later in this dissertation (See Figure 4.2). In what follows, I will explore the more technical aspects of databasing, in order to trace a connection between the literature on sound-based computer practices with that of the development of databases over the years. In this

way, I bring the discussion of databases into the sonic sphere.

Chapter 2

Databasing And The History Of Databases

2.1 Databasing: The Performance Of The Database

> The first step in working with a database is the collection and assembly of the data.... Sorting determines the sequence of presentation, while filtering gives rules for admission into the set presented [,] resulting in a database that is a subset of the "shot material" database. Editing is selecting from the database and sequencing the selections.... To go further: for a filmmaker the term "cutting," as "editing," loses its meaning, and "sorting," "assembling," and "mapping" become more apt metaphors for the activity of composition. (Weinbren, 2007, p. 71)

Like Manovich, Weinbren finds a redefinition in filmmaking that stems from the selection processes that the database calls for: data collection, generation, and assembly. Weinbren further breaks the selection process into sorting and filtering. With this new terminology, Weinbren makes a linguistic shift from 'editing' and 'cutting,' to 'sorting,' 'assembling' and 'mapping.' This linguistic shift is significant in the sense that it highlights the practice that is 'under' the filmmaker: databasing.

Databasing is a term I have chosen that best describes the practice of the database, that is, a term that includes the elements and actions of database practices, together with their temporality. The elements of databasing are the different data types and structures that build more

complex database systems. The actions of databasing are, on the one hand, the type of operations that a database allows, and on the other, the bodily activity that occur before and after these operations. That is to say, since the operational level occurs below the perceptual threshold of the body, I consider the actions surrounding the immediacy of computations to be defining aspects of databasing.

2.1.1 Data types and structures

Depending on the programming language, data types may or may not be part of a data structure, and they store different types of values such as `int, float, char`. These types are then interpreted in binary language by the compiler. Grouping these types into larger sets results in `arrays`. For example, in the C programming language, programmers 'declare' variables first —e.g., `unsigned char age`— and then 'initialize' them with some data —e.g., `age=30`. A simple variable like one's 'age' needs only one value, and given that the `unsigned char` data type only stores values from 0-255, it is safe to use in this case: no age can be negative, no human can live longer than 255 years.

A data structure is a set of data types kept generally in contiguous slots in memory space. It is built for fast allocation and retrieval. A very simple data structure can be thought of as, for example, a person's name together with an age (See Listing 2.1).

```
typedef struct Person {
        unsigned char age;
        char name[128];
} Person;
```

Listing 2.1: An example of a data structure in the programming language C. It is named `Person`, and it holds two variables: `age` and `name`, respectively a positive integer and a string of up to 128 characters.

2.1.2 Temporality of Databasing

At this point it is important to refer to the higher or lower levels of computer software. A software that is 'higher' means that its simplest operations are composed of multiple smaller operations. The user can thus 'forget' about certain complexities that come from low-level programs, such as memory management. In this sense, low-level programs operate 'closer' to hardware, and programmers need to work at a more granular level. While the above data structure contains low-level features such as setting the size of the name array, it releases the programmer from thinking binary conversion. This means that unless you are changing values directly on the memory card (which is unthinkable), there will most likely be an underpinning software layer.

The speed of regular house computers is so fast that high-level operations happen below the perceptual level (generally below 1-2 milliseconds), hence, for example, the capability for real-time audio processing at high quality sample rates. Therefore, the temporality of activity before and after potentially very large computations feels almost immediate. This means that the body continues almost as if nothing had happened besides a click, or besides the pressing of a key. The immediacy of computation is a feature, certainly, for arriving at extremely fast operations in no time (or zero-time). It is what feels like 'magic' around computers: ask a computer to count to a 1000, and it already has....

However, it may become a bug if we consider the computer as a tool to understand the world. As Manovich claimed, the world understood with computers is not only one that is presented in binary terms, it is one constructed upon a specific set of data structures with their set of algorithmic rules. The better and more efficient the data structure is, the better and faster the algorithm. In this light, it can be argued that software development is essentially data structure development. At every software release, the software becomes more efficient, using less or more restricted memory space, etc., affecting the scope of its functionality as well as the speed at which it runs. Glancing at the evolution of software in terms of data structure efficiency, therefore, is glanc-

ing at a constantly accelerating stream of bits. Because it is immediate, software is incorporated immediately, thus narrowing the temporal window for framing.

This is why the temporality of databasing is context-dependent. As Hansen pointed out, the world can only appear if it appears to the body (See 1.5). Data structures, therefore, are very efficient storage devices that have no relation to worlds in themselves, but that are the condition for the possibility of world creating with computers. In this way, the programmer feeds into the computer a notion of world that is then returned by the computer's performance. In each data structure there is a result of a feedback network. One one hand, this network refers to the history of software development, in the sense that each software release is a instance of the much larger event that is software in general. On the other, the network links this history with the practice at hand for which the software is being designed. The sound of a computer music oscillator, for example, even if it were programmed today from scratch, would have embedded histories of computer software design, computer music history, etc.

What is important to note here, is that these interrelations of what is *already there* in software development can be thought of as resonances colliding their way into stability; a stability that emerges not only as a 'stable release' of the code, but also as the condensed multiplicity of worlds that is displaced into a software package. Therefore, data structures are world-making and world-revealing devices that engage with our own capacity for virtuality, and thus they are nodes in our world-making networks.

2.1.3 Databasing and Writing

As with other new media, the terminology used to describe computer memory is often borrowed from earlier media practices like printed text: reading, writing, and erasing. Besides terminology, computer memory shares with writing the property of hypomnesis, that is, of displacing the role of human memory with an external nonhuman device. In the case of the computer memory however,

the scale of this displacement is extremely large, in terms of data storage amount and speed. The 40-bit long 4000 numbers that Von Neumann et al were aiming at for their memory 'organ' — which was more than plenty for the computational purposes required at the time— amounts to about 16 Kilobytes, something which might seem insignificant in comparison to current computer storage capabilities found in cloud computing. In light of this fact, we might ask ourselves how is human work transformed through interaction with these massive external memories? When designing computer software for art, the way in which data is structured, together with the speed and design of data flow, has significant effects on the temporality of art altogether as a practice both from the practicioner's perspective and from the perceived result.

Programming decisions geared towards software production are grounded on the possibilities that the combinations of data structures allow. However, the programmer can design and change these structures, and each new design has direct effects on the way in which the user behaves around the software. For example, effective data structures will enable faster computations, and thus the user's experience of time with the software will be different. In other words, a click of the mouse might condense a multiplicity of actions within an instant, and this condensation accelerates thought processes and bodily actions. Given the hypomnesic quality of data structures together with the infinite potential of expansion that computer memory enables, these actions of the body have the possibility to be at once analyzed and projected towards future activity. In this sense, as users, the next 'click' or the next movement of the body might already be suggested by the predictive analysis of a given database or by the tendency of a certain dataset. When these clicks constitute the production of an art work, then, programming decisions have an effect on the result, particularly in terms of style. In fact, much of the programming concerns in computer music software during the 1980s are grounded on a certain search for stylistic neutrality (See 3.3). Certain programming decisions are audible as some form of timbral quality of sound making software that, as listeners, we can identify a work of music as being composed in one specific software. For example, when we listen to music and recognize a certain 'sound' of a software, that is, the 'sound'

of Pure Data or of Csound, it is reasonable to advance that this 'sound' proposes a certain aesthetic quality to the work that comes as a contingency of the programming decisions of the software itself, as well as of the use of the software. The flexibility, however, that comes with open source software extends the limits of applicability of these structures and of the aesthetic possibilities that would emerge henceforht. I comment on this effect of programming decisions in 3.2.4.

I have proposed that memory —with its storing of instructions and information— is what enables the computer as such. The simplicity of this synthesis of data and command in Von Neumann's architecture led to its implementation in not only the mainframe computer for which he had intended, also the regular computer as we know it today. Without this architecture, computers would only be able to perform very simple arithmetic operations (like pocket calculators). That is to say, without the computer's ability to store data (the memory organ), the partial differential equations that Von Neumann was aiming at solving would not have been possible. In these equations, the next value of the solution depends on the present value. Therefore, when iterating through every step of the solution, the function in charge of solving the equation needs to access the present value, change it, output the next value, and finally update the present value with the outputted result (See 2.2). Therefore, in order to provide such solutions, Neumann proposed that: "not only must the memory have sufficient room to store these intermediary data but there must be provision whereby these data can later be removed" (von Neumann & Burks, 1946, p. 3).

```
present ← 0
next ← 0
iteration {
        output ← next ← function() ← present
        present ← next
}
```

Listing 2.2: Pseudocode showing a routine whose next value depends on the present value.

2.1.4 The Von Neumann Architecture

> Inasmuch as the completed device will be a general-purpose computing machine it should contain certain main organs relating to arithmetic, memory-storage, control and connection with the human operator. It is intended that the machine be fully automatic in character, i.e. independent of the human operator after the computation starts (ibid., p. 1).

Data structures are the turning point of the history of the database. Their appearance enabled the performance of automated algorithms. Within the history of computer technology, data structures begin to appear since Jon Von Neumann's designs of the computer architecture (ibid.). Von Neumann and his team implemented Alan Turing's original concept for a general-purpose computing machine. Of the "certain main organs," memory-storage enables the computer's architecture as we know it today. On one hand, the storage unit of the computer allows data to be written and erased in different locations and times. On the other, the stored data can be not only values to be used during computation, but also includes the algorithmia itself, that is, the commands — functions, operations, routines, etc.— which are used to access and process data for computation. Thus, the interaction of data and command is what defines data flow inside the computer.

Consider, for example, how curator Christiane Paul describes the database as a "computerized record-keeping system," that is, "essentially a structured collection of data that stands in the tradition of "data containers" such as a book, a library, an archive" (Paul, 2007, p. 95). However, when Paul suggests that databases are simply an instance of data collection this only points to the passivity of the container, and not to the potential that it has. An good analogy would thus be a book with the capacity to read itself, if reading were going through every letter in an orderly fashion. A database can also be understood as a library with no need for librarians because all queries are immediate; or, an archive without archeion. These considerations will be developed in the next chapter. While the more general practices of collecting and classifying data are part of the practice of databasing, on some level of the computer architecture, databasing comprises data flow within the Von Neumann architecture. This fact marks a distinction that is better seen in relation to

networks. Extending computers via networks like the Internet makes databasing a global activity that expands and changes with every user. This is why I propose that databasing reconfigures the passivity of data containers such as books, libraries, and archives, with a powerful agency that resonates aesthetically.

In order to understand how databases have changed the way we think of earlier types of containers, we need to revise the differences between database models in time. By doing this, I plan to reconfigure the notion of a database system. In general, database systems have been used in businesses, namely for administration and transaction. However, narrowing database systems this way raises the similarities or differences between systems to the level of the interface. I propose to delve into the structures of the models to find how the computer itself can be thought of as a database tree, and databasing can be thought of as the activity around databases, or simply: *database performance*. The main purpose of the following account is to understand how computer-based sound practices have participated as a particularly resonant branch of the database tree.

2.2 A Database Tree

The common use of the word 'database' within computer science came around the 1960s, when computers became available to companies throughout the United States of America. For the purpose of data processing, software developers began designing Database Managemet System (DBMS), which are still used in great demand by multiple contemporary companies. The computer's capability for data processing and storage is inherent in the constitution of database systems. In fields such as Computer Aided Composition (CAC), working with computers means being part of a system. The human operator has been regarded, for example, as a co-operator (Mathews, 1963). A further approach understands humans operating with computers as another component of complex systems (Vaggione, 2001). In this section, I describe the different levels of database systems as a tree (See Figure 2.1), starting from basic data structures to more elaborate database

systems, and then present a brief history of how databases were designed.

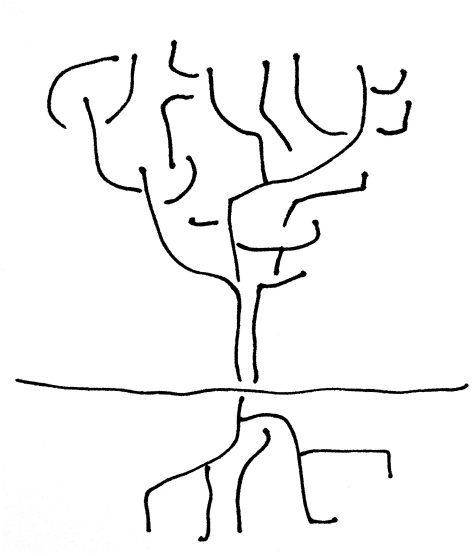


Figure 2.1: A Database Tree
A very simple sketch of a tree representing the database tree of computer evolution

Soil The tree is built on different interpretations of the Von Neumann architecture. That is to say, while this architecture went through several optimizations over the years, its three central aspects remained. Therefore, despite the fact that different industry standards for hardware construction resulted in different kinds of operating systems, the core elements of the architecture remained the same: memory (for data and program/code), central processing unit, and input/output interfaces.

Roots The below-the-soil level is accessed through machine and assembly code, which constitutes the core of low-level programming languages and are, to a certain extent, humanly unreadable: the world of bits. Above the soil, readability by humans is the main feature.

Macros The database tree metaphor relates to the concept of portability. The database tree only takes the form of a tree once it is instantiated as a software and it is run. That is to say, the database tree unfolds every time it is opened, and in this unfolding it emerges the possibility of dynamically

adapting to different soils. This is what is known in the programming world as defining conditions or macros. With these definitions, their programs can compile with different compilers, across a variety of hardwares and operating systems. Therefore, these database trees have as their main feature the capacity to unfold their roots in different directions upon demand.

Trunk The trunk of the tree is composed of data types and structures that provide flow between stored (underground) data and the above-ground components. Programming languages handle data types differently, but in essence, data types and structures are usually built in layers going from the lowest (close to roots) to highest levels.

Branches These language layers, after they reach a certain level of complexity, begin to form boughs or limbs that, while being separated from each other, are linked to the same trunk and roots. I consider branches to be programs with text-based interfaces such as Bash, C, C++, python, Java, etc. Their feature is their generic functionality.

Twigs More complex programs built on top of branches, such as Pure Data, Supercollider, R, octave, Processing, OpenFrameworks etc., are dedicated for a narrower scope of tasks. Their feature is their level of specialization for the task at hand: sound synthesis, statistics, visuals, etc. They might be more application-specific. In general, these programs are commonly considered layers on top of other languages, libraries, or software frameworks.

Leaves User interfaces (or GUIs) are the leaves of the tree. I relate the photosynthetic quality of leaves with user input/output interaction. Despite their simple, user-friendly appearance, software leaves are highly complex systems such as multimedia editors (Adobe Creative Suite or Microsoft Office), Internet browsers, mobile apps, etc. A particular kind of leave is the DBMS, generally used in businesses for data processing and editing, for example: MySQL, PostgreSQL, Non or Not only SQL (NoSQL), Appache Couch Database (CouchDB) and MongoDB.

Networks An important feature of database trees is their network capabilities. Networks can be established by connecting leaves, branches, or roots with each other, both within the same tree and with other trees. For example, software can establish a network between its graphical interface and its core program —as is the case with Pure Data, for example. Another example would be the way in which DBMSs interact with data: the MySQL database model allows the user to load a data set in working memory, and establishes a connection between the opened memory and the input/output mechanisms. Networks of trees are data streams running by way of an Internet Protocol (IP) and a client-server type of relation. Cloud storage services such as Google Drive, ICloud, OneDrive, and Dropbox are used as a networked way to store and share data. One tree can serve as data storage and processing repository, and other client trees can connect to the server tree and request data or processing of data from it. This is the essence of the internet and all the communication services that it enables, such as email services, social networking sites, and multi-user collaboration platforms like Github. This allows software like Pure Data and MySQL to have their respective core program and data sets in one computer, and their interfaces on a different one.

Clouds Combining networked databases with computer clusters forms what is known as cloud computing. For example, most universities provide clusters for data processing —e.g., NYU's Prince cluster— that can be accessed from remote locations. These clusters are massive server architectures made out of multiple processing and memory units joined together. These architectures began developing in the 1990s, coining terms like data mining (Kamde & Algur, 2011), data warehouses, data repositories (Silberschatz, Stonebraker, & Ullman, 1995).

2.3 The Realm Of Data Structures

Data structures are the building blocks upon which the entire database model is designed. A data structure is a way to organize data so that a set of element operations are possible, such as `ADD`,

`REMOVE`, `GET`, `SET`, `FIND`, etc. Data structures can be thought of in two ways: either implemented or as interfaces, what is also known as *abstract data types*:

> An interface tells us nothing about how the data structure implements these operations; it only provides a list of supported operations along with specifications about what types of arguments each operation accepts and the value returned by each operation. (Morin, 2019, p. 18)

In other words, the abstract data type represents the idea of the structure. When abstract data types are implemented in code, the speed and efficiency of the data structure can be physically evaluated. An implementation of this sort includes "the internal representation of the data structure as well as the definitions of the algorithms that implement the operations supported by the data structure" (ibid., p. 18). Because of the consequences that design has on computational performance, data structures have constituted a focal research point in the database and computer science communities.

Array data structure Arrays constitute one of the oldest and most basic data structures. They are contiguously stored, same-type data elements referenced to by indices. Most programming languages have implemented arrays. Most real-time software loads sound files or images to working memory as an array (or a buffer) of contiguous samples or pixels. Arrays are use less resources when reading than when writing, since accessing their elements is achieved by pointers, but editing demands copying large portions of the array back and forth.

Linked Lists One important technical shift in the use of data structures came with the concept of linked lists. A linked list is collection of data (usually a symbol table), with pointers to the 'previous' and/or 'next' item on the list. They are built to maintain an ordered sequence of elements. This functionality was only available after the FORTRAN '77 programming language (1977) and later it became integrated in the C programming language (Kernighan, 1978). They differ from arrays since they can hold multiple data types (including arrays and other data structures), and

they are accessed by traversing the list using the 'previous' and 'next' pointers. In the programs developed during the Structured Sound Synthesis Project (SSSP) and Computer Assisted Music Project (CAMP) years (See 3.3.1), linked lists were used in the (then) very recent C programming language. Ames (1985) as well as Rowe (1992) used linked lists, the former to represent melodies within an automated composition system, the latter within the `Event` data structures of the interactive music system *Cypher*.

Sequences Crowley (1998) claims, however, that neither linked lists nor arrays are suitable for large text sequences, since linked lists take up too much memory, and arrays are slow because they require too much data movement. Nonetheless, he argues, "they provide useful base cases on which to build more complex sequence data structures" (ibid.). In fact, data structures are generally built from arrays and linked lists. For example, in designing *Audacity*, Mazzoni and Dannenberg (2001) implemented the concept of sequences, into a set of small arrays whose pointers were traversed in a linked list. Large audio files were loaded and edited at very fast processing times.

2.4 A Brief History Of Database Models

I propose now to extend the concept of *abstract data types* to the concept of database *models*. Database models are the realm of data structures. These models, to be described below, constitute the abstract ways in which data can be organized within a database system. Database Managemet System (DBMS)s, in turn, are a specific type of software aimed at organizations, website design, server architectures, company management, among other uses in the business sector. Since an analysis of these systems falls outside the scope of this study, I provide a glimpse of the structure of the models without entering into their implementation. Figure 2.1 shows a development timeline that serves as a context for the appearance of these models. Their emergence over the years goes hand in hand with hardware and programming language development. Further, several imple-

mentations of these models depended on specific language development such as Data Definition Language (DDL) for structural specification of data, and a Data Manipulation Language (DML) for accessing and updating data (Abiteboul, Hull, & Vianu, 1995, p. 4).

Angles and Gutierrez (2008) name the three most important aspects a database model should address: "a set of data structure types, a set of operators or inference rules, and a set of integrity rules" (ibid., p. 2). Operators can be understood as the set of routines that constitute the query language and data manipulation. Integrity rules can be understood as data constraints preventing redundancy or inconsistencies, and checking routines preventing false queries. In a similar way, for Abiteboul et al. (1995) a database model "provides the means for specifying particular data structures, for constraining the data sets associated with these structures, and for manipulating the data" (ibid., p. 28). However, data manipulation (operators) and constraints (integrity) are built around the data structure, which is why, Angles and Gutierrez (2008) continue, "several proposals for [database] models only define the data structures, sometimes omitting operators and/or integrity rules" (ibid., p. 2).

In essence, all DBMSs share the same function: provide access to a database. This access, however, is restricted by the imperatives of the model. Database models have been thought of as collections of conceptual tools to represent real-world entities and their relationships (ibid., p. 1). In this sense, the models are fit to achieve a level of specificity and efficiency that is integrated with the notions of economic success. That is to say, the quality of database access has a direct influence on the operational level of businesses. For example, if the database system in charge of airline reservations fails to update an entry or does not restrict duplicates, this might result in either empty airplanes or double-booking, an economic loss that might result in a company going out of business. In relation to data structure design within Computer-Aided Algorithmic Composition (CAAC) software, Ariza (2005a) claims that design choices determine "the interaction of software components and the nature of internal system processing" (ibid., p. 18). Luckily, a failed database access in music might perhaps come as a minimal performative 'bump' that can be otherwise

forgotten. However, it is imperative that these models are analyzed because of the continuum between data structures and database models, and because of the internal relations that resonate from these structures to the implementations of computer music software. Therefore, to a certain extent, database models and computer music software share the resonance of data structures, and belong to their realm.

2.4.1 Hierarchical

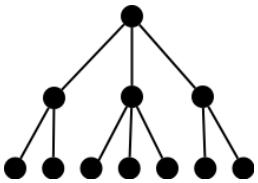


Figure 2.2: Hierarchical Model
Diagram of the hierarchical model

The hierarchical model was developed at International Business Machines Corporation (IBM) during the early 1960s, in conjunction with other American manufacturing conglomerates for National Aeronautics and Space Administration (NASA)'s Project Apollo, resulting in Information Management System (IMS) (Long, Harrington, Hain, & Nicholls, 2000). The hierarchical model is closely linked to the architecture of data within a computer. Therefore, it interprets records as collections of single-value fields that are interconnected by way of paths. Records can have type definitions, which determine the fields it contains. As a rule of this structure, a child record can be linked upwards to only one parent record and downwards to many child records. The structure stems from a single 'root' record, which is the initial parent-less record through which all other records are accessed.

This model is useful for nesting structures such as directory trees and path structures in most operating systems today. The relational model eclipsed their use within database systems during the 1980s, but it resurfaced through relational-type implementations of hierarchical models,

and with the appearance of semi-structured model in the late 1990s (See 2.4.7).

2.4.2 Network

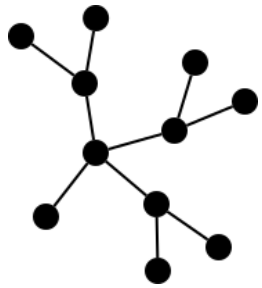


Figure 2.3: Network Model
Diagram of the network model

Invented by Charles Bachman in 1969 and published at the Conference/Committee on Data Systems Languages (CODASYL), the network model is a way of representing objects as nodes in a graph whose relationships can be represented as arcs. The programming language Common Business Oriented Language (COBOL) was designed for the implementation of network databases. The nodes in these networks are known as 'records,' and their relationships form 'sets' that have one-to-many relationships in between records, that is, one 'owner' and multiple 'members.' The main feature of a network model is that these relationships are not bounded to any hierarchical or lattice-like structures, providing a more natural way of record relation. Structurally, each node has an identity called a database 'key' which corresponds to the pointer to the physical address of the record on disk. This is how the network model maintains a close relationship between data structures and their traversal. Traversing the network means going from node to node, that is, keys can be used to implement linked lists for record navigation. These nodes do not have a hierarchical structure, meaning that the network can be accessed starting from any node. Due to the interlocking of the physical implementation and the internal logic of node identity and access, very fast retrieval speeds are obtained.

Navigational Paradigm The advent of disk-based database systems, in contrast to magnetic tape or punched card systems, enabled a different way of thinking database navigation. Working for General Electric's Integrated Data Store (IDS), Bachman (1973) later conceptualized and implemented a navigational paradigm within the networked model. Abandoning the "memory-centered view" of database system development, Bachman called for programmers "to accept the challenge and opportunity of navigation within an *n*-dimensional data space" (ibid., p. 657). Therefore, he proposed data records and attributes as *n*-dimensional space. This means that a database can be traversed not only by accessing the first element and then moving sequentially to the 'next' record. Secondary data keys could be made into sets for navigation starting from any of its members. In other words, given a database with records and attributes, all attributes can become a new dimension thus making retrieval times much more efficient. Navigating through a database within this paradigm is achieved by following record relationships instead of record order in physical storage. Therefore, with the navigational paradigm, a new level of abstraction was thus given to database management systems, resulting in better and more efficient database retrieval.

The navigational paradigm was implemented not only in network model, also in the hierarchical model, and it is still used today. Like I described with hierarchical databases, the navigational paradigm was eclipsed by the relational model, but after the 1990s, they re-emerged with non-relational databases. For example, since Document Object Model (DOM) websites contains a hierarchical structure, they can be accessed using this navigational paradigm.

2.4.3 Relational

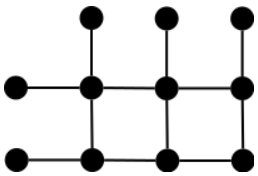


Figure 2.4: Relational Model
Diagram of the relational model

The relational model was first designed by Codd (1970), Codd (1972). Its main feature is the table-like organization of data, together with a separation between the physical level of data storage and the query language. These features allowed, on the one hand simple data visualizations, and on the other highly complex data manipulations by way of an algebra-based query language. Data is placed into uniquely identified rows (records) which can have multiple columns (attributes). A table thus becomes a relation. The main difference between the navigational and the relational paradigms, can be seen in the way users formulate queries. In the former, users specify which steps need to be made in order to arrive at a certain record. In the latter, users specify what needs to be found in terms of an algebraic expression. The query language developed for relational databases is Structured Query Language (SQL). In recent years, object relational database have emerged such as SQLObject, interpreting relations as classes in the object-oriented programming paradigm.

2.4.4 Non-Relational

This is a more general type of database model where the internal structure is different from the tabular kind that the relational model presents (See 2.4.3), and they are generally referred to as Non or Not only SQL (NoSQL). Within this class or group of non-relational models, some examples can be: Key-Value databases, which are centered on associative arrays (hash tables) such as python dictionaries; semi-structured databases (See 2.4.7), also called document-oriented databases such as Extensible Markup Language (XML), YAML Ain't Markup Language (YAML), and JavaScript Object Notation (JSON); graph databases and mixed graph models such as the way in which the

World Wide Web convention (W3C) structures websites, with a URL as a 'name' and their content as a 'graph' (See 2.4.5); object databases (See 2.4.6); and database systems using combinations of different models.

2.4.5 Graph

In their survey of graph-modelled databases, Angles and Gutierrez (Angles & Gutierrez, 2008) date the beginning of graph databases to the early 1980s, in conjunction with object-oriented databases. This model interprets records as 'nodes' and connections as 'edges.' Therefore, visualizations as graphs, as well as operations stemming from the mathematical theory of graphs, are features of the model. The visual programming paradigm takes advantage of graph representations of their object-oriented programming structure. In this sense, computer music software like OpenMusic, PWGL, Pure Data, MAX/MSP, Kyma, among others, present their objects as a directed graph on a canvas.

2.4.6 Object

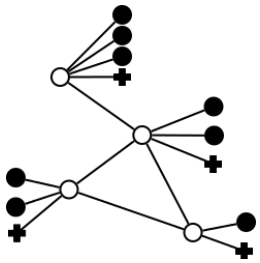


Figure 2.5: Object Model
Diagram of the object model

These databases combine the object-oriented programming paradigm with database concepts. On one side, each record is treated as an object, with capability to store variables (attributes) and functions (methods) that the object can perform. This way, when an object is instantiated in

the form of a record, all the attributes and methods become available to itself and to other objects, provided these are set up in a 'public' way, and so different interactions can occur throughout the database. Some programming languages are directly object-oriented, from which certain databases were created (See 2.1). From 2004, the open source community has been developing open source object databases that are easily accessible in several object-oriented languages.

2.4.7 Semi-structured

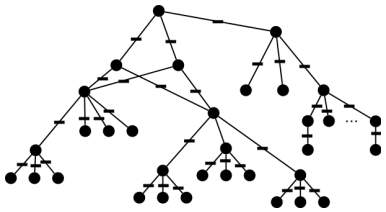


Figure 2.6: Semi-structured Model
Diagram of the semi-structured model

> We call here semi-structured data this data that is (from a particular viewpoint) neither raw data nor strictly typed, i.e., not table-oriented as in a relational model or sorted-graph as in object databases. (Abiteboul, 1996)

Abiteboul (ibid.) comments that given the amount of data that has grown in non-standard structures, a new way of accessing data has emerged. Furthermore, access to data can take place from a variety of different platforms such as browsers, query languages, application-specific interfaces, etc., making the process of obtaining useful information increasingly more difficult since these platforms call for specifically tailored methods and languages. Abiteboul claims, therefore, that first there is a need to extract the non-standard structure from the data, so that it can be traversed afterwards. These databases constitute the semi-structured model. Some examples of this model include XML databases, JSON files, YAML files, among others (Buneman, 1997). A well known database of this kind is the Internet Movie Database (IMDB).

2.4.8 Pure Data as Database System

While not technically a database system, Pure Data comprises (internally) a limited amount of data structures that are, nonetheless, different between each other. These structures are, in turn, arrays, linked lists, and symbol tables built as a layer of the C programming language. In terms of database models, Pure Data is mostly hierarchical when it comes to canvases. The windowing system that has a 'root', and multiple 'subcanvases' that can be (almost) infinitely nested. These canvases, while being hierarchic, are traversed as in the navigational model, either for a specific keyword (a query from the 'find' menu), or, most importantly, for signal processing. Besides this hierarchical structure, another important aspect of the Graphical User Interface (GUI) level is that it displays visually connected boxes with cords. Therefore, it is quite literally a directed graph where objects are nodes and edges are assigned to a node's inlets and outlets. The `.pd` file format, written in an application-specific language, is structured in such a way that elements on a graph are listed from top to bottom until the end of the list is reached. After this, the connections between objects inlets and outlets are subsequently listed. This graph model, however, comes out of Pure Data's internal design as an object-oriented program. Its core functionality depends on class instantiation. Every internal and external is a class made of C data structures with its own methods, that can be loaded in memory at run time and instantiated any time afterwards. Furthermore, Pure Data is already a networked environment, since in order to effectively 'patch' using the graphical interface, a network is established between Pure Data instance and the Tcl/Tk graphical interface. Added to this, the network capacity that Pure Data comes with, that is, the `pdsend` and `pdreceive` objects that support creation of endless Transmission Control Protocol / Internet Protocol (TCP/IP) connection sockets, literally exploding the concept of a hierarchical patch into the non-hierarchic, networked model.

A common warning that Pure Data developers have to announce is that if you open a listening port and share your port number, anyone can connect to that port, without any restriction

whatsoever.[1] This internet connectivity exposes users to one another in very direct ways, allowing system modifications that if used maliciously could potentially have detrimental effects. It can be argued that this loophole is a reflection of the internal openness of the source code itself. This openness enables programmers to create and load externals, but also to change the program itself. While changing something from the source code can be detrimental for the overall program, in being open, Pure Data prevents any definition to reach completion. A small gap, therefore, is left opened exposing users to the source, and to each other in a networked community.

Pure Data is just one example of many open and non-open source computer music softwares that expose such a plethora of database models for the user. Database models are what makes the realm of data structures reach any databaser: what touches any computer user that has ever pressed a key.

[1] Miller Puckette suggested this during an open discussion at **PdCon16**

Year	Model	Designer	Implementation
1959	Hierarchical	IBM	IMS
1960s	Network; Navigational	CODASYL; General Electric; Hewlett Packard (HP); United Information Systems (Unisys)	IDS; Integrated Database Management System (IDMS); Raima Database Manager (RDM); TurboIMAGE; Unisys OS 2200 databases (OS 2200); NoSQL
1960s	Deductive	J. Minker; L. Kuhns	
1960s	Non-relational	Apache; Sparsity;	MongoDB; Redis; Cassandra; Sparksee; NoSQL
1970s	Relational	E.F. Codd; P. Chen (1976)	MySQL; Oracle Corporation (Oracle); PostgreSQL; Microsoft Access (Access); Structured Query Language Lite (SQLite)
1975	Semantic model	U.S. Air force; J.H. ter Bekke (1991)	XPlain
1980	Graph	Oracle; Apache; Amazon	Neo4j; Oracle Spatial and Graph; ArangoDB; Amazon Neptune; Boost Software Library (BOOST); NetworkX (NetworkX)
1985	Object	Brown University; Texas Instruments; Bell Labs; Apache	GemStone (Smalltalk); Gbase (LISt Processor (LISP)); Appache Couch Database (CouchDB); SQLObject
1990s	Semi-Structured	W3C	XML; Sedna
1995	In-Memory	Oracle; Sybase; Exasol AG; VMWare	TimesTen; Adaptive Server Enterprise (ASE); High Performance Analytics Appliance (SAP HANA); EXASolution; WebDNA

Table 2.1: Database model development timeline with examples.

Chapter 3

Databasing Sound: Applications Of Databases In Sound

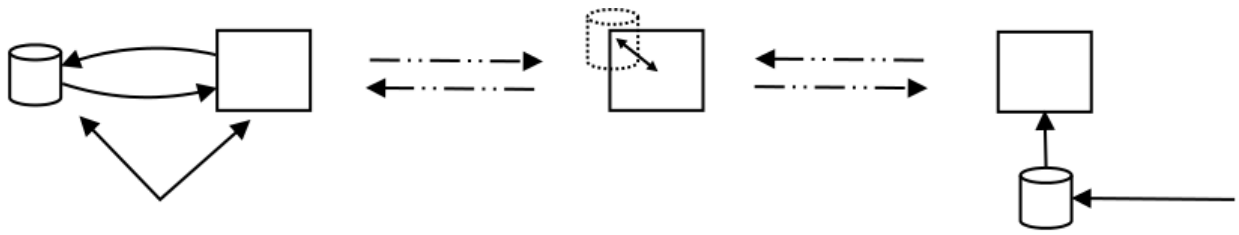


Figure 3.1: Database performance and interdisciplinary feedback.
The arrows between databases (cylinders) and computers (squares) represent data flow. Left: the database is 'visibly next' to the computer, as is the case with Music Information Retrieval (MIR); the two bottom arrows indicate the intervention of the human operator. Right: the database is 'visibly below' the computer as is the case with Sonification; the database feeds the computer from an external source (right arrow). Middle: the database is 'invisibly behind' the computer, within the softwares used for (and as) music works. The arrows in between the practices represent interdisciplinary feedback.

Having discussed the current state of new media theory and the theory of databases and data structures, in this section I theorize the use of databases in relation to sound. To a certain extent, ever since the first computers were used to make music the database has been an invisible partner in the music literature. I argue that by shedding some light on this inherent aspect of

computers we can arrive at a clearer notion of how databases sound. Particularly, by placing the database along a visibility continuum, we may find a reverse relation with audibility: the more invisible the database, the more present its sound. By this I do not argue in favor of either loudness or quietness. I am only addressing the different possibilities that come from multiple access points to computers. Here I will use the words 'database' and 'computer' somewhat interchangeably. This decision comes from the fact, as I described in earlier sections, that computers cannot exist without databases. From this, we can further ask ourselves if all computer music is database music.

As I demonstrate, there are overt and covert uses of the database, but the database is ubiquitous in all computer practices. The various disciplines at the intersection of music and computers take each a different approach to databases and, thus, to database performance. In this sense I describe and discuss the scope of actions that comprise database performance within three practices using computers and sound: MIR, sonification, and computer music. While there may be different ways of representing the position of the database in relation to each practice, I provide a set of figures throught this section that can be used metaphorically to point to the differences with which each practice interacts with databases (See Figure 3.1). Furthermore, these positions are only fixed within a graph in the form of snapshots. Thus, in no ways I mean to fix a certain position or to priorize one in relation to another. The purpose of these graphs is to describe a motility inherent to database practice that suggests furhter a way of thinking database performance in a choreographic way. By coreography, I mean the movement that we as 'computer musicians,' as 'sonificators,' and as music information 'retrievers' (in sum, as databasers), perform in the continuous dance of our practices.

3.1 Music Information Retrieval

In Music Information Retrieval (MIR), the database is *in front* of the programmer, *next* to the computer. This practice combines Information Retrieval (IR) with Music Theory, and it has been

present in academia for a while, most generally within Electrical Engineering departments. The objective of MIR is to obtain useful information from the analysis of sound signals. That is, MIR seeks to represent a complex signal with a small number of data points, thus defining a a navigable 'information space,' which is, quite literally, the discretized space of the database.

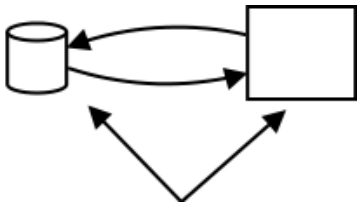


Figure 3.2: Diagram of database performance in MIR practices.
The database is visibly next to the computer, and the two bottom arrows indicate the intervention of the human operator.

For instance, out of sound file containing millions of samples, information space reduces these points to a database of few 'descriptors' that point to certain 'features' of the sound file. A descriptor is, in essence, a small amount of data that identifies other larger data. In this case, a feature descriptor relates to the values of a certain characteristics of the analyzed audio file, such as spectral centroid, brightness, flatness, etc.

Over the 18 years of the International Society for Music Information Retrieval (ISMIR) conference, more than thirty databases of this sort have been publicly created and released, as a means to classify millions of songs and musical genres. This type of database navigation has been used to perform automatic tasks such as categorization for recommendation systems (G. Tzanetakis & Cook, 2002; Dinuzzo, Pillonetto, & Nicolao, 2008; Poddar, Zangerle, & Yang, 2018), track separation or instrument recognition, and score transcriptions, among other uses (see below). A recent emphasis in open source database creation has gained momentum (Fonseca et al., 2017), such as the Freesound or Looperman.com (Looperman) databases, or Centre des Musiques Arabes et Mediterraneennes (CMAM)'s Telemeta, both collaborative database systems: the first two for general sound file sharing and classification, the latter for ethno-musicological purposes. Audio

databases such as Freesound or Looperman have been growing exponentially, as well as their use within live performances and interactive systems (Correia, 2010). Automatic audio description and clustering among these databases automatic have improved greatly their usability (Xambo, Roma, Herrera, & Laney, 2012). Collins (2015) created open-source software implementing MIR techniques for navigation, analysis, and classification of the electronic music archive within UbuWeb.

Before sound and audio descriptor databases, however, music notation databases have been developed with a variety of file formats (See 3.3.2). Some examples of these notation databases can be the Polish folk song database in the Essen Associative Code (EsAC) format, the electronic library for musical scores MuseData, the Repertoire International des Sources Musicales (RISM) database, the *Kern Scores* database,[1] among others. In turn, these databases have been a fruitful area of exploration in Computational Musicology (Yolk, Wiering, & van Kranenburg, 2011), for which toolkits such as Massachusetts Institute of Technology (MIT)'s Music21 have been developed. Two examples of widely used libraries for audio analysis, classification, and synthesis are Music Analysis, Retrieval and Synthesis for Audio Signals (Marsyas) (George Tzanetakis & Cook, 2000) and the Essentia (Bogdanov et al., 2013). For a more general overview of MIR software, see (ibid.). The different applications of databases are endless and so varied that would extend the scope of this study.[2] Some specific uses that MIR has given to databases have been:

- for audio classification and clustering (Yang, 2001; Homburg, Mierswa, Möller, Morik, & Wurst, 2005; Queiroz & Yoshimura, 2018)
- for genre recognition and classification (G. Tzanetakis & Cook, 2002; Xu, Zang, & Yang, 2005; Jr., Koerich, & Kaestner, 2008; Sanden, Befus, & Zahng, 2010; Dehkordi & Banitalebi-Dehkordi, 2018; X. Wang & Haque, 2017; Mitra & Saha, 2014; Correa, Saito, & Costa, 2010; Dinuzzo et al., 2008)
- to describe performance expression (Hashida, Matsui, & Katayose, 2008; Hashida, Naka-

[1] http://kern.ccarh.org/

[2] For example, consider the Musical Instrument Museums Online (MIMO) database, a project dedicated to the cataloguing of musical instruments, and how it was used for the statistical tracking of the evolution of the violin based on pattern recognition of its shapes (Peron, Rodrigues, & Costa, 2018)

mura, & Katayose, 2017; Hashida, Nakamura, & Katayose, 2018)

- for emotion recognition and color associations in the listener (Pesek et al., 2014)
- for multimodal mood prediction (Delbouys, Hennequin, Piccoli, Royo-Letelier, & Moussallam, 2018; Hu & Yang, 2014; Corona & O'Mahony, 2015)
- for multi-instrument recognition (Humphrey, Durand, & McFee, 2018)
- for the evaluation of multiple-source fundamental frequency estimation algorithms (Yeh, Bogaards, & Röbel, 2007)
- for contextual music listening pattern detection using social media (Hauger, Schedl, Kosir, & Tkalcic, 2013)
- for melody (Karydis, Nanopoulos, Papadopoulos, Cambouropoulos, & Manolopoulos, 2007; Bittner et al., 2014) or singing voice (Stoller, Ewert, & Dixon, 2017) extraction
- for structural analysis (J. B. L. Smith, Burgoyne, Fujinaga, Roure, & Downie, 2011)
- for schenkerian analysis (Kirlin, 2014)
- for harmonic analysis (Devaney, Arthur, Condit-Schultz, & Nisula, 2015)
- for melodic similarity (Haus & Pinto, 2005)
- for forensic analysis as a complement of video analysis (Serizel, Bisot, Essid, & Richard, 2016)
- for the evaluation of tempo estimation and key detection algorithms (Knees et al., 2015)
- for tonal music analysis using Generative Theory of Tonal Music (GTTM) (Hamanaka, Hirata, & Tojo, 2014)
- for counterpoint analysis (Antila & Cumming, 2014)
- to train models for phoneme detection (Proutskova, Rhodes, Wiggins, & Crawford, 2012) and music source separation (Miron & Janer, 2017)
- for training and evaluating chord transcription algorithms (Eremenko, Demirel, Bozkurt, & Serra, 2018)
- for training querying methods (Cartwright & Pardo, 2012; Brzezinski-Spiczak, Dobosz, Lis, & Pintal, 2013; Nagavi & Bhajantri, 2014; Nagavi & Bhajantri, 2013; Melucci & Orio, 1999)
- for adversarial audio synthesis (Donahue, McAuley, & Puckette, 2018)
- for orchestration (Crestel, Esling, Heng, & McAdams, 2017)

- for modeling carnatic rhythm generation (Guedes, Trochidis, & Anantapadmanabhan, 2018)
- to create digital libraries (Dunn, 2000)
- to store music notation (Good, 2000)

For further reference, the following citations point to different audio databases which have been created over the years: Goto, Hashiguchi, Nishimura, and Oka (2002), Goto, Hashiguchi, Nishimura, and Oka (2003), Wüst and Celma (2004), Maxwell and Eigenfeldt (2008), Bertin-Mahieux, Ellis, Whitman, and Lamere (2011), Karaosmanoglu (2012), Jaimovich, Ortiz, Coghlan, and Knapp (2012), Mital and Grierson (2013), Bortz, Jaimovich, and Knapp (2015), Jaimovich and Knapp (2015), Nort, Jarvis, and Palumbo (2016), Defferrard, Benzi, Vandergheynst, and Bresson (2017), Vigliensoni and Fujinaga (2017), Meseguer-Brocal, Cohen-Hadria, and Peeters (2018), Donahue, Mao, and McAuley (2018), Xi, Bittner, Pauwels, Ye, and Bello (2018), Wilkins, Seetharaman, Wahl, and Pardo (2018).

3.2 Sonification

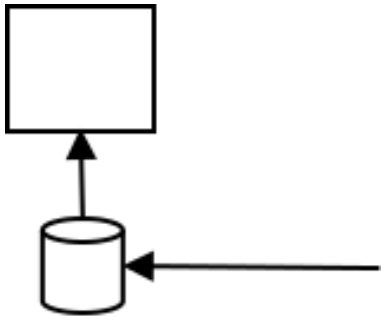


Figure 3.3: Diagram of database performance in sonification practices.
The database is visibly below the computer, and it feeds the computer from an external source represented by the right-most arrow.

The database is the ground floor of sonification. The sonified data is very likely to be

digital,[3] which means that data needs to be stored in a structured way for fast access by computers, and the role of the sonifier is to acoustically translate the database's inner relationships (Walker & Nees, 2011, p. 9).

According to Walker and Nees (ibid.) there are three types of sonification: event-based, model-based, and continuous. I see these types of sonification as ways of performing a database. Continuous sonification (audification) consists of directly translating waveforms of periodic data into sound, that is, reading non-audio data as if it were audio data (ibid., p. 17). Model-based sonification consists of distributing data points in such a way that enables data exploration. Generally, these models are interactive interfaces with which users navigate the database to find relationships (ibid., p. 17). Event-based (parameter mapping) sonification is aimed at representing changes in a database as acoustic saliences or tendencies. In this sense, dimensions of the data need to be translated (mapped) into acoustic parameters (frequency, periodicity, density, etc.), so as to listen how the generated sound behaves over time and interpret these changes within the database (ibid., p. 16).

Sonification depends on databases, on the interaction between databases, and on their traversing, but also on the human body's perceptual limits. In sonification, the data comes first, and it needs to be pre-processed so that it can be adapted to the sound synthesis engines of choice. Sonification is a subset of auditory display techniques, and it belongs to the broader scope of information systems and visualization practices (ibid., p. 10). Therefore, since sonification belongs to the process of information, as a practice it has taken into account the auditory system's ability to extract biologically relevant information from the complex acoustic world (Carlile, 2011). What this emphasis on sound perception and cognition abides to, however, is the fact that there is no one-to-one correspondence between sound parameters (frequency, amplitude, spectral content) and how these are perceived (pitch, loudness, timbre). Therefore, the success of a sonification is the result

[3]There are cases where sonification is entirely analog, such as the first sonification tool ever created: the Geiger counter

of the play between, on the one hand a rigid link between data and sound, and on the other, the perceived acoustic relations. From this interplay of relations is how information can be obtained from data. In other words, in sonification practices there is no communication unless the data has been acoustically shaped, and perceived as information (*in*-formed) by the listener.

In what follows, I present some instances of sonification practices as described by their authors.

3.2.1 Parameter mapping

DOW Rossiter and Ng (1996) sonified the Dow Jones financial stock market data with Csound. Since the Csound program depends on two separate files (orchestra and score), they implemented another program to control the data flow. Within this second program, the Csound score was automatically generated based on a 'configuration' file which was used to map the 'data file' holding the stock market data, as it was read in separate window frames into the Csound-formatted score.[4]

Medical Images Cádiz et al. (2015) proposed a sonification approach based on statistical descriptors of Region of Interest (ROI) selected from medical images. In their study, they focused on enhancing breast cancer symptom detection in mammograms by mapping statistical descriptors, such as mean, minimum, maximum, standard deviation, kurtosis, skewness, among others, to different synthesis techniques in various ways. They then surveyed the usefulness and pleasantness of the sonifications to different subjects in order to better adjust the technique to the task. What is novel of their approach is on the creative use of statistical curves obtained from pixel distributions within computer music techniques.

[4]Other examples of stock market sonification include Ciardi's set of tools for downloading and sonifying real-time data, see Ciardi (2004); and Ian Whalley's research on telematic performance, see Whalley (2014)

3.2.2 Model-based sonification

Space One example of model-based sonification is the *Data Listening Space* installation by the QCD-Audio project at the Institute of Electronic Music and Acoustics (IEM) of the University of Music and Performing Arts in Graz (Vogt, Pirro, Rumori, & Hoeldrich, 2012). Within this installation, they proposed a three dimensional, navigable space holding a Monte Carlo simulation of the theory of Quantuum Electrodynamics (QED). Within this QED *lattice*, a walking participant holding sensors —x, y, and z coordinates— could explore the simulated data by way of sonification.

3.2.3 Artistic sonification

Wolves J. Klein (1998) composed a piece called "The Wolves of Bays Mountain", using a set of recordings she took along the Bays Mountain Park in Kingsport, Tennessee, for a period of six months. In this period she researched the sonic activity of a pack of wolves, and in her recordings she achieved a level of intimacy with the pack that translated into the recordings, and resulted in a strong animal rights activism (J. Klein, 2017). Therefore, her compositional choice was to treat the sound file in a non-destructive and non-intrusive way: "for the composition I used the Csound computer music language. All of the sounds came from the recordings, in unaltered or slightly modified form as the source material in musical settings and transitions" (J. Klein, 1998). Thus, by analyzing spectral contours of extremely precise frequency bandwidths of the data and resynthesizing into the soundscape in almost unnoticeable ways, she sonified a space in between the wolves. This space invites the listener into a space of action, and to reflect on human activity itself and how it always returns to resonate with the wolves.

Selva Barrett (2000a) composed an electroacoustic work called "Viva La Selva" (Barrett, 2000b) using 14-hour long recordings taken with an array of four microphones from a a biological field station called *La Suerte* in Costa Rica. From these recordings, she extracted location (by difference

in arrival time) and timestamps (by manual logging) of different animal sounds, and long-term energy distribution in various frequency bands, to describe various environmental sounds such as airplanes, wind, insects, etc. While the spatio-temporal data of the animal sounds was used for sound spatialization of sounds within the electroacoustic work, the long-term energy distribution was scaled down to 20 minutes so as to constitute the form of the piece.

Ocean B. L. Sturm (2002) sonified ocean wave conditions of the USA Pacific coast obtained by the Coastal Data Information Program (CDIP) since 1975. The database until 2002 contained over 50 Gigabyte (GB) of spectral and directional content of the wave-driven motions at the location of the sensing buoys. By scaling to hearable range and then performing an Inverse Fourier Transform (IFT) of the data, Sturm composed a piece called *Pacific Pulse*, on which frequency sweeps indicate storms beginnings (rising) and endings (falling).

Molecules Falk Morawitz (2016) composed *Spin Dynamics* using molecular sonification by two audification processes (direct audification and via a straightforward additive synthesis process) applied to the Human Metabolome Database (HMDB), a database holding Nuclear magnetic resonance (NMR) spectroscopies of molecules.

Gender Distribution Emma Frid (2017) derived a database of gender distribution by applying the python module `genderize` to author names in three main computer music conference proceedings databases: International Computer Music Conference (ICMC), New Interfaces for Musical Expression (NIME), and Sound and Music Computing Conference (SMC). By assigning polar frequency ranges for each group (male and female), her sonification emphasizes the significant inequality of gender in the resulting acoustic stream segregation into male background (continuous drone-like sound) and female foreground (fewer and sparser sounds). Her conclusion, therefore, is that "there is a need for analysis of the existing environments and social relations that surround music technology and computer music. If we identify the challenges that women are facing in our

research community, we will be able to create more initiatives towards changing practices" (ibid., p. 238).

3.2.4 Sonification Installations

IP-based soundscape Ballora, Panulla, Gourley, and Hall (2010) sonified a database of Hypertext Transfer Protocol (HTTP) requests at Penn State's Center for Network-Centric Cognition and Information Fusion (NC2IF). This database contained entries with four fields such as timestamp, location (latitude-longitude), Internet Protocol (IP) address, and response type. Using parameter mapping, Ballora controlled rhythm and spatialization with the first two, and pitch and timbre with IP data. However, the latter ranged from the more concrete (IP to frequency) to the more abstract (IP as formant and high-pass filters for brown noise), thus resulting in a soundscape with different but simultaneous sonifications of the data. This multi-layered approach to sonification stems from his PhD dissertation on cardiac rate sonification (Ballora, 2000).

Earthquakes Lindborg (2017) sonified real-time earthquake data as a sound sculpture. Within "Pacific Bell Tower, a sculptural sound installation for live sonification of earthquake data", he used data from the Incorporated Research Institutions for Seismology (IRIS) Data Services, which transmits seismographic data packets updated every thirty minutes from multiple observation sites. He spatialized this data using coordinates of the events and using a four-speaker array located at the center of the gallery space, and mapped the rest of the data to Frequency Modulation (FM) synthesis parameters.

GPU-based waveforms Schlei and Yoshikane (2016) proposed a novel way to generate waveforms by populating an array using vertex data obtained from the Graphics Processing Unit (GPU). In order to carry this out, they used the Metal API[5], and intervened on the processing pipeline to

[5]Apple's built-in framework to interface with the GPU. See https://developer.apple.com/documentation/metal

output Central Processing Unit (CPU) accessible data. The audio engine running on the CPU was able to interpret as waveforms the values of the vertex and fragment shaders, thus sonifying the position data related to a rendered shape and the pixel values respective to its display. Therefore, they obtained simultaneous visualization and audification of the rendered three dimensional shape. In their installation *The Things of Shapes*[6], they used the generated waveforms as a database, composing each waveform together with their visual generators as a collage.

Uncanny Faces Simonelli, Delgadino, and Halac (2017) designed *Hally*, an installation based on face tracking and real-time sonification of spectral features present in both pixel information containing the face, and the x and y coordinates of the moving data points of the face mesh used for tracking. Furthermore, by video-based audio convolution, *Hally* aims to simulate a theory of perception based on IFT (Connes, 2012). Parting from previous work by Thiebaut, Bello, and Schwarz (2007) on simultaneous sonification and visualization, *Hally* explores the role of both sound and image in the definition of the self, by immersing the participant in an uncanny spectrality (Cámara Halac, 2018a). Among other sounds, there is one drone-like sound produced throughout the installation that comes out of a set of programming decisions which bring into surface certain aspects of the software used in the piece. The within Pure Data is scheduled in blocks of samples that the user can set. A certain combination of block size, overlap, and sample rate results in a frequency, in this case, of about 43 Hz (Simonelli et al., 2017, p. 3). In fact, these combinations around block size result in a set of octave-spaced pitches that are tethered to Pure Data's only possible block sizes (2^n, at least in audio computations). In this sense, when making these pitches audible, the listener can recognize a certain 'sound' proper to the internal memory management and computation rate of the software. In this installation, it is the human participant that enters as a filter to the spectrum of this 'sound' by the position of their faces on the camera sensing area. Thus, participants change this 'sound' with the unique ways in which they behave in front of the

[6]https://vimeo.com/167646306

sensor.

3.2.5 Sonification Software

SonArt Originally intended for sonification purposes, SonART (Ben-Tal, Berger, Cook, Daniels, & Scavone, 2002) was an open-source platform that enabled users to map parameters to sound synthesis, and later (W. Yeo, Berger, Lee, & Zune, 2004) to obtain cross-correlated image and sound synthesis. In other words, users were able to easily translate a database into sound parameters, or image and sound data into one another. The program acted in a modular way, that is, it was networked with other software via Open Sound Control (OSC) connections. This software enabled W. S. Yeo and Berger (2005) to generate novel image sonifications, by combining two methods of sonification into one interface: sonified data in a fixed, non-modifiable order (*scanning*) and sonified selected data points (*probing*).

DataPlayer In his Computer Aided Data Driven Composition (CADDC) environment called *DataPlayer* programmed as a standalone MAX/MSP application, Nardelli (2015) sonified data from the Automatic Flow for Materials Discovery (AFLOWLIB). His sonification intent was aimed towards data navigation by means of a unique mapping that would convey an overall trend (a gist) of each material compound. Furthermore, this environment allowed for artistic remixing and exploration of the sonification procedures, simultaneously touching on the scientific and the artistic uses of the environment.

madBPM Fox, Stewart, and Hamilton (2017) devised madBPM, a data-ingestion engine suitable for database perceptualization, that is, sonification and visualization. This modular C++ software platform enables data loading from Comma Separated Values (CSV) files, multiple mapping via tagging, several traversing algorithms and units, and networked connectivity to SuperCollider for sound and OpenFrameworks (OFX) for visual output. Their approach is innovative since they

provide features for database behaviors. By 'behavior' they mean ways of structuring, traversing and perceptualizing the database. These behaviors define the dual purpose of the software: finding relationships among the inputted data and interpreting them artistically. Furthermore, users can structure and re-structure potentially any type of data set (ibid., p. 504). However, in order to design new behavior objects the user needs to implement them in the source code and compile them. Thus, besides real-time data streaming and networking functionality, in their future work the authors aim at designing a Domain Specific Language (DSL) that would enable extending the functionality of these behaviors in real-time.

For further sonification software, see Sonifying Data (SonData) and the following references: Wilson and Lodha (1996), Pauletto and Hunt (2004), Lodha, Beahan, Joseph, and Zaneulman (1998), Beilharz and Ferguson (2009), Hildebrandt, Hermann, and Rinderle-Ma (2014), Worrall, Bylstra, Barrass, and Dean (2007), Walker and Cothran (2003), Vicinanza (2006)

3.3 Computer Music

Computer music software is computer music's playground. Composing and programming blend into different forms of play that can be understood by a closer look at the playground's design. A key aspect of software design is delimiting constraints to data structures. The first choice is generally the programming language, after which the database tree unfolds its way up to the leaves. Among these leaves is where computer music programs reside. At this level of 'leaves' software users are certainly aware that there is a 'tree' in front of them. However, their awareness does not necessarily extend to the branches, trunk, or roots of the tree. There is endless music that can be made with leaves just as it can with paper. However, neither music quantity nor music quality are the point here. My argument is that working with data structures changes how we think and perform music making. I claim that composers using these leaves of computer music software are working indirectly with data structures, and unless they engage with programming, they re-

main unaware of data structures and their constraints. 'Indirectly,' because the twigs and branches connect the leaf to the trunk, but these connections become invisible to the non-programmer composer *by design*. Like a phantom limb of the tree, the database remains invisibly *behind*. In this section, I present different approaches from composers and programmers that show how music concepts change with the presence and performance of the database. By database performance I mean neither the quality of musical output, nor the dexterity of the programming activity. Database performance in music composition is the activity of the databaser: databasing to make music.

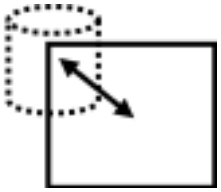


Figure 3.4: Diagram of database performance in computer music practices.
The database is invisibly behind the computer, within the softwares used to create musical works.

3.3.1 Hierarchical environments

> One of the most important aspects in the design of any computer system is determining the basic data types and structures to be used... we have been guided by our projection of the interaction between the tool which we are developing, and the composer. (Buxton, Reeves, Baecker, & Mezei, 1978, p. 119)

Reducing cognitive burden In William Buxton's survey of computer music practices (Buxton, 1977; Buxton, Reeves, et al., 1978; Buxton, Patel, Reeves, & Baecker, 1980), he distinguished between *composing programs* and *computer aided composition*, arguing that they had both failed as software, the former on account of their personalization and formalization, and the latter on their lack of interactivity. On his later interdisciplinary venture called Structured Sound Synthesis Project (SSSP), he focused on Human Computer Interaction (HCI) —a field in its very early stages

in 1978—[7]. Buxton's concern throughout his work on SSSP was to address the"problems and benefits arising from the use of computers in musical composition" (Buxton, Fedorkow, et al., 1978, p. 472). His solution to the problems was to reduce the cognitive burden of the composer, who "should simply not have to memorize a large number of commands, the sequence in which they may be called, or the order in which their arguments must be specified" (ibid., p. 474). He argued that reducing the amount of information given to composers helped them focus on music making. Therefore, in SSSP, the composer's action was reduced to four main selection tasks: timbres, pitch-time structure, orchestration, and playback. Timbres were assigned by defining waveforms for the table lookup oscillators, and pitch-time structure consisted on pitches and rhythms on a score-like Graphical User Interface (GUI) program called SCRIVA (Buxton, 2016a). Orchestration consisted in placing the previously chosen timbres on the score, and playback meant running the score or parts of it. With this simple but very concise structure, stemming from a somewhat dated programming philosophy in relation to audio software, Buxton delimited the scope of action of the composer

A Hierarchical Representation Buxton, Fedorkow, et al. (1978) based their research on differing approaches to composition: Iannis Xenakis's score-as-entity approach in his 1971 *Formalized Music*, an unpublished 1975 manuscript by Barry Vercoe at Massachusetts Institute of Technology (MIT) studio for Experimental Music, where Buxton found a note-by-note approach, and Barry Truax's computer music systems (Truax, 1973) which was, for Buxton, located somewhere in between the first two but did not provide a solution for "the problem of dealing with the different structural levels of composition —from note to score" (Buxton, Reeves, et al., 1978, p. 120) (See 3.3.1). Buxton, however, condensed these different approaches into what he called a "chunk-by-chunk" composition, where a 'chunk' represented anything from a single note to an entire score, and thus reframed the question of a compositional approach as one of scale. For Buxton, "the key

[7]William Buxton is now considered a pioneer in HCI, and he is now a major figure in the Microsoft Research department.

to allowing this 'chunk-by-chunk' addressing lies in our second observation: that the discussion of structural 'levels' immediately suggests a hierarchical internal representation of scores" (ibid., p. 120). That is to say, his solution for the scalability problem relied on a hierarchical representation of scores.

In Buxton's SSSP, the hierarchical design depended on a data structure called *symbol table*, which he subsequently divided into two objects called `score` and `Mevent` (musical events). The `score` structure had a series of global fields (variables) together with pointers to the first (head) and last (tail) `Mevents`. In turn, `Mevents` had local fields for each event together with pointers to the next and previous `Mevents`, so as to keep an ordered sequence (See 2.3) and enable temporal traversing of the tree. In turn, `Mevents` could have two different types: `MUSICAL_NOTE` and `Mscore`, the former relating to terminal nodes editable by the user —what he referred to as 'leaves' of the tree structure—, and the latter consisting of nested `score` objects that added recursivity to the structure. Buxton's model was thus hierarchic (a tree structure) implemented in nested and doubly-linked symbol tables.

Buxton, Reeves, et al. (ibid.) gave a detailed exposition of the data structures and their functionality. Buxton's general purpose in his HCI philosophy was to make the software work in such a way that it became invisible or transparent to the user. This is also known as a black-box approach. His innovations in this and other projects have had enormous resonances in computer science, and the concept of reducing cognitive burden of the user has developed as a standard of HCI (Buxton, 2016b).

Black-boxing Media theorist Vílem Flusser (2011) proposed the term 'envision' to describe a person's power to visualize beyond the surface of the image, and to bring the technical image into a concrete state of experience. The 'image,' in Flusser's case is the television screen in its abstract state of "electrons in a cathode ray tube." Therefore, he argues, "if we are asking about the power to envision, we must let the black box remain —cybernetically— black" (ibid., p. 35). By seeing past

the abstract quality of media we bring an image into experience. The black box allows envisioning to take place. In a similar way, by seeing past the hidden complexities of the software, composers are able to create music with unrestrained imagination. However, as I have shown before, Hansen makes a divergent point claiming that the virtuality inherent in the body is the creative potential of image *in-formation* (See 1.5).

Understanding the process of information as the experience of technical images, it follows that virtuality and envisioning can be considered complementary. On one hand, there is the technical device, whose multidimensionality is as complex as it is hidden from the envisioner. On the other, the human body with its capacity to create and embody. Flusser's point is, however, paradoxical: "The envisioner's superficiality, to which the apparatus has *condemned* him and for which the apparatus has *freed* him, unleashes a wholly unanticipated power of invention" [emphasis added] (ibid., p. 37). Therefore, the black-box is what condemns and frees the envisioner to a state of superficiality. However, Flusser continues, "envisioners press buttons to inform, in the strictest sense of that word, namely, to make something improbable out of possibilities" (ibid., p. 37). In other words, Flusser justifies the invisibility of the technological device in favor of its most useful consequence, that is, its ability to make the user create something "out of possibilities." Composers, therefore, are often given these possibilities to create, at the cost of a restricted creation space.

Generality and Portability

> Music data structures must be general enough so that as many styles of music as possible may be represented. This implies that the data structures (or the application's interface to them) should not enforce a musical model (such as equal temperament) that is inappropriate for the musical task at hand. (Free, 1987, p. 318)

The SSSP lasted until 1982 due to lack of funding, and in the mid-1980s its research re-emerged with the work of Free and Vytas (1986), Free (1987), Free and Vytas (1988), under Helicon Systems' Computer Assisted Music Project (CAMP). Free's programming philosophy

called for generality, portability, and simplicity. Due to SSSP's many hardware dependencies, the code had to be completely re-written (Free & Vytas, 1986). A crucial aspect of Free's programming concerns was portability (See 2.2), which moved him to create higher levels of software abstractions, so that software continued to live on in newer hardware. Free also developed SCRIVA, SSSP's GUI program into extensible data structures for music notation arguing that software had to be general enough so that composers could work in multiple styles. The larger implication in Free's argument is that enforcing musical concepts in data structures limits the style that the program can achieve. Therefore, if the program fails to provide a certain level of generic functionality, the composer's output will be modelled by the data structure. On the one hand, it can be argued that this implication is simultaneously overestimating the agency of the database and underestimating that of the composer. In any case, the database works for the composer by taking care of the more tedious task. The cost of this, nonetheless, is that by working for the composer, the database guides the composer through certain paths while hiding other paths.

Simplification Hardware-independence led Free to imagine a general purpose, or *vanilla* synthesizer, with which students in "a music lab with multiple users on a networked computer system" (Free & Vytas, 1988, p. 127) could seamlessly use the timbre world offered by various synthesizers made by different manufacturers. Free created a database that enabled simultaneous interaction among different types of hardware. The *Music Configuration Database* consisted of an intermediate program between the physical Musical Instrument Digital Interface (MIDI) input devices (such as the Yamaha DX7 or Casio CZ101), and the computers in the network, so that "rather than have the user tediously specify the MIDI device properties for each synthesizer" (ibid., p. 133) (channel management, control mapping, etc), these processes were handled by an intermediary database. Free's approach, in comparison to Buxton's, was not entirely black-boxed, since the database was open to modification by a specific set of commands provided to the user. The user could edit the database with a library of database access subroutines such as open/close, create/delete items,

querying fields/keys, and loading/storing property items. With this library, Free simultaneously simplified user's interaction and reduced the "chance of corrupting the database" (ibid., p. 137).

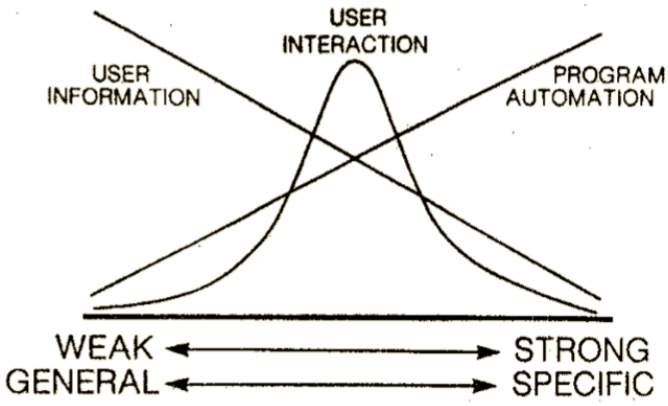


Fig. 1. Variation of user information, program automation, and user interaction as a function of the generality and strength of a system.

Figure 3.5: Generality vs. Strength
Barry Truax' "Inverse Relation Between Generality and Strength" (Truax, 1980, p. 51). Another version of this graph can be found in (Laske & Tabor, 1999, p. 38).

Balance

> . . . all computer music systems both *explicitly and implicitly embody a model of the musical processes that may be inferred from the program and data structure of the system*, and from the behavior of user working with the system. The inference of this model is independent of whether the system designer(s) claim that the system reflects such a model, or is simply a tool. [emphasis added] (Truax, 1976, pp. 230–231)

Truax (1973), Truax (1976), Truax (1980), Emmerson (1986, Chapter 8) often compared grammatical structures of natural language to the structures of computer music systems, claiming that in both cases one can find certain constraints and facilitations for thought (Emmerson, 1986, p. 156). Arguing for balance between generality of applicability and strength of embedded knowledge within models for computer music systems (See Figure 3.5), he writes:

> In a computer music system, the grouping of data into larger units such as a sound-object, event, gesture, distribution, texture, or layer may have a profound effect on the composer's process of organization. The challenge for the software designer is how to provide powerful controls for such interrelated sets of data, how to make intelligent correlations between parameters, and how to make such data groupings *flexible according to context*. [emphasis added] (ibid., p. 157)

Truax's notion of balance speaks of a 'meeting halfway' between the system and the user regarding the programmer's capability to embed a more complex conception of hierarchy in the system. What provides this balance is a certain flexibility among data structures which would enable them to adapt to the different hierarchical contexts with which music is understood. That is to say, since data structures can embody models of musical processes, they have an effect on the composer's overall performance of the database, and by extension, on the resulting music.

3.3.2 Music Notation Software

Music representation has occupied an important area of research within the programming community. Formats and specifications such as MIDI, MusicXML (MusicXML), the Humdrum `**kern` data format (Sapp, 2005), GUIDO Music Notation Format (GUIDO), to name a few, have appeared over the years in conjunction with music engraving software. An extensive guide on musical representations can be seen in Selfridge-Field (1997a). In this section, I point to certain aspects of music notation software development that reveal different approaches towards data structures, and the possibilities that arise henceforth.

DARMS and SCORE Two major programs were developed during the 1960s and 1970s: Stefan Bauer-Mengelberg's Digital Alternate Representation of Musical Scores (DARMS) project for music engraving which started in 1963 (Brinkman, 1983; Erickson, 1975), and Leland Smith's SCORE (L. Smith, 1972). Both of these programs worked first in mainframe computers and were used for music printing and publishing. At first, SCORE's character scanner was designed to inter-

pret complex musical input into MUSIC-V output, thus acting as an link between music notation and computer music synthesis. However, with the appearance of vector graphics in the 1970s it shifted solely to music printing. With the appearance of the PostScript format in the 1980s, it became commercially available thus becoming one of the earliest music engraving softwares still in use today by major publishing houses (Selfridge-Field, 1997b).

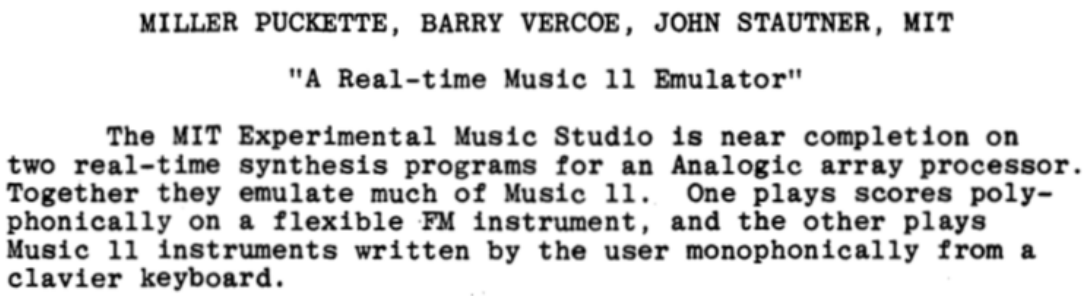
From Staves to Speakers Other programming approaches stemming from DARMS and SCORE were developed during the 1980s. Clements (1980) joined together the DARMS data structures with those used in MUSIC-V in a first attempt to obtain sonic feedback out of a notation system. Clements' attempt was nonetheless overshadowed by SCORE's success. Later, Dydo (1987) worked on an interface to the DARMS language called the *Note Processor*, which became one of the earliest commercially available music notation systems. Dydo's data structures, however, were not publicly released when he presented his software at the International Computer Music Conference (ICMC) in 1987. He later released it commercially in the early 1990s at a significantly lower price than other notation software such as *Finale* which is still available today by MakeMusic, Inc. (Skinner, 1990a; Skinner, 1990b). Brinkman (1981) modeled the SCORE input format into *Score-11*, adapting it to Barry Vercoe's MUSIC-11. Written in Pascal, *Score-11* used circular linked lists traversed by an interpreter to produce MUSIC-11-formatted output. The user creates a text file with blocks dedicated to individual instruments and specifies parameters such as rhythm, pitch, movement (glissandi, crescendo), amplitude, etc. These parameters are then reformatted to fit the less musically-oriented notation of the MUSIC-N programs. Brinkman argued that such a software would result in faster and less arduous performance on the composer's end: "a crescendo over several hundred very short notes requires several hundred different amplitude values representing the increasing volume. *Typing in several hundred note statements each with a slightly larger amplitude number would take forever.* If the computer could be instructed to gradually increase the amplitude value over twenty seconds then *life would be much simpler*" [emphasis

added] (Brinkman, 1982). Brinkman emphasized on the program's extensibility by users, inspiring Mikel Kuehn's recent *nGen* program (McCurdy, Heintz, Joaquin, & Knevel, 2015), a version of Brinkman's program for the current Csound. Brinkman (1983) later designed an interpreter for the DARMS language, which became useful for obtaining computable data structures for automated music analysis (Brinkman, 1984). Another approach to music notation was carried out at Center for Computer Research in Music and Acoustics (CCRMA), when G. Diener (1988), G. Diener (1989) devised a "pure structure" devoted to the "hierarchical organization of musical objects into musical scores:" the *TTree* (G. Diener, 1988, p. 184). Stemming from his PhD research on formal languages in music theory (G. Diener, 1985), this data structure was based in the hierarchic structures of the SSSP project. The change Diener introduced to these structures was their capability of sustaining links between not only the previous and the next data records, but to the 'parent' or 'child' data records to which it was related. This is known as 'inheritance,' and it enabled "any event in the [structure] to communicate with any other event" (G. Diener, 1988, p. 188). While Diener implemented this data structure in the object-oriented programming language Smalltalk, he later developed it into *Nutation* (G. R. Diener, 1992), a visual programming environment for music notation. *Nutation* was written in Objective-C, and it combined the previously developed *TTree* structure with glyphs and a music synthesis toolkit called *Music Kit* that the NeXT computer provided. This resulted in an extremely malleable Computer Aided Composition (CAC) environment, which enabled fast manipulation and sonic feedback at the cost of limiting timbre to a predefined, hardware-specific set of digital instruments.

Theoretical Performance What notation software is most often criticised for is the way in which sonic feedback often comes to be equiparated to (human) music performance. When Leeland Smith presented SCORE as "not a 'performer's' instrument, but rather a 'musician's' instrument," for example, he claimed that "theoretically, any performance, clearly conceived in the mind, can be realized on [the computer]" (L. Smith, 1972, p. 14). It is indeed a fact that computers can

offer automated tasks to an unimaginable extent. However, to translate this type of automation into music composition and performance, results in a disembodied music conception. In other words, an algorithmically generated stream of notes may result in physically impossible tasks for a performer, or for the listener. This is the point of inflexion when envisioning goes beyond the threshold of embodiment. It can be argued, however, that further developments in musical performance techniques can be achieved by pushing the limits of bodily skills. Nonetheless, what I am stressing here is the extent to which music composition can be reconfigured by the possibilities data structures have brought to the field. Furthermore, what is at stake with notation-based music software is yet another musical concern that governed most of music software development during the 1980s: style.

3.3.3 Enter Objects



MILLER PUCKETTE, BARRY VERCOE, JOHN STAUTNER, MIT

"A Real-time Music 11 Emulator"

The MIT Experimental Music Studio is near completion on two real-time synthesis programs for an Analogic array processor. Together they emulate much of Music 11. One plays scores polyphonically on a flexible FM instrument, and the other plays Music 11 instruments written by the user monophonically from a clavier keyboard.

Figure 3.6: A real-time version of MUSIC-11.
A bodiless abstract published at the ICMC (1981) stating that a real-time version of MUSIC-11 was “near completion” by a group at MIT (M. Puckette, Vercoe, & Stautner, 1981).

Max Faster, cheaper, and portable microcomputers with real-time capabilities for audio processing began to appear onstage within institutions such as MIT and Institut de Recherche et Coordination Acoustique/Musique (IRCAM), and a growing interest among composers and programmers circled around real-time computer music software (See Figure 3.6). Towards the end of the 1980s,

after the proliferation of MIDI (Loy, 1985), composers were already incorporating real-time techniques within musical instruments and software (Vercoe, 1984; M. Puckette, 1991). This is the context for Miller Puckette's development of Max for the 4X real-time audio processor at IRCAM (M. Puckette, 1986). With an emphasis on time and scheduling, Puckette devised a new approach towards complexity in computer music software:

> ...complexity must never appear in the dealings between objects, only within them. Three other features currently in vogue seem unnecessary. First, there is no point in having a built-in notion of hierarchy; it is usually a hindrance. Second, I would drop the idea of continuously-running processes; they create overhead and anything they do can be done better through [input, output] related timing. Third, there should be few defaults. Rather than hide complexity I would keep it visible as an incentive to avoid it altogether. (ibid., p. 43)

Puckette keeps complexity "visible" within the concept of the programming *object*. Furthermore, he removes the notion of hierarchical programming proposing a light-weight, on-the-spot programming practice based on discontinuous processes: "the scheduler keeps the runnable-message pool in the form of a separate queue for each latency" (ibid., p. 46). In other words, the structure of the database was placed *horizontally* along the time axis, and Puckette's efforts were dedicated to optimizing the internal timing of audio processes. Specifically, linked lists are used to keep track of the order of processes that are run, and each process is scheduled according to its own temporality (latency). Thus, the entire network of processes that can be run is maintained in a dynamic list (stack) that can be changed at any time by adding or removing elements (push/pop). The way in which these processes (methods) are called is by messages that can be sent (input/output) by the user or objects themselves.[8] In sum, the object-oriented paradigm was thus applied to the scheduling system, resulting in a ground-breaking implementation that changed the

[8] "The scheduler always sends the first message in the lowest-latency non empty queue. When the associated method returns the scheduler sends another message and so on. The only situation in which we need to interrupt a method before it is done is when I/O (including the clock) causes a lower-latency message to appear... In this case the scheduler causes a software interrupt to occur by pushing a new stack frame onto the stack and executing the lower-latency method. When this method returns ... we pop the stack back to the prior frame at latency d_2 and resume the associated method" (M. Puckette, 1986, p. 46).

real-time computer music performance scene: "... rather than a programming environment, Max is fundamentally a system for scheduling real-time tasks and managing intercommunication between them" (M. Puckette, 2002a).

Kyma Another powerful example of an object-oriented language for non-real-time music composition is Carla Scaletti' Kyma, developed at the University of Illionis' Computer Based Education Research Laboratory (CERL) (Scaletti, 1987). It is designed as an interactive composition environment for the Platypus digital signal processor. Scaletti's language was hierarchical in its structure, enabling data records to be linked vertically and horizontally. Together, these data structures formed objects, enabling the composer to treat any set of sounds within the composition, and even starting from the composition itself as an object. In such a way: "... the composer could create a 'sound universe,' endow the sound objects in this universe with certain properties and relationships, and explore this universe in a logically consistent way" (ibid., p. 50). Given the "vast amounts of data required for sound synthesis" (ibid., p. 50), Kyma's objective was to fit timbre creation and temporal event lists into the same traversable database underlying the program. Like Puckette and Free, Scaletti's design was aimed at a language that "itself would not impose notational or stylistic preconceptions" (ibid., p. 50).

On one hand, Scaletti based her research on Larry Polansky's Hierarchical Music Specification Language (HMSL), another "non-stylistically based" music composition environment that was "not fundamentally motivated by a desire to imitate certain historical compositional procedures" (Rosenboom & Polansky, 1985, p. 224). Polansky's focus was on a language that would "reflect as little as possible musical styles and procedures that have already been implemented —like conventional music notation—" (ibid., p. 224). On the other hand, the "notational bias" (Scaletti, 1987, p. 49) that Scaletti recognized in languages such as MUSIC-V and FORMES (Rodet, Barrière, Cointe, & Potard, 1982; Boynton, Duthen, Potard, & Rodet, 1986), prescribed a very clear division between composition and synthesis which, in turn, was a very difficult and

time-consuming "wall" she had to "circumvent" (Scaletti, 1987, p. 49). Therefore, she imagined a language in which the composer could "choose to think in terms of notes and keyboards and staves but in which this structuring would be no easier and no harder to implement than any of countless, as yet uninvented, alternatives" (ibid., p. 49). Both HMSL and Kyma are still in used today, the former with a further Java version by Didkovsky and Burk (Didkovsky & Burk, 2001), and the latter embedded into a commercially available workstation.[9]

Pure Data Ten years after Max, M. S. Puckette (1997) moved on to Pure Data. The commercially available MAX/MSP (Zicarelli, 1998) presents, like Pure Data, the Max programming paradigm (M. Puckette, 2002a). In resonance with the neutrality of the 1980s, Puckette introduced more data structure flexibility as a means to provide a musical instrument without stylistic constraints. Data structures became a more accessible feature for the user to define and edit:

> The design of Max goes to great lengths to avoid imposing a stylistic bias on the musician's output. To return to the piano analogy, although pianos might impose constraints on the composer or pianist, a wide variety of styles can be expressed through it. To the musician, the piano is a vehicle of empowerment, not constraint. (ibid.)

Puckette, therefore, aims to a certain stylistic neutrality, which he represents by the way in which the user opens the program: a blank page: "no staves, time or key signatures, not even a notion of 'note,' and certainly none of instrumental 'voice' or 'sequence"' (ibid.). While acknowledging that even the 'blank page' is a culturally loaded symbol referring to the use of paper in Western Art Music (much in the same way that it is favoring complexity altogether), Puckette reconfigured computer music design, composition, and performance by considering the way in which the structure of the program resonates aesthetically.

Graphic Scores In order to include graphic scores for electronic music within the Pure Data, Puckette implemented a data structure deriving from those of the C programming language, which

[9] https://kyma.symbolicsound.com/

can be used in relation to any type of data: "the underlying idea is to allow the user to display any kind of data he or she wants to, associating it in any way with the display" (M. Puckette, 2002b, p. 184). Puckette's philosophy, as I have mentioned earlier, was aimed at detaching music software from music concepts, leaving these aesthetic decisions to the user. To this end, anything within the canvas can be customizable, and there is no notion of time assigned to canvas coordinates. However, Puckette provided the user with a sorting function, "on the assumption that users might often want to use Pd data collections as x-ordered sequences" (ibid., p. 185). In fact, this is the only sorting function within Pure Data, and it is the same function that sorts the patch 'graph,' only now made accessible to the user. A common and elementary database routine (`sort`) that emerged to the program's surface because of traditional music notation practices.

Puckette contextualized his research with the *Animal* project by Lindemann and de Cecco which allowed users to "graphically draw pictures which define complex data objects" (Lindemann, 1990), three cases of graphic scores used to model electroacoustic music: Stockhausen's *Kontakte* and *Studio II*, Yuasa's *Towards the Midnight Sun*, and Xenakis' *Mycenae Alpha*, and, most interestingly, the SSSP's user-defined features for graphical representations. Although in the Max papers Puckette does not quote Buxton's research, the latter's numerous publications at ICMC towards the end of the 1970s suggests that they reached the scope of MIT's Experimental Studio where Puckette studied with Barry Vercoe. Furthermore, in Puckette's later introduction of graphic scores to Pure data (M. Puckette, 2002b) (See 3.3.3), he references the SSSP quoted by Curtis Roads (1985) as one source of inspiration, indicating that at least in 2002 Puckette was aware of Buxton's research. In any case, both Buxton's and Puckette's approaches can be considered musical resonances that go beyond geographical limits, reaching the level of data structures in computer music software.

An interesting point in common, however, between much of the interactive composition programs that emerged during the 1980s is that stylistic neutrality became a leitmotif. Computer music software designers were interested in providing stylistic freedom by user-definability. This

became a programming need that stemmed from earlier computer music software implementations, and their experimentation. This shift in the course of computer music programs can be understood from two perspectives. On the one hand, by experiencing first-hand the extent to which data structures can indeed structure musical output, the composer-programmers of the 1980s took charge on data structure design and devised new approaches to music-making software. On the other hand, the novel flexibility allowed by the object-oriented model within the programming world made its way to the community by the younger generation of composer-programmers. In any case, the database was moving, expanding through computer music networks, institutions, and softwares.

OpenMusic In the same ICMC 1997 where Pure Data was presented, two object-oriented languages appeared: Realtime Cmix (RTcmix) (Garton & Topper, 1997) and OpenMusic (Assayag, Agón, Fineberg, & Hanappe, 1997). While neither real-time nor a synthesis engine, the strength of OpenMusic resides in its ability to provide the composer access to a variety of sound analysis tools for composition (Bresson & Agon, 2004; Bresson & Agon, 2010), as well as the possibility to generate algorithmic streams that output directly into a traditionally notated score. For example, OpenMusic introduced the concept of a *maquette*, which is a graphic canvas upon which a heterogenous set of elements as varied as audio waveforms, scores, or piano-roll type notation can be displayed. The LISt Processor (LISP)-based graphic language developed as a collaboration at IRCAM held music notation as a focal point, distinguishing it from other stylistically neutral software.

Heaps and Nodes Garton and Topper (1997) presented RTcmix, a real-time version of Paul Lansky's Cmix (Lansky, 1990). What they described as innovative in this project was, in a similar way to the data structures for time management that Puckette presented, the scheduling capabilities of the program. In contrast to the Cmix language, which assumes a non-real-time access of objects, "event scheduling is accomplished through a binary tree, priority-queue dynamic heap. . . " (Garton

& Topper, 1997). A heap is a tree-based data structure where both keys and parent-child relationships follow a hierarchical logic. Garton and Topper thus introduced hierarchy into the scheduler. What this allowed, in turn, was "scheduling-on-the-fly," that is, "allowing notes to be scheduled at run-time (usually triggered by an external event, such as a MIDI note being depressed)" (ibid.). The real-time problem became once again a scheduling problem of computational tasks, and it was solved differently with yet another element: instruments instantiated "on-the-fly" could also establish their own Transmission Control Protocol / Internet Protocol (TCP/IP) connection sockets in order to allow for networked access to the individual synthesizers (ibid.). That is to say, whenever a new instrument appears, it has the potential to enter into networked communication with earlier and future nodes. This means that synthesizer nodes could enter and leave the scheduler at any time, always in communication with each other. A musical equivalent would be for a violin player to enter in an out of the orchestra at will, while being able to lend the violin to any other player, and also play any other instrument except the conductor. In a similar networked way, SuperCollider (McCartney, 1996; McCartney, 1998) is a high-level language that provides the user with a different paradigm to handle audio processes scheduling. The innovation that this language implemented, however, is the "garbage collection" of each process. McCartney took the hierarchic structure of the object-oriented paradigm and defined 'nodes' in a tree-like structure, each with its own capability of nesting groups of other nodes, but most importantly, with its own initiation and expiration times. In other words, in contrast to Pure Data and MAX/MSP's constantly running audio processes, SuperCollider only consumes Central Processing Unit (CPU) resources whenever it needs to.

Both RTcmix and SuperCollider meant a step forward towards networked musical environments that have resulted in recent forms of music making such as laptop orchestras and live coding, along with new music software such as ChucK (G. Wang & Cook, 2003). The literature on computer music software for composition alone would extend beyond the scope of this dissertation. For further reference in other sound synthesis data structures, see: the Diphone synthesis

program (Rodet, Depalle, & Poirot, 1988; Caraty, Richard, & Rodet, 1989; Depalle, Rodet, Galas, & Eckel, 1993; Rodet & Lefèvre, 1996; Rodet & Lefèvre, 1997); the Otkinshi system (Osaka, Sakakibara, & Hikichi, 2002). For an overview of existing audio software up to 2004, see Xamat's PhD Dissertation (Amatriain, 2004, Chapter 2). See also the Integra project (Bullock & Coccioli, 2009; Bullock, Beattie, & Turner, 2011), and Ariza's work on python's data structures (Ariza, 2005a).

3.4 Intersections

The computer music software race that took place at the level of data structures has moved from music to media in an attempt to generalize applicability by maximizing stylistic potentials. To a certain extent, this motion can be understood as an axis between sound and music data structures. On one hand there is music tradition with its notational baggage. On the other, sound synthesis and programming, with its multi-stylistic promise grounded on the more general use of media. In any case, the shape that this motion takes is given by the composer-programmer's needs, ideas, and implementations. The computer music scene today builds on these struggles, and continues to propose novel approaches that reconfigure the practice.

In this section, I provide a glimpse of the many shapes that this reconfiguration has taken. I focus on artistic ventures, program extensions, and innovative research that has appeared under four main aspects of database performance: corpus-based approaches, querying methods, traversing methods, and resource sharing. These examples point only to some moments in which data structure design changed computer music.

3.4.1 Corpus-based Approaches

Modern uses of databases in computer music take the general form of a corpus of sounds from which descriptors are obtained and then used to create sounds. These are known as corpus-based

approaches, also known as data-driven approaches. Their difference is a matter of scale. These approaches have emerged in opposition to rule-based ones, highly useful still in many applications. In what follows, I show some implementations of the corpus-based model in sound.

Concatenative Synthesis Diemo Schwarz developed the concept of data-driven concatenative sound synthesis in his PhD thesis at Institut de Recherche et Coordination Acoustique/Musique (IRCAM) (Schwarz, 2000; Schwarz, 2003; Schwarz, 2006a). By segmenting a large database of source sounds into units, a selection algorithm is used to find any given target by looking for "units that match best the sound or musical phrase to be synthesised" (Schwarz, 2006a). In contrast to rule-based approaches in which sound synthesis is arrived at by models of the sound signal, concatenative synthesis is data-driven, or corpus-driven (when referring to larger databases). That is to say, by joining together recorded samples, Scwharz obtained a model for sound synthesis that preserves even the smallest details of the input signal. Schwarz later contextualized 'information space' as a musical instrument in itself (Schwarz & Schnell, 2009; Schwarz, 2012).

Other approaches The variety of applications of corpus-based or data-driven is still a fruitful research area. I present here only some data-driven cases that arrive at other ways to generate sounds than sample concatenation. Kobayashi (2003) used a database of Short Time Fourier Transform (STFT) analysed sounds in an original way. Upon calculating the distances between the results of these analysis he was able to define a database of similarity between his original database which he then re-synthesized. Collins (2007) developed an audiovisual concatenative synthesis method where "databases tagged by both audio and visual features, then creating new output streams by feature matching with a given input sequence" (ibid., p. 1). A recent case in which concatenative synthesis was applied to rhythm can be found in Nuanàin, Jordà, and Herrera (2016). Ariza (2003) was able to implement a model for heterophonic texture by pitch-tracking the highly ornamented

music of the Csángó[10] music into a database that enabled him to present a data structure of the ornament. The implementation of analysis and subsequent algorithmic rule extraction can be thought of as a form of analysis-based sound generation: by inputting a sound file, a dataset of rules was obtained to approach a model for the ornament. Therefore, a rule-model was obtained by means of a data-driven approach. This relates to *Orchidée* (Carpentier, Tardieu, Rodet, & Saint-James, 2006), a computer-aided orchestration tool based on database input-matching and a series of candidate orchestration targets. The data-driven approach is combined with a highly dense corpus of instrumental techniques, in order to concatenate orchestral targets.

Software Libraries One of the central concepts of the object-oriented programming is extensibility. The list of objects that can be added to the main program tends to grow exponentially as a function of its use. A list covering all extensions would require a research project of its own. However, I would like to focus on those extensions that enable further and more specific use of databases in the context of music composition. B. Sturm (2004) developed *MATCONCAT*, a concatenative synthesis library for *Matlab*. Schwarz (2006b) designed Real-Time Corpus-Based Concatenative Synthesis (CataRT) as a concatenative synthesis toolkit both as a standalone application and as a MAX/MSP external. Another concatenative synthesis library is Ben Hackbarth's python module AudioGuide. William Brent's research on timbre analysis developed into a timbre description library for Pure Data called `timbreID` (Brent, 2010). Within this library, users are able to analyze sound files using most available timbre descriptors. Since Brent's library enables users not only to analyze sounds and store the resulting descriptors in a database, but also to cluster them within the database, it allows for a variety of applications of which only one of them is concatenative synthesis.

[10]"The Csángó, in some cases a Szekler ethnic group, are found in eastern Transylvania (Kalotaszeg), the Gyimes valley, and Moldavia" (Ariza, 2003).

3.4.2 Querying Methods

Query-by-content One of the innovations that brought forth Music Information Retrieval (MIR) is high-level audio feature analysis. This enabled computers to understand keywords such as 'bright', 'sharp', 'dark', 'metallic', etc., that would describe timbral content of audio files. When applied to database querying, these keywords enable 'query-by-content' searches. Many online databases such as Freesound or Looperman.com (Looperman) have this type of querying. The Content Based Unified Interfaces and Descriptors for Audio/music Databases available Online (CUIDADO) project at IRCAM consisted of a database system aimed at content based querying of sound files (Vinet, Herrera, & Pachet, 2002a; Vinet, Herrera, & Pachet, 2002b; Vinet, 2005). This project enabled Disk Jockeys (DJs) to browse through files, apply beat-synchronized transitions between them, among other automated tasks during performance. CUIDADO later developed into the *Semantic Hi-Fi* project and influenced subsequent software. Norman and Amatriain (2007) enabled users generation of personalized audio description databases that could also be queried by content in *Data Jockey*.

Similarity-based Frisson (2015) provides an overview of multimedia browsing by similarity. Real-time audio analysis moved users beyond descriptive keyword, with sound based input by singing or by providing a sample array. These systems calculate the spectral similarity between the incoming signal to obtain a match from a sound database. In this sense, a different type of performativity was enabled with systems with query-by-content in live contexts. For example, *SoundSpotter* (Casey & Grierson, 2007) was dedicated to real-time matching of audio-visual streams by using audio input as feed for a shingling algorithm based on Log Frequency Cepstral Coefficientss (LFCCs).[11]

[11] "Audio Shingling is a technique for similarity matching that concatenates audio feature vectors into a sequence of vectors, and matches the entire sequence" (Casey & Grierson, 2007). "Shingles are a popular way to detect duplicate web pages and to look for copies of images. Shingles are one way to determine if a new web page discovered by a web crawl is already in the database" (Casey & Slaney, 2006).

Querying a database by similarity appeared in contexts other than performance workstations. Based on both the *Semantic Hi-Fi* and *SoundSpotter* projects, Price and Rebelo (2008) developed an installation with an interface to a relational database of percussive sounds.[12] This database contained description data of the beginning of each analyzed sound file. Thus, participants were able to query a bank of percussion timbres based on brightness, noisiness, and loudness.

Concatenative synthesis uses similarity for the purpose of sample concatenation at the analysis frame level. In this sense, concatenative synthesis can be understood as a real-time query-by-content engine feeding a granular synthesis engine. Therefore, the difference between concatenative synthesis and content-based-queries is a matter of scale (samples as opposed to sound files) and in the use (new sample combinations as opposed to previously stored sound files).

Hybrid Queries Some authors have managed to conjugate disparate database uses by hybridizing the queries. The following modal translations represent only some of the many examples in the literature. Schloss, Driessen, and Peter (2001) used audio analysis to obtain gesture features from the non-audio signals obtained from the *Radio Drum*, an variant of Max Mathews's *Radio Batton* (Boie, Mathews, & Schloss, 1989). Schloss et al. (2001) searched for peak detection in the incoming signal to determine mallet (air) strokes. At Center for Computer Research in Music and Acoustics (CCRMA), Serafin et al. (2001) managed to invert the concept of physical modeling by estimating violin bow position, pressure, and speed using Linear Predictive Coding (LPC) coefficients of violin audio recordings. Oliver and Jenkins (2008) presented a controller composed of an elastic head suspended along the rim of an empty drum shell. The player presses the head making different shapes with the hand, fingers, or mallets, and "these shapes are captured by a video camera that sends these images to the computer, which analyzes them and outputs the tracked parameters" (ibid., p. 1). This instrument enabled a possibility for "dissociating gesture with sound" (ibid., p. 1). That is to say, gestures whose sound could be visually anticipated (the

[12] In their project, they used a MAX/MSP library called *net.loadbang-SQL* to query and import data for the communication with Structured Query Language (SQL) databases.

hitting of a drum) were extended by micro-gestures only visible to the camera sensor. Further, in contrast with acoustic drums, with the silent drum "one can manipulate continuous sounds through a new gestural vocabulary (ibid., p. 1). Oliver La Rosa developed the sensing algorithm into a Graphics Environment for Multimedia (GEM) external called `pix_drum`, and later moved on to `pix_mano`, where he removed the fabric and focused on what he calls "direct hand-tracking" (Oliver, 2010) (See 4.4). Caramiaux, Bevilacqua, and Schnell (2011) proposed gestural input for the query-by-content method. They used gesture-to-sound matching techniques based on the similarities of temporal evolution between the gesture query and the sound target. Another example of hybrid querying is Cartwright and Pardo (2014), where a database of computer synthesis parameters was queried by vocal input, enabling users to mimic sounds with their voices in order to obtain parameter settings (presets) that would approach the analyzed vocal sound.

3.4.3 Traversing Methods

Given that querying methods have resulted in novel ways to approach information space within databases, many authors have proposed their own approaches towards navigating this space. Like browsing, or surfing the Internet, database traversing is a form of navigation across the *n*-dimensional space that databases have to offer. Despite their differences, the approaches I refer to now point to the hybrid qualities that data can take when used in performance, specifically in terms of the mixed use of data coming from multiple sensing mechanism, and the networked quality that reconfigures music performance and composition.

Sensorial Networks Insook Choi, Zheng, and Chen (2000) presented an interactive installation (Choi, 2000) at the Dorsky Gallery in NYC where a 'sensorial network' made from a sound database of speeches by famous leaders was distributed along the installation space. Choi et al. implemented a motion tracking computer vision algorithm enabled sounds to be modulated as a function of the different 'clouds' of pixel data where values gradually changed as participants

moved across the sensing area: "pixels do not switch on and off, they fade in and out forming clusters in the 2D camera plane according to the degree of movement projected from the corresponding floor positions" (ibid., p. 4). In this sense, participants were able to walk the database itself: "Traversing the [sensorial network] can be thought of as rotating its shadow such that one moves through a semantic neighborhood which includes sound synthesis and residual tuning as well as speech acts" (ibid., p. 3) In addition to this tracking system, however, she included hysteresis within the system. Thus, the recorded history of the participant's interaction with the system enabled condition-dependent events to occur as participants' interaction lasted longer. Within this installation, the artist prototyped a "sensory information retrieval system where the acquisition of information is an acquisition of an experience" (ibid., p. 1).

Involuntary Navigation Bioinformatic data taken from galvanic skin sensors attached to a cellist's toes within a live performance environment is the point of departure for a complex network for performance (Hamilton, 2006). The Galvanic Skin Response (GSR) activity was correlated with intervallic distance between adjacent musical notes in a database of 'cell nodes' previously written by the composer. However such score acted as a "filter for the autonomic control signals generated by the performer" (ibid., p. 601). What this means is that the music fragment database, involuntarily navigated by the performer, becomes a parameter for a live-generated score. The performer is thus embedded within a convoluted networked loop that goes through voluntary and involuntary agents that intertwine composition, interaction, and performance.

Networked Collaborations Among the many cases of network performances with multiple players that exist in the literature, I would like to point to one case where the rules of 16th century counterpoint demanded a relational database (Nakamoto & Kuhara, 2007). By implementing a database system such as MySQL, to store and retrieve vocal parts, Nakamoto and Kuhara enabled performers to sing together in canon form from distant locations. Going beyond any notion of

anachrony, what is interesting about this approach is the fact that by "using a PC and database server with the internet" two or more performers can engage seamlessly in musical performance (ibid.). Telematic performances have spawned ever since Internet connectivity enabled networked audio and video feeds. Whalley (2014) considers that the listener's body within telematic electroacoustic concerts has been traditionally left out. Therefore, in he devised a set of parameter constraints within these performances, based on musicians who were used as baseline. His argument was grounded on an affective approach towards networked performance, and it is aimed at addressing the limitations that arise from the separation between performer and listener, specifically within telematic electroacoustic performances.

Mobile Devices The mobility that networks enabled can be represented in the work of Liu, Han, Kuchera-Morin, and Wright (2013), who created an audiovisual environment for live data exploration that implemented simultaneous sonifications and visualizations of networked database queries made by participants using I Operating System (iOS) devices. Taylor, Allison, Conlin, Oh, and Holmes (2014) implemented centralized database systems to include user-defined interfaces to be saved and shared within their mobile device platform. Carter (2017) presented a work that gives each member of the audience their own instrument through their cell phones. By accessing a website that loads custom synthesizers made with the Web Audio Application Programming Interface (API), the audience becomes the performer in an innovative way. While the title of the work ("On the expressive potential of suboptimal speakers") refers to the potentials and the ubiquity of small transducers, the 'score' (source code) of the work lives on a server and travels wirelessly to the audience to become a (mobile) instrument.

These are some of the many examples that point to the many shapes that traversing a database can take. These shapes have given different resonances within the concert and installation spaces, as well as within the performativity of the music involved. Further, the possibilities of these reconfigurations can be seen in terms of a need for sharing resources and experiences through

networks.

3.4.4 Resource Sharing

Sharing resources can be interpreted in many ways. On one end, it points to networked environments on which multiple client users connect to a server that provides shared data flow among the network. This is the case of live coding, where multiple users share the same network. Another definition pertains to the data itself, the way that it is formatted, and how to access or edit it: the file format, where users can read the same data. Lastly, the activity of sharing relates to publishing results like in research or academic communities. This is the case of the multiple datasets that exist.[13] In any case, what is common between these forms of sharing is an entropic and endless plurality.

Multimodal Datasets Among the many datasets that are available (See 3.1), a research interest has been growing among gesture datasets. This is the case of a hand drumming gesture dataset that uses data from a two-dimensional pressure sensor that could be compared to the membrane of a drum (Jones, Lagrange, & Schloss, 2007). Jones et al. aimed to provide physical model designers with a collection of six techniques of hand drumming, recorded as matrices at a slow rate (100 Hz) suitable for non-real-time synthesis by way of interpolation into a model for physical modeling of wave propagation called 'waveguide mesh.' Andrew Schmeder (Schmeder, 2009), stemming from the research at Center for New Music and Audio Technologies (CNMAT) on the Open Sound Control (OSC) format, proposed a real-time application for efficient storage and retrieval of gestural data using the relational model offered by the PostgreSQL Database Managemet System (DBMS). The motivation behind these datasets, besides research is mostly to provide open access to any user with a computer and an Internet connection. Young and Deshmane (2007) created web-accessible databases of gestural and audio data concerning violin bow strokes. Hochenbaum,

[13] 'Dataset' differs from 'database' in terms of scale: multiple datasets may reside in a single database.

Kapur, and Wright (2010) developed a gestural and audio joint database that enabled identification of a given performer between a group of performers, gaining insight on musical performance itself. These joint databases combining more than one sensing mode are called 'multimodal.' Multimodal databases can be extremely focused —combining different blowing profiles on recorder flutes (along with their sound) (García, Vinceslas, Tubau, & Maestre, 2011)—, or radically plural: listening subjects asked to move as if creating a sound (Visi, Caramiaux, Mcloughlin, & Miranda, 2017).

Formats While the purpose of a format is to store as much information as possible, using as little space possible, and in an efficient way so that read and write operations occur seamlessly, formats are the equivalent of database models within files: they can be implemented in endless ways, and they are contingent upon programming decisions (Sterne, 2012, p. 8). One way to categorize formats is based on human readability. Readability of the format is a function of the task at hand and the quantity of the data involved. In cases where the data is very little, for example, a `.pd` file (Pure Data), MetriX in Extensible Markup Language (MetriXML) (Amatriain, 2004), JavaScript Object Notation (JSON), YAML Ain't Markup Language (YAML), or `.bib` (LaTeX bibliography file), data structures can be stored in (text) characters, and thus be readable by humans. In this sense, data does not need to be highly structured. For example, within the *Integra* project, programmers implemented a data format called Videoalive Indexer Extracted Closed Captions and Metadata (IXD), capable of containing sequences, tags and meta-data, and presets, for shared use among different multimedia environments. Their argument for a semi-structured model resided in the semantic richness that can be allocated in opposition to the binary format only readable by machines. To this end, they implemented IXD using the Extensible Markup Language (XML) language (Bullock & Frisk, 2009). In other cases, data is large enough to justify the need for binary format with a simple header such as `.timid` (**`timbreID`**). At this level, by structuring the format and sacrificing human readable semantic richness, faster write and read times are achieved,

and less resources are used. However, in the case of larger media files such as audio, image, or video, and also multimodal gesture data, these demand high-performance compression algorithms that reproduce data in 'streams.' Some formats for sound and gesture analysis were standardized in recent years, as is the case of Sound Description Interchange Format (SDIF) and Gesture Description Interchange Format (GDIF), which are widely used in audio analysis software like Sinusoidal Partial Editing Analysis and Resynthesis (SPEAR) and OpenMusic (Bresson & Agon, 2004; Nymoen & Jensenius, 2011). In one case revealing the extent to which data can reside in multiple combinations, the SDIF format was used for audio spatialization data (Bresson & Agon, 2004). There is still little format standardization within datasets, and in general, the plurality of formats demands database creators to either implement routines to interpret as many formats as possible, or to rely on external libraries for transcoding. In any case, the plurality of formats is almost as great as that of datasets and, to a certain extent, almost as numerous as there are software developers.

Live Coding Live coding has now a long history and it occupies a fair portion of the computer music scene today. In terms of database performance, the practice of live coding in audio or in video exposes both computer technology and art performance in simultaneity to the cutting edges of both worlds. For a more general overview on live coding, see (Collins, 2006; Collins, Mclean, Rohrhuber, & Ward, 2003; Nilson, 2007; m zmölnig & Eckel, 2015). In this brief section, I would like to point to the work of Roberts, Wright, Kuchera-Morin, and Höllerer, 2014, who implemented within a real-time live coding web-based environment called *Gibber* a centralized database for the storage and quick access of digital instruments that can be prototyped in the environment. This type of on-the-spot database system enables shared access to sound files that have potential use throughout the performance. By means of a networked database, two or more players can grab and record sounds from different locations. Another case of networked situations in live coding is a system that incorporates content based searches (query-by-humming, query-by-tapping) of various Creative Commons (CC) sound databases such as Freesound or to user-defined databases (Xambo,

Roma, Lerch, Barthet, & Fazekas, 2018).

3.4.5 Closing Remarks

The many shapes that database performance has taken over the years can be approached with what I have shown so far. Since many applications of the database in music continues to grow, I have only selected a few areas in which the database has had some agency. One area that I have not included above is that of artificial intelligence for music applications, where databases have been used for training models, and other forms of machine learning. For example, in interactive music systems (Rowe, 1992), in improvisation systems (Assayag, Dubnov, & Delerue, 1999; Bloch & Dubnov, 2008), to model Electronic Dance Music (EDM) patterns (Vogl & Knees, 2017), analog synthesizer parameter settings (Loviscach, 2008). Multimodal datasets have also been used in training (Schöner, Cooper, Douglas, & Gershenfeld, 1998). Notwithstanding the multiple gaps and omissions that these lines reveal, I believe the plurality of shapes speaks for itself. As I have shown, from bytes to terabytes, from data structures and files to datasets and databases, has had different positions in relation to sound practices. These positions can be summarized in a three-dimensional diagram (See Figure 3.7) where the database can be placed along three axes. Visibility of the database can be represented by the sign: positive values indicate visibility and negative indicate invisibility. 'Negative' in this context relates only to the sign of the value, and not to any 'judgment' whatsoever. If anything, this graph is intended as a metaphor. As I have mentioned earlier, the database is generally the grounds of sonification, so it can be represented by the y-axis. In MIR, the database is next to the databaser, that is, in the x-axis. Finally, I mentioned that the database in computer music is behind the databaser, therefore the z-axis seems appropriate.

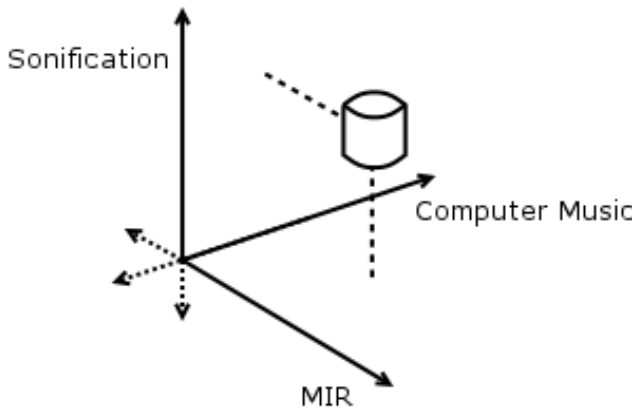


Figure 3.7: Intersection space.
A snapshot indicating a possible position of the database in terms of visibility among MIR, sonification, and computer music. Positive values indicate visibility and negative indicate invisibility. This graph is only one of multiple frames in which we can capture the visibility of the database in relation to the different practices. Despite the fact that the location in which the database can be placed within this graph is reduced to a single point, the database can occupy multiple points simultaneously. Thus, the more complex coreography inherent in databasing escapes the bi-dimensionality of this graph, but it can nonetheless be hinted by it.

The simplicity of this diagram is intentional, to avoid any attempt to quantize the actual value that the database represents in the plurality of shapes that I have discussed. There is no percentage that can be drawn from how visible a database can be. Therefore, when practices begin to intersect, as I have shown here, the visibility of the database can thus be understood as in constant motion along these axes. Database performance, in this sense, provides a key to understand the motion of this intersection. Furthermore, there is one dimension not contemplated within this diagram: time. The intersections referenced here are always moving in time, which indicates that the diagram that I have shown here is but one frame in the movement of this database choreography. At each point in time the databaser can pause for a second, analyze the frame, and perhaps describe the motion that the database has taken thus far, and the position that it occupies at that point. And further, perhaps the database occupies two positions at once. The dimensionality reduction carried out when graphing the position of the database should not misguide the reader into thinking that the database is fixed, or that it may have only one position. Quite the contrary, the

database is in constant motion and incessant fluidity, and databasers engage in a form of dance that sometimes brings the database to a certain visibility, and other times might veil it beyond layers of code. In any case, this description has been my task until now, and it is safe to say that we have glanced at the database dance. In what follows, I will change the focus and approach the database from a different perspective, one not guided by light, but immersed within sound.

Part II

Database Aesthetics

A person that encounters a database feels a resonance, recognizes some of her or his bodily and mental functions in the artwork. In this part, I explore a way in which we can experience database art as a mirror or echo of ourselves, yet at the same time, we recognize the presence of something different from ourselves that guides and structures our experience of that work. This unclear presence, like a specter, often reveals through a combination of a variety of databases that can be found at the intersection of art and technology. This specter is what I consider the key to understanding how databases claim a certain aesthetic agency that is underway when computers are involved in art, particularly in the field of music composition. Therefore, by addressing a specter of the database, I take on the adventure of outlining a framework to discuss an aeshtetics of database music. To this end, I propose an aesthetics that is not humanly derived but that it is still dependant upon human temporalitites. In other words, while databases operate below human thresholds of perception, we can still speak of an aesthetics of the database precisely because we can perceive the presence of the database. Therefore, we can understand certain aspects of what is perceived of database music works as comprehensive of a working definition of aesthetics that I will only suggest throughout this text. The reader might perhaps find these ideas about aesthetics as a starting point to further questions and developments about the deeper resonances database music proposes.

A restless encounter at the limit Databases can be understood as certain restless creatures that exist and participate in the world in ubiquitous ways. This restlessness can be understood as the dynamic feeling we have of the database as not being settled in one place and of being, at once, ubiquitous and in constant motility. I begin to delineate this restlessness in the previous sections, by suggesting the kind of motion that both visibility and development of the database has been taking place over the years in and between sound practices with computers. In this part, I propose to deepen the understanding of this restlessness, suggesting it now as a starting point to understand an aesthetics of database music. To this end, we can consider this restlessness as being

a result of the encounter with the uncanniness of a spectral force, at the limit, or in the moment of a certain 'touch' of this liminality in between the human and the nonhuman. An aesthetics of database music can thus be approached from this restless encounter at the limit. The uncanny feeling that results from an encounter with the spectral speaks of an experience with that which is familiar and, at the same, not familiar to us; known to us as being close to us but, the moment we recognize this closeness, it slips away from our proximity into an ungraspable feeling that shakes our comprehension of reality. Thus —and through its performance—, the presence of the database within a database music work makes its ghostly appearance, and reconfigures our understanding of the composer (but also the programmer, the performer, etc.) into a hybrid 'databaser' comprising both human and nonhuman aspects. This specter is an illusory figure that is imprinted upon what I call the (infra) skin of the database, and it is the source of our encounter with the uncanny. As such, this source suggests a certain agency within the database music work that resonates differently in the production and reception of the work. On the one hand, the productive force of this agency can be summarized with the way in which small changes to a database have significant effects on not only the resulting music work, but also the operativity throughout that music work, and thus they lead to new and uncharted artistic territories. I understand these territories to be 'worlds' that databases reveal: sets of possibilities, but also the creative potentials awakened in the producer. On the other hand, the social, ethical, and political resonances that database music works have upon reception, reveal the tensions that exist between control and ungovernability in databasing. These tensions can serve, in sum, as a metaphor for a certain contemporary subjectivity in which we are embedded by the ubiquity of the database in most aspects of our lives. Thus, the aesthetics of database music evidences a limit between the human and the nonhuman, as well as between the self and community; a limit that is in no way fixed, but that exists as a textural play of resistance and expansion. What the aesthetic agency of the database might suggest, then, is new approaches to our own mundane activities that are reconfigured by the awareness of the presence of the database that, as listeners, we obtain from database music works and the encounter with the spectral.

Chapter 4

Listening Databases

4.1 Interlude: I Am Sitting In A Room...

I am sitting in a room*mm*

diff~~e~~rent from the one you are in now

inhale (long)

I am re~~c~~^{kh}ording the ~~*s*~~^{ss}ound of my*hh* speaking ~~*voice*~~^{voisss}

inhale (short)

and I am going to play it . . . backin-to the room*mmm*

again*nnn* ~~*and*~~^{an} again*n*

inhale (long)

un~~*t*~~$^{tx!}$il the

re~~*s*~~^{ss}onant*thh*→*ff* frequencies of-the-room

inhale (short)

reinfor~~*c*~~^{sss}~~e~~ them~~*s*~~^{ss}elve~~*s*~~^{sss}

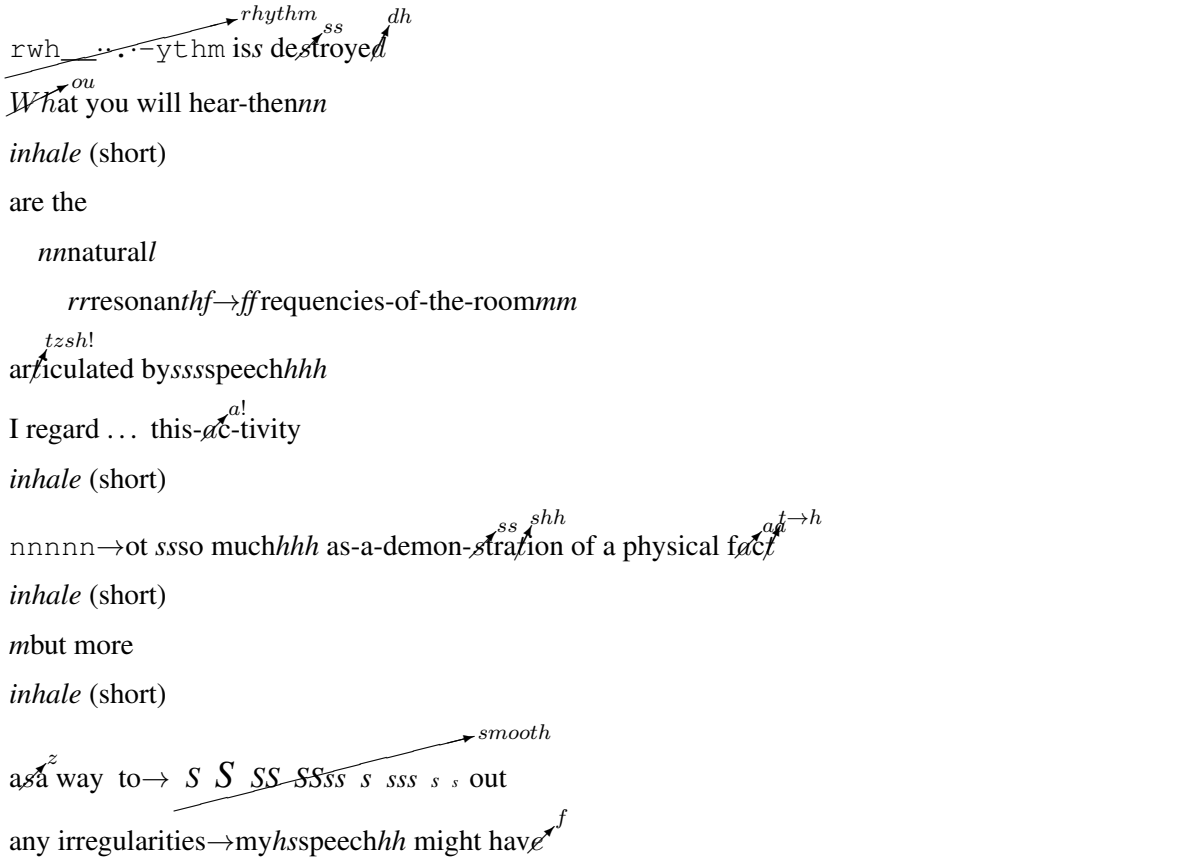
inhale (long)

so that any

sss→emblance of my ~~*s*~~pee~~*ch*~~^{sss}$^{ch→shwh}$

inhale (long)

with per~~*h*~~^{hhhh}ap~~*s*~~$^{s!}$ the exce~~*p*~~^{shh}tion of



Transcription made from a recording of Alvin Lucier's work "I am sitting in a room" (Lucier, 1970). It is meant to notate the many "irregularities" that differ from the text.[1]

4.2 The Resonance Of A Return

A Reverb Sound reaches, enters, and traverses bodies in media. Media here refers to the matter through which sound propagates, such as a space filled with gas, liquid, or solid particles of matter including human and nonhuman bodies. More generally, sound propagation is conditioned by the qualities of the medium. Waves change direction by way of reflection or refraction, and they fade out by way of attenuation. Furthermore, while the combination of density, pressure, temperature,

[1]The recording is available online and it was made at Lovely Music, Ltd (1981) https://www.youtube.com/watch?v=fAxHlLK3Oyk

and motion affect the speed of sound, a medium's viscosity affects the sound's attenuation rate. For instance, within hot and humid climates sound will move slower than in cold and dry climates; or, if there is wind blowing in the same direction of a sound, it will make the sound travel faster. This means that sound waves are affected in different ways by different media, some being more (concert halls) or less (anechoic chambers) resonant.

A Filter A listening body is part of the medium through which sound propagates: the body's sense perception is immersed within that medium. Sound, in its most basic and general form makes listeners vibrate as listeners become part of sound. Being part of sound, bodies change sound even before listening. On a mechanical level, we can think of the body as an a priori physical filter. Sound is filtered differently and uniquely within each body: my body changes the incoming sound for me, just as it does for others. In other words, a longitudinal wave passing through a body affects how it will arrive at other points in space. Therefore, bodies filter sounds for other bodies while affecting sound waves before they reach the listener. That is to say, since the listener's body itself refracts, reflects, and attenuates waves, the singular filter that is the body changes wave propagation not only for itself and its own listening experience, also for the listening experience of others. Empty concert halls are thus more reverberant than filled ones.

A Loop The filtering qualities of the listening body reveal the extent to which listening is such a singular and personal experience. Furthermore, this singularity can be understood as emerging out of the plurality that is sound. Plurality, in this sense, refers to the infinitesimal activity of waves. The interaction between the singular and the plural, in this sense, can be approached with the structure of a loop. I listen to myself as resonant subject, while creating meaning from a certain quality of a sound. I do this in simultaneity with others, who also create themselves as resonant subjects, while giving meaning to other sound waves. In this resonance, the vibrating link in between ourselves is also simultaneously changing the way we are listening. Thus, every singular

listening subject is in a state of being (mutually) (self) exposed to every other listening subject, that is, in resonance or in touch with one other.

An Attack Philosopher Jean-Luc Nancy (2007) brings forth an ontology of sound that can be understood in terms of resonance. He speaks about a "sonorous presence" (Gratton & Morin, 2015, pp. 143–144) that exposes listeners to themselves and to one another. The duration of this exposure is always an instant. All mechanical waves require an initial energy input and in the case of sound, particularly in musical contexts, this input is generally referred to as an attack. Instead, Nancy uses this term to describe the exact moment when a sound arrives and simultaneously leaves the body: the instantaneous appearance of sound within the body. An attack therefore instantiates the sonorous presence. Within this attack, that is, during the experience of this exposure, sound is understood as a sensing experience in itself as well as the experience of what a given sound might signify. As Brian Kane writes, to be listening in the sonorous presence constitutes "a mode of listening that exposes itself to sense" (ibid., pp. 143–144). This means that in the sonorous presence, the body begins to listen to itself listen. As I described with Hansen's notion of virtuality (See 1.6), virtuality can be understood as our brain's capacity to create images from the world. In listening, the virtuality of the human mind engages with the attack of the sonorous presence. In this sonorous present the first 'image' is the body as such. In this sense, the creative capacity of the body enables the body to be *self*-in-formed during the sonorous present. Furthermore, the body listening to itself listening results not only in the self-image of the body, it also creates an image of the listened.

A Sampler Consider, for example, an acousmatic concert in which one of the music works is made with pre-recorded violin samples. When this violin begins playing sounds, an illusion may very well begin to emerge: we can feel a violin player. If this virtual player continues to play sounds and moves them in space, this illusion continues in the direction of a physical but illusory motion in

space, that is, we can perceive an actual violin and an actual violin player. Our aethetic experience becomes a virtual experience. Therefore, this virtuality may project itself throughout the complete music work, thus grounding the music work on an affective force that is only *there* because of the listener's own capacity for virtuality. The ghostly qualities of this force will be addressed further down this text (See 5.3). Most presently is the fact that this 'magic' show can be understood in terms of a resonant link between the human and the nonhuman: a web of interconnected objects that refer to each other. In this listening process, therefore, the listening subject exposes itself not only to itself, but also to a virtual self of the listened.

A Texture Since every 'body' is immersed within sound, and since sound refers to the thing that makes it, for Nancy this immersion is within a web of references. Furthermore, this web of references moves like waves: in time and space, back and forth, delaying in every next moment and distinguishing in every other reference. Therefore, instead of a loop, a more convoluted circuitry appears that can be understood as a Feedback Delay Network (FDN). The trick here is that this delay network sounds without input or output: it is already playing and sounding as a web-like endless texture. Brian Kane refers to this structure as "a structure of infinite referrals and deferrals" (ibid., p. 143), where references are at once postponed or delayed, and distinguished from each other. Within this 'texture,' Nancy approaches a notion of meaning: "meaning is made of a totality of referrals: from sign to a thing, from a state of things to a quality, from a subject to another subject or to itself, all simultaneously" (Nancy, 2007, pp. 4–9). Therefore, since sound "is also made of referrals: it spreads in space, where it resounds while still resounding 'in me'" (ibid., pp. 4–9), the result can be understood as a process that intertwines sense and signification. If this is the case, then sense refers to the body sensing itself sensing, and signification points to the referential quality of the texture. In both cases, what is at stake in the listening experience is this interconnected web-like texture of delays and distinctions. On the one hand, the points in this texture are distributed in time, delayed to further moments. On the other, these same points

are marks that indicate the extent to which they differ or resemble each other, thus they can be understood as a spatial distribution of references.

A Return For Nancy, to be listening is to enter into "tension" and to be attentive for a relation to self (12 ibid., All subsequent quotes from this passage.). In this tension, 'self' refers neither to yourself —"not . . . a relationship to 'me' (the supposedly given subject)"—, nor to the self of another —"the 'self' of the other (the speaker, the musician, also supposedly given, with his subjectivity)." The structure of resonance can be understood in terms of a "relationship in self." That is to say, because of this relationship (in self) that appears in the play of the web-like texture of delays and references, to be listening is an ontological passage: "passing over to the register of presence to self." The self appears, it becomes present, as something that emerges from a resonant plurality. However, 'self' is not an expressive substance inherent to bodies, or already in the body, as if it were some originary essence that appears out of resonance. For Nancy, the 'self' is "nothing available (substantial or subsistent) to which one can be 'present."' On the contrary, the self comes in the form of a return, the "resonance of a return [*renvoi*]"

The more general implications of this ontology would extend the limits of this dissertation. For a commentary on Nancy's work, see Gratton and Morin (2015). Nevertheless, I would like to point to one particularity of this ontology of sound: listening is an activity of sensing bodies through which their ontological condition becomes available. In this sense, to what extent can we consider the database as a listening body? And if so, to what extent is there an ontology of the database? These are the questions that I address during the following sections.

4.3 Resonant Network

The Recorded Movement of a Thing Philosopher Bruno Latour (1990, 1993) developed a theory of networks called Actor-Network Theory (AT). AT can be understood as a way of connecting

and associating entities to one another. It is a tool that builds an image of the world made of nodes along a decentralized web of meaning. Latour reformulates nodes and edges, with what he calls 'semiotic actors,' 'actants,' or 'agents' (nodes) and of the interconnected accounts that these have of each other (edges). As I have mentioned earlier with the network model in databases (See 2.4.2), navigating through networks is traversing from node to node. However, the (visual) two-dimensional metaphor of a 'network' as a 'surface' limits the understanding of its topology: "instead of surfaces one gets filaments." (Latour, 1990, p. 3) In this sense, he points to a misunderstanding that comes from giving AT a technical definition such as the one described with the database model: "nothing is more intensely connected, more distant, more compulsory and more strategically organized than a computer network" (ibid., p. 2). AT points towards a topological shift. Nevertheless, the navigational paradigm advanced by Bachman (1973) in relation to databases whose 'keys' become *n*-dimensional space, does translate well to Latour's model, because nodes have "as many dimensions as they have connections" (Latour, 1990, p. 3). In any case, what this navigation points to is that AT is comprised entirely of motion and activity: "no net exists independently of the very act of tracing it, and no tracing is done by an actor exterior to the net" (ibid., p. 14). Thus, meaning and connectivity are enabled by the activity or work of actors: "In order to explain, to account, to observe, to prove, to argue, to dominate and to see, [an observer] has to move around and work, I should say it has to 'network"' (ibid., p. 13). This work, *net*-work, or 'tracing,' is not only the movement of associations and connections, it is also the 'recording' of this movement. In this sense, Latour claims that "a network is not a thing but the recorded movement of a thing" (ibid., p. 14). Furthermore, nothing falls outside the network: "the surface 'in between' networks is either connected —but then the network is expanding— or non-existing. Literally, a network has no outside (ibid., p. 6)." The network encompasses its own actors and its own expansive motion. Most importantly, AT is a tool aimed at describing the nature of society. However, in this description, AT "does not limit itself to human individual actors but extend[s] the word actor... to non-human, non individual entities" (ibid., p. 2).

Howling Thinking networks in terms of a (sonic) three-dimensional metaphor (as a mechanical wave) is thus misunderstanding it. Coupling 'resonant' and 'network' results in a sort of (impossible) positive feedback. While a network expands in redundancy, overflow, accumulation, and self-reference, a sound attenuates towards imperceptible and infinitesimal thresholds. Lending an ear to the sound of AT we would find ourselves listening to expanding filaments. However, as an acoustic experiment that would combine the circuitry of a feedback with the accumulative quality of networks, I propose to consider Lucier's (1970) "I am sitting in a room". I have transcribed this sound art piece at the beginning of this chapter, as it is remarkably self-explanatory (See 4.1). It can be understood as a triple crossfade, first between speech and music, gradually crossfading into a second crossfade, between timbre and space. Through the circuitry of a closed and controlled feedback loop between a microphone, a speaker, and a room. More considerations of this work I will leave for some other time, and I will refer to Valle and Sanfilippo (2012) for further readings on feedback systems. What I would like to bring here is an experimental revision of what is known as the Larsen effect: "in every sound reinforcement system, where the loudspeaker and the microphone are placed in the same acoustic environment, a certain amount of the signal radiated by the loudspeaker is fed back into the microphone" (Kroher, 2011, p. 11). When these systems become unstable, the Larsen effect appears (also referred to as 'howling'), "resulting in a positive feedback producing pitched tones from the iterated amplification of a signal" (Valle & Sanfilippo, 2012, p. 31). Therefore, in Lucier's room, what occurs is quite literally the Larsen phenomenon, but "stretched in time," and thus the "room itself acts like a filter" (ibid., p. 34). Considering the mechanical contradiction in thinking resonant networks, I believe it necessary, then, to expand the 'mechanical' side of the feedback system in question: Lucier's *room* needs to be expanding as well. As a consequence, the "resonant frequencies" (nodes) of the expanding network would cease to "reinforce themselves." However, (and here is the experiment) this does not mean that these nodes would cease to act, let alone resonate. In this sense, we can ask ourselves how would *this* sound like? *Where* would the 'I' be actually *sitting*?

The Resonant Movement of a Thing Such a feedback network would redefine the notion of a temporal delay into a *spatial* delay. Instead of the Larsen effect being "spread in time," in Lucier's work it would also spread through space. The room as a filter would resonate differently because it would be understood as a texture, a networked resonance. If Latour's semiotic actors are in constant reference to each other, it can be argued that they are in resonance with each other, in a permanent state of vibration, or simply, *listening*. Thus, Latour's phrase can (perhaps) be reformulated: *the network is not a thing, but the resonant movement of a thing*.

I am sitting in a loop This is the crucial leap that comes out of the idea of a resonant network: the moment the nonhuman in the network is comprehended as resonant, it is the moment that they engage with an approach to self (in Nancy's terms). Following this logical thread, a database can be considered as a semiotic actor as well as a resonant subject. On one hand, databases are not just networks, they are actor-networks: acting, tracing, and listening. On the other hand, since databases are indeed listening, to what extent can we think of them as listening to themselves listening? Bringing back Lucier and his room, I would like to address this question with another aspect of this work. (And by 'work' I begin to introduce an important aspect of this dissertation, a concept that embraces activity, productivity, but also product, and objects: operativity and opus.) There is indeed a fourth crossfade, between the 'I' in the text, and the 'I' in the voice that reads it. The simplest way to approach this is by asking ourselves, if after recording the first input signal Lucier remained seated *in* the room or not. This is a difference that cannot be approached from the recording itself because it is inaudible. I will refer to this difference further down this text. For now I point to the fact that the moment Lucier recorded his voice, the 'I' began residing *in the loop*. I believe this is one of the most crucial 'irregularities' that can be found throughout the work. In the interlude at the beginning of this chapter, I transcribe the text as Lucier reads it. I attempted to be as clear as possible, crossing out, replacing, extending all the consonants into what I thought was a more faithful score for the read fragment. I resorted to these words being "under

erasure," that is, "to write a word, cross it out, and then print both word and deletion" (Derrida & Spivak, 1976, p. xiv), because in this way one would have a visual cue of both the instruction and a more verbose inscription of the voice. (To a certain extent, *I am sitting in a room...* can be understood as a music work 'under' constant 'erasure.') Thinking as a composer, this score (with all its notated irregularities) would explain the first minutes so fiercely that the mystery of the last minutes would be solved. But, the most crucial aspect of the piece cannot be rendered in symbolic transcription, because the 'I' is somewhere in between the transcribed and the inscribed (See 5.3). This 'I' is what is at stake when databases begin to resonate, that is, it is the approach to this notion of 'self' what begins to redefine ourselves in general. That is to say, within the resonant network we face a 'self' that changes our own notion of 'self' in general. In this sense, a 'self' *sitting in a loop* returns to us (resonates back) putting into question a relationship (a difference): what is the difference between the two 'I's? Is is this same difference at stake between the human and the nonhuman? The implications of these questions I will move forth in the remaining sections of this dissertation. However, the most present step is analyzing the conjunction that the two clauses of the question points to: the sharing of the 'I,' an exposure of community.

4.4 The Unworking Network

Community as unwork As I have described above, for Nancy, a resonant self (self, from now on) is made of "the singular occurrences of a state, a tension, or, precisely, a 'sense"' (Nancy, 2007, p. 8). Since these occurrences are in a permanent state of occurring, that is, never fixated in a whole (in 'tension'), we can understand singular beings as being 'interrupted' or 'suspended.' These descriptions come from an earlier text of Nancy (1991), in which he extends this definition of the self to the definition of a community: "community is made of the interruption of singularities, or of the suspension that singular beings are" (31 ibid., All subsequent quotes from this passage.). This is why we can understand 'community' as ontological (grounded on a nature of being), and

not as teleological (grounded on a purpose or an objective). In other words, if community is thought of as a product of the work of 'selves' (teleologically), it follows that selves could also become the product of community.

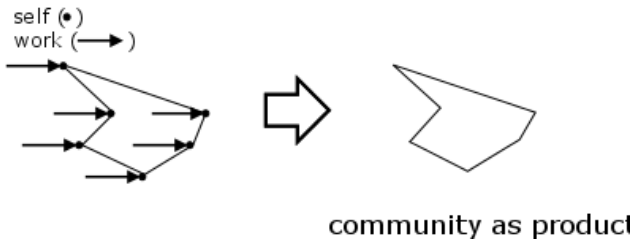


Figure 4.1: Community as work
A graph displaying the teleology of the work (arrows) of selves (dots) in community (dots joined by lines)

Nancy thus underscores that "community is not the work of singular beings, nor can it claim them as its works" (See Figure 4.1). Since community is ontological, Nancy understands it as the being of singular beings "suspended upon its limit." Therefore, within this ontology of community, how is it possible for us to speak of the 'work' of 'selves' and of community if 'work' is something that does not enter into its definition? Furthermore, how is the concept of the work of art redefined or framed within this ontology? Nancy's conclusion is that community can never result out of 'work,' but it is something that unfolds as 'unworking.' Borrowing from Maurice Blanchot's concept of *desoevrement*, Nancy proposes 'unworking' or 'inoperativity' as a way to understand 'work' within an ontology of community. 'Unwork' is work withdrawing from itself: "that which before or beyond the work, withdraws from the work." That is to say, Nancy points to a moment in which operativity separates from itself, and by that gesture, it distinguishes itself from both production and from a whole (finished product): "no longer having to do either with production or with completion, encounters interruption, fragmentation, suspension" (See Figure 4.2). Is this not the resonance of a return; work returning as the interrupted resonance of its own unworking?

At the Limit Nancy's concept of community can be recognized within his later and broader concept of 'resonance.' Given the fact that Nancy's ontology of sound points to the distance or the interval between sense and signification, and thus, to the emergence of a resonant subject during the sonorous presence, this distance can be thought of as suspended at a limit. We can think of this limit as an edge in the resonant network that, in Nancy's terms, exposes selves to themselves and to one another.

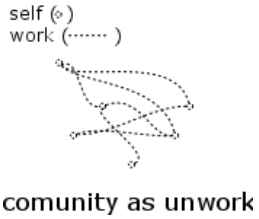


Figure 4.2: Community as unwork
A graph displaying community as an ontology of unworking.

Further, Nancy provides us with an essential insight, suggesting that "it is not obvious that the community of singularities is limited to 'man' and excludes, for example, the 'animal"' (ibid., p. 28). Therefore, by understanding resonant networks in terms of community we can speak of an exposure of selves, or a self-exposure, between humans and nonhumans in a liminality can be thought of as a skin: "a singular being *appears*... as finitud itself: at the end (or at the beginning), with the contact of the skin (or the heart) of another singular being..." (ibid., p. 28). This skin (or heart) makes at least two references, one to the affective presence of the body in relation to touch, and another to the finitud that singular beings are, that is, what (for Nancy) is the being in common of beings: their life, but fundamentally, their death. First, with this latter reference, we can depart a bit from Nancy's 'skin,' and stretch it away from this living/nonliving distinction altogether. Latour's "ontological hybrid" (see below) can thus enter into the considerations I am proposing here: "an actor-network is an entity that does the tracing and the inscribing. It is an ontological definition and not a piece of inert matter in the hands of others, especially of human planners or designers. It is in order to point out this essential feature that the word 'actor' was

added to it" (Latour, 1990, p. 7). Therefore, what is in common along the network is not finitud in terms of death, but in terms of the very condition of liminality: actor-network at the limit.

Infralanguage Latour considers that the descriptive project in Actor-Network Theory (AT) does not compose a metalanguage with which to define overarching explanations. Instead, AT searches for explications by retaining "only a very few terms —its *infra*language— which are just enough to sail in between frames of reference" [emphasis added] (ibid., p. 16). On the one hand, with this infralanguage AT arrives at an "empty frame for describing how any entity builds its world" (ibid., p. 16). That is to say, it is a descriptive project focused on "meaning-production" by means of creating associations and connections. On the other hand, with this infralanguage AT "grants back to the actors themselves the ability to build precise accounts of one another by the very way they behave" (ibid., p. 16). That is to say, precisely by disarticulating the search for 'overarching' explanations by means of a *meta*language (*the* key to open *the* door), this infralanguage is a keychain (of tiny keys) that opens many doors for the new "ontological hybrid" that is the actor-network, because it is what allows actors themselves to build their own account, that is, their own world: "world making entities" (ibid., p. 16).

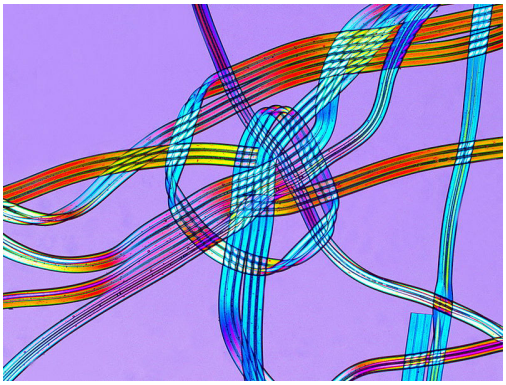


Figure 4.3: Spandex fibers under an optical microscope
"We required the material for the head to be elastic, to have a contrasting dark color, and to resist deformation and breaking. We are currently using spandex" (Oliver & Jenkins, 2008, p. 3). Image taken from the "Spandex" entry of Wikipedia: "Polyurethane Fibers Optical microscopy Polarized Light Magnification 100x (cross-polarized light illumination, magnification 100x). Created: 1 January 2015."

Infraskin Unlike in Latour's network, affectivity appears in Nancy's 'sense,' that is, on the body, its touch, and as before, in listening. AT quite literally forbids this, treating any "homogenous morphism" as "exceptions which should be accounted for" (ibid., p. 16). That is to say, if we may think of a skin upon which we (humans) resonate in self-exposure (community as unwork), then this skin must be an exception, because it is only human, or animal, or reduced to living entities with affective bodies. However, perhaps we can "account for" this skin as exceptional, and consider it "x-morphic," that is, "anthropo-morphic, but also zoo-morphic, phusi-morphic, logo-morphic, techno-morphic, ideo-morphic" (ibid., p. 16). This x-morphic skin is what we (humans and non-humans) have in common as singular beings in the network: what for Nancy is "at the confines of the *same* singularity" (Nancy, 1991, p. 28) would be at the irreducible limit of an 'infra' skin. Nancy is aware the this skin is not a total, complete, or superior skin:[2] like selves and community, it is interrupted. Interruption, however, in Latour's terminology relates to an ethics of the network where 'good' and 'bad' are understood in terms of navigation, mediation, and reduction. More

[2]For Nancy "there is no communion of singularities in a totality superior to them and immanent to their common being" (Nancy, 1991, p. 28).

connectivity is good, less is bad: "either an account leads you to all the other accounts…or it interrupts constantly the movement, letting frames of reference distant and foreign…" (Latour, 1990, pp. 13–14). Nevertheless, we can understand interruptedness as a general condition of irreducibility, which is, contradictorily enough "the highest ethical standard for AT" (ibid., p. 14). In being interrupted, the relation itself becomes ontological, since it is thus the unworking of the work of actors. The new "ontological hybrid" actor-network can be understood, then, as composed through the *unwork* of actors. This is the suspension at the limit: within the irreducibility of this interruptedness, this infraskin is simultaneously "always *other*, always shared, always exposed" (Nancy, 1991, p. 28). Like the skin of the Silent Drum Controller (See 3.4.2), suspended along the rims, stretching and compressing to touch, and making mechanical waves through the resonances of its silence, this infraskin would resonate at the limit. The Silent Drum tracks shapes made with the elastic fabric folding and unfolding against the human skin, therefore it "acts as the limits, but also as an extension of the human body" (Oliver & Jenkins, 2008, p. 2). I would interject here, and propose that instead of an "extension of the human body," the Silent Drum can be thought of as extending Latour's network *with* the body, that is, it grants affectivity to the network. In this sense, this infraskin is not a layer that separates the interior form the exterior of a body, or a surface under which or over which two selves can connect or extend. This infraskin is not just the promise of a new cultural algorithm (in Manovich's sense). This skin is not a surface, but it can be thought of as a texture; not a layer, but an interweaving of elastic fibers (See Figure 4.3) that, in their own locality are fragile, but due to their reticulated structure expand into a redundancy of fragilities. And we can listen to it.

Database Community With resonant networks as an instantiation of a process of unworking, we can think of database communities. That is to say, the database and the databaser engage in a form of touch. This touch is easy to feel in the case of the Silent Drum because of the elasticity of the drumhead resisting the pressure of the hand. The material resistance of the skin

represents, thus, the evidence of the strength of connectivity itself: "strength does not come from concentration, purity and unity, but from dissemination, heterogeneity and the careful plaiting of weak ties" (Latour, 1990, p. 3). In this sense, if one removes the skin one would be left with no resistance. Hence, the Silent Drum would be rendered weak. But, this infraskin or its resistance does not need to be visible for the community of the human and the non to exist. This is the case of the MANO controller (Oliver, 2010), where the elastic tissue that was tracked in the Silent Drum is removed, and what is tracked is now the skin of the hand (*mano* in spanish) directly. This physical elision points equally to the (human) invisibility of the infraskin, as to the sonorous presence of its resonance. In making this controller, it can be argued that Oliver La Rosa was precisely making audible not only the gap between the sensor and the dark background interrupted by manipulation, and not only the different gestures recognized and tracked by the algorithm, but also the gap (the limit) between the human and the nonhuman, what I am calling here the infraskin. In making this skin resonate with a database, the result is a sonic event that instantiates the community of this limit. That is to say, all the elements that compose the multiplicity of the construction (sensor, video stream, C++, C, Pure Data, the parameters obtained from video analysis, etc.) enter into resonance with each other, and thus manifests the sound of a community. The "trace of the edge of the hand" (ibid., p. 2) becomes the trace of the resonance of a database community. Or simply, database music.

In a database community, database music can be understood as a hybridly social and communicative event. As an unworking of databasers and databases, it can be considered as the infraskin upon which we (human and nonhuman) resonate, as well as the trace and the tracing of its resonance. The qualities of the music work that result from it can also relate to incompleteness, suspension, as well as to fracture, instability, and interruption. In this sense, databaser and a database can be heard in resonance with each other, as well as in communication with each other, but it is a resonance or a communication that is not teleological. Communication, in this context, refers not to language, or to information, but to a property of being in common. In Nancy's

sense, the being in common of singularities; in Latour's sense, the connectivity and associations of actors. For Nancy, "communication is the unworking of work that is social, economic, technical, and institutional (Nancy, 1991, p. 31). Communication, in this sense, is also the unworking of work that is resonant, and thus it can be considered as a music unwork. I will refer to this in later sections (See 7.6). In what follows, I will focus on the 'trace' of this resonance.

Chapter 5

Databases And Memory

5.1 The Effraction Of The Trace

According to Jacques Derrida (1978), Freud understood memory as the essence of the psyche: "Memory... is not a psychical property among others; it is the very essence of the psyche: resistance, and precisely, thereby, an opening to the effraction of the trace" (ibid., p. 201). 'Effraction' is a legal term that refers to making a forcible entry into a house. Derrida uses it in relation to the process of impression (pressing *in*) with which memories are unconsciously inscribed. The inscription of a trace relates thusly to a certain violence. This does not mean that memories are violent in themselves, but that in order for something to leave a mark (a trace), there has to be a force acting against a certain resistance. If traces can be thought of as 'paths' or tracks along which memories are inscribed, then making these paths is a form of 'breaching' or path breaking. Derrida points to a crucial issue with Freud's idea of opposing forces: if resisting forces met equally strong resistive forces, the result would be a 'paralysis' of memory. Upon this possibility, he recognizes that "trace as memory is not a pure breaching... it is rather the ungraspable and invisible difference between breaches [traces]" (ibid., p. 201), a difference that exists both in space and in time. This spatio-temporal play of difference is what Derrida calls *différance*, and how he understands the

"work of memory" in itself (ibid., p. 226). *Différance* has many names throughout Derrida's text, making any attempt to address it an extensive process.[1] In what follows, I will briefly delineate what I consider its most important qualities for the purpose of discussing the database in relation to memory.

Différance The play of difference with which Derrida describes the psyche can be understood in terms of how it applies to language. For example, in order to describe an object ('database'), one needs another word (e.g. 'data,' 'algorithm,' 'structure'), which in turn demands yet another word (e.g. 'bit', 'computer,' 'model'). Up to this point, we still don't know what the word means in the first place, or if one means *this* or *that* database, or if one will ever describe the object in question. To explain fully the point, for instance, the reference would need to *become* the object. In fact, the reference will *never* become the object, hence the moment of deferral that plays *in time*. Likewise, the word 'database' is not the same as, for example, 'archive,' 'library,' or 'dictionary,' pointing at the differing references that play *spatially*, across the multiple meanings to which each word points. In any case, the play of difference engages us with an adventure that makes us think about 'database' without necessarily aiming at one answer, but might help us 'traverse' its trace. Within this traversal we can find the two meanings of the the French word for difference: to defer and to differ. Derrida makes this distinction 'evident' with the use of the 'a' on the spelling of the word 'différance,' which in French makes no audible change. In doing this, he points to the distinction between spoken and written words, while emphasizing the dual meaning of the play in question: deferring-differing. In relation to the psyche, the play is between the traces. On one hand, the 'spacing' of memory traces relates to the difference among traces, how they relate to each other in terms of their identity, and also to the difference between consciousness and unconsciousness. On the other, difference relates to the deferment, delays, or the postponement of the inscription of traces. In this sense, he understands Freud's concept of 'death drive,' that is, the gravitational pull

[1] For further reference on *différance*, see (Gratton & Morin, 2015, pp. 71–72; Derrida, 1978, p. 219; Derrida, 1982)

that the inorganic exerts on the organic: "death at the origin of life which can defend itself against death only through an *economy* of death, through deferment…" (ibid., p. 202).

Funes I would like to return to one more aspect of Funes' story that relates to *différance*. Consider how the narrator writes about his own unfinished project of learning Latin: "The truth is that we all live by *leaving behind*; no doubt we all profoundly know that we are immortal and that sooner or later every man will do all things and know everything" [emphasis added] (Borges, 1994, p. 113). A more literal translation of the first sentence would read: *what is certain is that we live deferring all that can be deferred* [Lo cierto es que vivimos postergando todo lo postergable (Borges, 1942)] The narrator touches upon two crucial points with which we can understand *différance*. On one hand, the postponement that is assigned to 'every man' but that is deprived from Funes who cannot afford to defer given his condition of total temporality. On the other hand, the absolute knowledge that the narrator assigns to the multiplicity of man and the multiplicity of things, that is nonetheless made present in a certain sameness that Funes represent (a cancelling out of that which differs). Thus, extending Oviedo's consideration of Funes as the 'antithesis of the writer,' Funes can be further thought of as comprising an antithesis of *différance*, a man whose psyche has no resisting unconscious. For Spivak, what Derrida's reading of Freud allows him is to understand how forgetfulness is "active in the shaping of our 'selves' in spite of 'ourselves"' (Derrida & Spivak, 1976, p. xlv). That is to say, because of the unknowable forces of the unconscious, we have no control on neither tracing nor the undoing of traces: "we are surrendered to its inscription" (ibid., p. xlv). Furthermore, in contrast to a Nietzschean view of an active search for forgetfulness "or the love of chance," what Derrida finds is that "we are the play of chance and necessity [and] there is no harm in the will to knowledge (ibid., p. xliv). Unlike Funes, we are already immersed in (and shaped by) a world of *différance*. Therefore, we can now ask ourselves in what way does *différance* change our understanding of the database? Databasing can perhaps be understood as one way of traversing *différance*, not as an aim, but simply as something you are: in

databasing, you *are* databasing. Funes, in this sense, it not databasing, he simply is the database. To be databasing one must be in a loop, in resonance, differed-deferred in time and space *with* a database.

Writing and Databasing As Derrida points out, western philosophy has construed writing as a process of *hypomnesis* or an externalization of memory (Derrida, 1978, p. 221). Writing media can be considered hypomnesic or an externalization of our memory that on one hand materializes our memories and on the other constitutes a operable tool. Furthermore, Freud's metaphor of writing to define the apparatus of the psyche as tracing and erasure indicated that "'Writing... is the name of the structure always already inhabited by the trace," meaning that it is a broader concept that goes beyond writing itself, that is, beyond a "system of notations on a material substance" (Derrida & Spivak, 1976, p. xxxix). How does 'writing' relate to the resonant network I described earlier? We can approach this question in two ways. On the one hand, we can find in Latour's 'tracing' of the network a relation to the 'trace' as described by Derrida, insofar as they both relate to meaning. Latour's project, however, leaves behind the (human) body so as to only maintain this 'tracing' of the network, thus making it exceptional to think of memory in terms of networks. Nevertheless, there is an activity of tracing that takes place in the resisting forces of the unconscious. On the other hand, with his conceptualization of memory as writing, Derrida reconfigures the notion of authorship within writing. In this sense, when memory is thought of as writing, the classical notion of 'self' begins to disappear, opening up the space for the nonhuman. In what sense can this disappearance of the self be accounted for in databasing? Along the trace of the resonant network (what I called infraskin) every resonant node of its trace can be considered an agent in the constitution of selves. The database becomes an agent of selfhood as well as an agent of authorship relating to the resulting work of database music. Further, as an instance of hypomnesis, the database appropriates the qualities relating to memory and *différance*. Thus, not only can the nonhuman be reconceptualized within these qualities, also the human becomes reconfigured when faced upon

this common linkage. This is how a further step into the conceptualization of memory can be of aid, one that extends 'tracing' or 'writing' into what I am calling 'databasing.' 'Writing' can thus function as a link between human and nonhuman memory. For example, from the beginning of the process of impression, traces are "constituted by the double force of repetition and erasure" (Derrida, 1978, p. 226). Memories are inscribed with the very structure that enables their own effacing, they "produce the space of their inscription only by acceding to the period of their erasure" (ibid., p. 226). Programming languages imply writing in terms of symbols (words and characters) that call certain functions. However, the structures that can be instantiated with code, such as data structures, appropriate the concepts of writing and erasing as well. This is known as memory management. Erasure is embedded within the structure of coding. For instance, C++ classes include within their own data structure a call to their `destructor`. This means that, whether explicitly or implicitly, all classes —i.e., all data structures which correspond to instantiated objects— have a way to self-erase, or self-destruct after the object is no longer needed. This self-destruction means, precisely, releasing the object's resources, that is, to free the physical memory space that it has occupied throughout its 'lifetime.' This comparison between `destructors` and erasure serves as a starting point to determine the extent to which computer memory (data structure handling in restricted, discrete space) can be thought of as human memory as such, and, if so, the extent to which it also constitutes an instance of an unconscious structure. We can imagine a (useless?) program written in C++ with self-destructing classes that instantiate and destroy themselves at their own pace. Considering the `destructor`, it is simply an automation of the otherwise manual memory allocation or deallocation. In a practical sense, allocating memory is like calculating how much and what kind of paper you will need to write your next short story, ordering it online, and then throwing it in the recycling bin, I presume, only after you have written and digitized your story. In any case, we can think of this 'paper' as the space that is needed for writing, as well as the resisting force against the pen, but also as a base upon which data is stored in relation to its size. Memory management is a feature with which databasing relates at the level of writing code.

5.2 The Archontic Principle

Derrida's broader notion of 'writing' relates to what he calls an 'originary' trace. As I described earlier, the inscription of the trace contains from the start its own erasure. This means that, in a reciprocal motion, the origin of the trace is at once originary and non-originary: the origin of a disappearance and the disappearance of the origin. This apparent contradiction can be approached by the actual process of writing something on a page: where or when does the writing begin, when the pen touches the paper or when the paper lends itself for writing? The same question applies to the crossfade between the 'I's in the case of Alvin Lucier's work (See 4.3), where or when does the 'I' begin 'sitting' in the loop? Further, no matter how many metaphors we may give to this paradox, it simply falls out of empirical quests. Therefore, this paradox of the trace is understood by the concept of *différance* that we outlined above. The repercussions of this word that "is not a word and it not a concept..." have now a long history that does not interest us fully, therefore we can will simply jump thirty years ahead in time, after computers entered the scene, and after the Internet became popularized by the World Wide Web.

Derrida and Prenowitz (1995) exemplified the intricacies that come out of the process of archivization, delineating an economy of archives into what he calls the *archontic* principle.[2] A brief etymological impasse, coming from Derrida's text, might be pertinent here. 'Economy' is related to the greek word *oikos* (house), and *nomos* (man-made law). In ancient Greece, the official documents of the law were kept under guard in the house of the magistrates (*arkheion*), in a form of house arrest. Funes, as I have described earlier (See 1.7), was in a similar form of house arrest (arrested from the outside world). The archive condenses this economy, however, within the confines of a public place: by an institutional passage from the private (house) to the public (archive, museum). The word 'archive' (*arkheion*) comes from the greek *archē* which relates, on the one hand, to the originary, as well as to the ruling. Thus, the archontic principle is a type of

[2]His example was the Sigmund Freud Museum, which is located in the same house where Freud lived: https://www.freud-museum.at/en/

authority that the archive exerts, which can be understood as the law of the house, or the law that is before anything else. Hence, its categorization as principle, which is also related to the origin (e.g., the latin root *principium* which refers to the beginning) and to the figure of the ruler (principal or prince). Derrida argues that the archontic is embedded first with a sense of filiation, that is, with fatherhood and the relationship between father and child. Thus, the archontic is "paternal and patriarchic" (ibid., p. 60), and it is grounded on a domicile (house) or an institution (family). Within this domiciliation or institutionalization is from where rules are prescribed, and where the ruling takes place. The archontic has thus the form of Freud's Oedipus complex. An Oedipus complex constitutes a desire of the child's unconscious hatred towards a parent. It is itself based in Sophocles' drama *Oedipus Rex*, in which such desire results in eventual parricide: "this archontic, that is, paternal and patriarchic, principle only posited itself to repeat itself and returned to re-posit itself only in the parricide" (ibid., p. 60).

As hypomnesis, archives represent for Derrida another instance of the movement of technology. This movement consists in "a transformation of the techniques of archivization, of printing, of inscription. . . " (ibid., p. 16). These 'archival machines' which developed in the thirty years since he first considered Freud's *Note on the Mystic Pad* had reached Derrida's email inbox: his comments on how email technology would have (and will) affect psychoanalysis attest to this fact. Further, they pose a question that he leaves nonetheless unanswered, which is why I consider his discussion of the archive relevant to our discussion of database aesthetics. Derrida finds in Freud's metaphor for the psychic apparatus a point of departure to the question of how psychoanalysis changes by the presence of a technological device. In this case, the device was a toy called the Mystic Pad, a children's writing board made of wax and a thin layer on top: upon impression it leaves a trace; when one lifts the layer, the trace erases. The structure of the psyche for Freud could thus be understood by using both hands simultaneously, one pressing (writing), while the other was lifting (erasing). Writing in 1995, Derrida posed the following question:

> Is the psychic apparatus better represented or is it affected differently by all the technical mechanisms for archivization and for reproduction, for prostheses of so-called live memory, for simulacrums of living things which already are, and will increasingly be, more refined, complicated, powerful than the 'mystic pad' (microcomputing, electronization, computerization, etc.)? (ibid., p. 16)

The structure of this question points to two possible answers. Either technology is a representation of the psyche, or the understanding of the psyche is changed by technology. On one hand, the question refers to how distant we are in the movement with which we can arrive at the object (psychic apparatus). This is a temporal movement which relates to deferral: how much longer until we get our object? The answer, as we have seen, is simply never. In order for the represented to be fully represented it must become present. On the other hand, the 'technical device' has the potential to affect the object: it differs from and might reformulate thus the psychic apparatus. To a certain extent, this suggests that these 'prosthesis' of 'live' memory, or 'simulacrums of living things' represent some form of (disembodied) nonhuman (life) against which the question is asked. We can only guess. In any case, Derrida claims it is a question of progress or evolution, in which "neither of these hypothesis can be reduced to the other" (ibid., p. 16). It appears, therefore, that this quest reaches a dead end. Derrida does not continue on this quest, and instead takes the text in directions that are not pertinent for our purposes here. Therefore we can press on this question for a while.

The structure of the memory ("the essence of the psyche") in opposition to the archive can be understood with the opposition of *anamnesis* (the act of recalling, remembering) and *hypomnesis* (the technical storage device); the former internal, the latter external. We can advance now that, like the archive, the database is hypomnesis and external (to the psyche), just as well as it exerts the principles of the archontic. As I have described earlier (See I), a database comprises the partition within computer memory where data is stored. However, in order to store data, one has to assign data types and structures, thus providing with the necessary structural frame that indicates how to access the data. One of the key concepts of the 'archontic principle' is 'consignation,'

which has at least three meanings: assigning residence, entrusting something in reserve or deposit, and what Derrida calls "gathering together signs" (ibid., p. 10). All of these can be represented with memory management, that is, by 'declaring' and 'initializing' variables, as well as by making 'unions' or more complex data structures. Consignation in archives has, for Derrida, a presupposing aim: to "coordinate a single corpus" (ibid., p. 10). This is exactly the case of a database, which extends this coordination with the possibility of networks. The various database models that I have shown in previous sections point to the different ways with which to coordinate the economy of database systems. In order to flip Derrida's question upside down, we can ask ourselves what to do with all these technical mechanisms that microcomputing enables? How do we make something out of them? And further, what is in them that we can call aesthetic? Instead of thinking how memory is represented in the database, we can ask how we can represent the database within memory. Further, instead of asking how memory is affected by the database, we can ask ourselves how can memory affect databases. The theoretical and aesthetical aspects of this reversal can be seen as follows. On one hand, this reversal points to a reconceptualization of databases within the "evolution of archival techno-science" (ibid., p. 16). On the other, this reversal calls for a restructuring or a technical reconstruction of the database in relation to memory.

In his architecture, Von Neumann proposed that the storage unit of the computer would allow for data to be written and erased in different locations and times. He was following Turing's conceptualization of the *a-machine* —i.e., the *Turing machine*—, which was a mathematical model for computation, that can be represented by a symbol scanner and an infinite tape, where the scanner gets, sets, or unsets a symbol on the tape, and the tape moves to the next slot accordingly. These setting and unsetting movements represent inscription and erasure, to the point that, as Kittler notes: "the two most important directing signals which link the central processing unit of the computer to external memory are being called `READ` and `WRITE`" (Ernst, 2013, p. 131). An important distinction needs to be made here. While I am arguing for the similarities that exist between human memory and databases, Ernst instead proposes that databases or "digital an-archives"

replace human memory. As media tend to converge toward digital media, 'reading' gives way to mathematical processes that interpret data: "signal processing replaces *pure* reading" [emphasis added] (ibid., p. 130). This statement resonates with the Kittlerian, disembodied worldview (See 1.3), could be extended to saying that signal processing replaces *listening*.

Convergence, understood in this way, points to a world view where data structures and algorithms replace our own memory: the ultimate convergence is the absorption of human memory (the psychical apparatus) in the database. In this sense, the database would remove our embodiment from our bodies completely. There is a fine line between this convergence, and what Derrida mentions about consignation in relation to archives. In both terms, there is an aim that is presupposed: everything goes in the same place, archive or database, and from this technological place we find a hint of our own destruction as humans. That is to say, the Kittlerian residue of humanity understood as an expression of the pull towards the inorganic, the death drive, the archive 'fever.' But, if the database reads itself, it also writes itself, and, in the case of music, if it listens to itself, it also sounds itself, and we, as humans, simply are part of this loop and of its resonance. This moment of resonance enables both the nonhuman and the human to coexist. Of course, the paradox of a destructive force that is only possible through its destruction might be no welcomed in our houses, in our databases, and in ourselves. What this resistance indicates is the presence of an absence, a sound without a body, and the spectrality of an uninvited guest: and it is the sound of this force that makes an impression. Derrida calls these "lovely impressions" or "memories of death" (Derrida & Prenowitz, 1995, p. 14), relating the death drive to the anarchic and an-archontic force that leaves behind no trace because it "always operates in silence" (ibid., p. 14). Under these terms, could we not reconfigure this fever in relation to databases as a force of inoperativity, one that unworks in resonance?

5.3 The Spectral Database

Anarchic Records In his conceptualization of the *anarchoarchive*, Wolfgang Ernst (2013) opposes technical recording with symbolic transcription. A microphone captures the entire sonic environment forming an involuntary memory that becomes a form of anarchic archive, or *anarchive*. For Ernst, an 'anarchive' presents no intrinsic symbolic ordering, in contrast to the highly structured ordering inherent to archivization. From this distinction he draws two comparisons. First, he compares analog and digital recording technologies: the former are anarchic because they "operate…in the material sphere of magnet spots and electromagnetic induction;" the latter are "microarchives" due to their "clear address structure" Ernst (ibid., p. 92). Second, he finds in musical transcription another expression of archival order. For example, Bela Bartok's transcriptions to musical notation of Milman Parry's Serbian epic song recordings represent an archivization process by which symbolic transcription leads to the ordered archive that constitutes a score: "an anarchive of sound in technological storage as opposed to the archival order of musical notation" (ibid., p. 174). What these oppositions point to is the middle term that constitutes the object of the "media archaeological" project, whose central focus is an awareness of the mediating device: "at each technologically given moment we are dealing with media not humans" (ibid., p. 183). Ernst bases much of his considerations of the archive on the work of media theorist Sven Spieker, who analyzes Derrida's conceptualization of the archive and its destruction drive. For Spieker, the central feature of archives is not its hypomnesic quality, but a certain need to "discard, erase, eliminate" everything not intended for archivization (ibid., p. 113). In other words, archives filter and frame, and thus any archivization project is tailored to the technical intricacies of the media involved. However, Ernst radically reminds media archaeologists: "we are not speaking with the dead but dealing with dead media that operate" (ibid., p. 183).

At the core of archivization projects, as I have mentioned earlier, is an institutional passage from the private to the public. The force that constitutes this passage is the destructive drive,

the archive fever, which exists as the very threshold of the inside and the outside of the archive. The passage (and not the force) is thus marked on the mediating technology. The reverse happens in the psychic apparatus. The public becomes private by the effraction of the trace, which is also an expression of the same destructive force that is the death drive. The traces are thus the marks of the psychic activity. To a certain extent, we can understand this drive with transducers.

Nonhuman Eardrums Audio recording can be seen as either memory or archive depending on their role in mediating private and public spheres. The world can be considered public because it is in a constant state of availability at any time, but a microphone can be considered an actor of privacy, since it prevents some sounds from being recorded while allowing others to pass through. In this sense, a transducer's filtering capacity enacts the passage from the public to the private and in the case of speakers from the private to the public. For Jonathan Sterne (2003): "every apparatus of sound reproduction has a tympanic function at precisely the point where it turns sound into something else... and when it turns something else into sound" (ibid., p. 34). Therefore, considering these nonhuman eardrums (tympanic membranes) as actors of privacy and publicness, audio reproduction technology can be compared to the structure of human memory and archives. A microphone becomes the threshold from the public to the private, from the outside to an inside that as far as we can tell for our knowledge of our own memories, is made of psychic activity, that is, memory. A loudspeaker, as a reversed eardrum would publish all there is to sound of an archive, the privately stored waves to the publicly reproduced air pressure waves in space. Perhaps this function of the transducer is in itself so representative of the very own death drive, that when in front of them, or around them, we feel a certain call for performance, and a certain disposition towards listening, which might have something to do with something other than just media. That is to say, even though they are "media and not human" transducers engage us humans with a ghost, neither dead or alive, neither material or immaterial: in lack of a better word, what is known as *spectral*.

Spectrality of Archives Transducers as nonhuman eardrums constitute the limit between the sonic world and the binary world of databases.[3] Comparing these transducers to memory and archives suggests a hybrid object: one that, on the one side, becomes a private and singular 'trace,' and on the other, becomes a public and resonant 'space.' Thus, databases represent neither a trace nor a space. According to Derrida, "the structure of the archive is *spectral*. It is spectral *a priori*: neither present nor absent 'in the flesh,' neither visible nor invisible, a trace always referring to another whose eyes can never be met…" (Derrida & Prenowitz, 1995, p. 54). The hybridity that the database projects when compared to memory or archives points to a certain uncanniness, that is, precisely to the hauntedness that comes from its spectrality.[4] Derrida claims that, addressing a phantom is a "transaction of signs and values, but also of some familial domesticity" (ibid., p. 55), meaning, on the one hand, that in the uncanny encounter with a ghost, there is familiarity, that is, there emerge feelings of what is known to be close to us, but also that which composes the authority of that closeness. Therefore, embedded in this familiarity is the archontic, the Oedipal, etc., and thus the expression of power that this apparition brings forth. On the other hand, the familiar is also related to an economy, that is, to the passing through (trans-action) of signs, but also of translation —or better, the 'transduction' of things. This is why Derrida considers any encounter with the spectral to be an instance of 'addressing,' which speaks of a certain uncanniness coming from the hauntedness of that which is at once familiar and unfamiliar: "haunting implies places, a habitation, and always a haunted house" (ibid., p. 55). Like Funes' acousmatic voice, from the shadows as a shadow, the transduced publishes and the database haunts.[5] Furthermore, it can be argued that considering the database in such way is plain and simple delusion, that is, insanity at its best. Not surprisingly, this is the point exactly; within this delusion exists a certain truth:

> …it resists and *returns*, as such, as the spectral truth of delusion or of hauntedness. It *returns*, it belongs, it comes down to spectral truth. Delusion or insanity, hauntedness

[3] Transducers can also be understood as the limit be between databases and the tactile world (van Eck, 2013, p. 223)
[4] Not to be confused with the Fourier-based French 'spectralists' of the 1970s.
[5] For a detailed revision of acousmatic history, see Kane (2014)

> is not only haunted by this or that ghost. . . but by the specter of the truth which has been thus repressed. The truth is spectral, and this is its part of truth which is irreducible by explanation. . . (ibid., pp. 54–56)

Agency of the Uncanny Building on Derrida's views on the structure of the archive as spectral, I suggest we can consider databases spectral too but in a different sense. The spectrality of the database comes when we (databasers) become part of the loop by databasing. The loop, that is, the circuit, the access, the programs, the transducers, the feedback, etc., in performance, in listening, in composing, etc. We engage with spectrality at this loop, that I also have called skin (or infraskin), and I also referred to as resonant network. As humans, we have no direct access to data space with our bodies and thus we cannot engage in transaction with the specter directly: we need transducers, we need media. Once we begin residing in the loop, we begin resonating with the database, but we also begin resonating with its spectrality. However, this does not mean that we have become an 'actual' ghost, or a spirit, or that we have suddenly become enlightened with a transcendental *deus ex machina*. Upon resonating in this spectral loop we can recognize a certain aesthetic agency of the database, one we can encounter whenever there is a database in art: "Recognition of the uncanny nonhuman must by definition first consist of a terrifying glimpse of ghosts, a glimpse that makes one's physicality resonate (suggesting the Latin *horreo*, I bristle). . . (Morton, 2013, p. 169) The illusory violin in the acousmatic concert I mentioned earlier (See 4.2), exemplifies Hansen's concept of the creation of auditory images our brain's capacity for virtuality has: our imagination. The sound of a violin can be recorded or synthesized into the privacy of a database. Then, it can be played back with loudspeakers located in such a way that they emulate the location of an actual violin player. As a result, the listener could very likely imagine a physically present violin in the room, a sound without a body, that is, a ghost. This ghost comes in as the phantom of a human player; of the violin; of the histories and traditions that those two elements bring forth; of the presence of the nonhuman that the database implies; of the privacy that is not human but is still uncannily private; of the hauntedness of the archontic that the above sets forth; and so on. In this

way, the spectrality of the database attests to its relation to memory and archives, and, thus, to its aesthetic resonance within our experience.

This hauntedness can be indeed embodied, not only in the form of authority, as I have shown in the case of its archontic presence, but also in the form of a style, and as we will see, within the constitution of gender.

Chapter 6

Performativity Of Databases

6.1 Gendered Database

> Gender is not passively scripted on the body, and neither is it determined by nature, language, the symbolic, or the overwhelming history of patriarchy. Gender is what is put on, invariably, under constraint, daily and incessantly, with anxiety and pleasure, but if this continuous act is *mistaken for a natural or linguistic given*, power is relinquished to expand the cultural field bodily through subversive performances of various kinds. (Butler, 1988, p. 531)

Philosopher Judith Butler (ibid.) distinguishes between an expressive and performative self. The former comes from an essentialist view from the self as being 'inside' and displaying itself on the outside. The latter is an illusory self, strictly outside and unrelated to the "natural or linguistic given." She understands gender within this performativity of the self. Like the self, gender emerges temporally, at the surface level of the skin of the body. This notion of gender relates with Jean-Luc Nancy's notion of resonance and the self (See 4.2).

Skin of the Database In the performativity of databasing resides the possibility for what I have called infraskin of the database to emerge. The prefix 'infra' that I have added to this skin, however, does not indicate interiority. It simply suggests a positioning that cancels any hierarchical order

in the resonance among the human and the nonhuman. It is 'infra' in relation to the dynamics of an interaction along the interweaving of its texture. On the one hand, this skin is this spectral texture of the database's illusory self. On the other, it is the limit upon which the human and the nonhuman engage in resonance. The skin of the database carries the mark of a style and the possibility of its gender. That is to say, in defining style as a repetition of acts, it is a form of embodiment that is ascribed to databases. This enactment can be understood as the enactment of a gender "which constructs the social fiction of its own psychological interiority" (ibid., p. 528), and thus the database can have as many genders as there are 'social fictions,' in the permanent play of difference suspended upon its limit. In this sense, we can understand "its own psychological interiority," as referring precisely to the gendered self of a database, one that is established in its own historical sense: the history of its resonance, authority, and transformations. The gendered database participates aesthetically, dramatically, and with its own authority in the history of its practice. The infraskin is gendered at any point, in any time, in any way, and the uncanniness of its appearance redefines our own social categories, and our own realities. Databasing, as the performative condition of databases, is gendered, and it exposes us to the gendered resonances of our acts.

Expressing Nothing Butler sets forth a critical genealogy of gender which relies on a "phenomenological set of presuppositions, most important among them the expanded conception of an 'act' which is both socially shared and historically constituted, and which is performative…" (ibid., p. 530). The difficulty Butler recognizes in this view of gender, is that "we need to think a world in which acts, gestures, the visual body, the clothed body, the various physical attributes usually associated with gender, *express nothing*" (ibid., p. 530). Therefore, how can the database itself be conceived in these terms of performativity, to the point that, while having the capacity to store millions of data, a database can indeed express nothing? Furthermore, how does a gendered understanding of databases reshapes our own practice as databasers?

A Historical Situation Butler defined gender identity as a historical situation, distinguishing between the physiological facticity of the body (sex) and the cultural significance of such facticity in terms of gender. Added to this distance between body and identity, Butler speaks of the body as a performative process of embodying cultural and historical possibilities. These possibilities, which are delimited by historical conventions, are thus materialized on the body: "one does one's body differently from one's contemporaries and from one's embodied predecessors and successors" (ibid., p. 521). Therefore, the body comes to be a "historical situation" that results from the performativity of embodiment. In other words, the actions related to what Butler calls the "structures of embodiment" constitute an ontological sphere of present participles, such as 'doing,' 'dramatizing,' and 'reproducing.' Furthermore, what this structure of embodiment entails is the constitution of not only gender, but also style. Since gender is constituted temporally, it is necessarily historical:

> to be a woman is to have *become* a woman, to compel the body to conform to an historical idea of 'woman,' to induce the body to become a cultural sign, to materialize oneself in obedience to an historically delimited possibility, and to do this as a sustained and repeated corporeal project. (ibid., p. 521)

Subversive Repetition Far from being a prescribed given, the constitution of gender on the body is itself a result of mediated history. In other words, gender is a creative act of interpretation and reinterpretation that reveals itself on the body, not as an expression that comes from within, but as the sedimented layers that deposit themselves in time. Furthermore, within this notion of temporality there is a need for repetition that is susceptible to breakage, or what Butler refers to as "subversive repetition" (ibid., p. 520). In being a temporal identity which reveals itself through a "stylized repetition of acts" gender constitutes an "illusion" of a gendered self. These acts take place *on* the body, by the mundane instantiation of bodily gestures, movements, and enactments. Furthermore, these acts are necessarily discontinuous, and it is because of this discontinuity that exists the possibility of gender transformation. In this sense, gender performance is neither linear

nor nonlinear. It resides along an anarchic temporality that replaces teleology with the multiplicity of resonant nows. It is an inline iterative function with random breaks.

Gendered Database The database is a collection of facts. This is what Butler's gendered self can teach us about databases: in performing the database, the database appears like gender, as a historical situation. Its body is felt neither as the database body —as if the materiality of the computer's architecture could come as a proxy for the nonhuman body— nor as the extension of the embodying databaser, that is, as a prosthesis that expands the databaser in an expressive way. The body of the database emerges as a phantom, as spectrality itself, and it is this nonhuman presence that engages with us in the publicness of performative acts. The specter of the database must not be understood spiritually, or as a *deus ex machina*, or as a soul, or singularity that begins to act *as* human and, by extension, supersedes the human. It is simply a nonhuman fabrication of selfhood: there, around, making its way through the rupture of the permanent condition of performativity to which we (humans and nonhumans) are phenomenologically bound. This is how the style of the database appears. This nonhuman self, like Butler's gendered self, is equally 'outside;' "constituted in social discourse" (ibid., p. 528). In other words, the skin of the database —what I called infraskin (See 4.4)— is open for perception outside of itself, and in fact, nothing about of the database can be considered expressive. Inside the database there is literally nothing but zeros and ones, nothing but data; in the same way, nothing is inside of the body but flesh, bones, and veins. When considered as internal, inherent, or essential, the classical notion of the self, in its heteronormativity, is seen as a "publically regulated and sanctioned form of essence fabrication" (ibid., p. 528). In this state of being fabricated, expressivity serves as the foundation for what Butler refers to as the 'punitive' aspects of wrong gender performance. In this sense, the social quality of acts that fall outside the regulated binary gender construction work their way into punishing the body, incarcerating it, severing it, as is, for example the famous case of Alan Turing:

> Turing's later embroilment with the police and court system over the question of his

> homosexuality played out, in a different key, the assumptions embodied in the Turing test. His conviction and the court-ordered hormone treatments for his homosexuality tragically demonstrated the importance of *doing* over *saying* in the coercive order of a homophobic society with the power to enforce its will upon the bodies of its citizens. (Hayles, 1999, p. xii)

6.2 Towards The Limits

Communities of Skin The limit of the database, as performative, spectral skin, allows for a community to emerge between the human and the nonhuman. This means that the agency locus of the database needs to be placed precisely on its skin. In other words, given that this skin is available to the perception of others, it becomes touchable, it reaches our own limit as databasers. By exposing our own limits to ourselves and to each other, the database changes our definition and delimits the extent of our own singularity. However, this does not mean that this skin stands in the way of our performativity, or worse, that it precludes or determines ourselves. If this were true, we would be once again subject to technical determinism, essence fabrication, etc., and falling out of considering any possibility of community between anything that is not human. As I have already commented, this skin is human and nonhuman. Thus, the fact that the skin of the database changes our own skin simply means that we are already in communication with it, that is, in community, and also in a state of resonance with it. This is the function of the skin of the database: like the skin of a drum, or the skin of a loudspeaker, the skin of the database resonates with our own skin, engaging the resonant body with a resonant spectrality. In sum, what this infraskin allows is for a community of resonance, which has no purpose, no intentionality, and no essence; only appearance and motility, performance and repetition.

Hybrid Pluralities Database models tend to reside next to each other, either within a single database system or within an interconnected networked system. Databasers have access to the many features that each model offers, focusing on those features that are suitable for their needs.

The skin of the database is as fluid as the constitution of gender, and if this is true, then the fluidity of databasing itself comes to represent the constitution of gender through the performativity of databasers. By resonating in such performativity, databasers approach the limit of the database. This approach to the skin of the database mutually exposes database and databaser. What this exposure amounts to is not, however, an opposition of forces. It results in the fragmented state of community that resides in the different spaces opened by this exposure. In other words, this exposure is of a hybrid plurality that resonates at the limit. Engaging with the touch of the spectral database means reconfiguring, resounding, and remembering our own sense of touch, just as well as our own sense of self.

6.3 Contingencies Of Style

> . . . style, supplementing timbre, tends to repeat the event of pure presence, the singularity of the source present in what it produces, supposing again that the unity of a timbre —immediately it is identifiable— ever has the purity of an event. . . The timbre of my voice, the style of my writing are that which for (a) me never will have been present. I neither hear nor recognize the timbre of my voice. If my style marks itself, it is only on a surface which remains invisible and illegible for me. (Derrida, 1982, p. 296)

A database without performance represents a disembodied ‘base’, that is, the spatially ordered set of computer hardware together with the software routines that it embeds. It is its most basic level, a foundation upon which the database tree can be performed. This ‘base’ in database comes as a stage for databasing itself: a stage without performance is an empty stage, extension of space. Databasing projects its own style as a result of its performance, and through this projection comes the exposure of its skin. The “stylized repetition of acts” (Butler, 1988, p. 519) in the dramatic case of the gendered database is now revealed as style. Like skin and voice, singularity emerges as style and timbre.

Style and Timbre 'Style' comes from the latin *stilus*, meaning a sharp object with which you can write: like the stylus of a record player, it is a writing tool. Its meaning extends through writing to the manner in which writing is carried out: the variations and oscillations of the pen and of the text. Hence, these variations result in the style of a certain text or, for that matter, in certain programming or music styles. Both of these stylistic idiosyncracies of the written text extend to the style of an author, programmer, or composer; and, to the style with which we can identify certain literary, programming, or musical peridos. Furthermore, style can also be seen in the way in which the human body moves and looks. Thus, style is a manifestation of the singular. In the sense that style does not lend itself to duplication, and provided that it happens as the apparition of an event, it exposes singularity as such. Style is thus comparable to the voice of a certain author, and also to the sound of the voice: timbre. That is to say, style and timbre can be understood equally as the presence of the singular: the signature of a unique and irreproducible quality:[1]

> In its irreplaceable quality, the timbre of the voice marks the event of language. By virtue of this fact, timbre has greater import than the *forms* of signs and the *content* of meaning. In any event, timbre cannot be summarized by form and content, since at the very least they share the capacity to be repeated, to be imitated in their identity as objects, that is, in their ideality. (Derrida, 1982, p. 296)

Endless Databases The skin of the database unfolds in the duration of the performative act. What is exposed as its singularity is the ruggedness of the traces of which it is composed. That is to say, the discontinuities of its reticulated constitution of style. The length of this skin can only be estimated: there is no possibility of rendering it complete. In this fractured state it points to infinity. In this sense, databasing means participating in the infinite, taking a small part of the infinite: performing the infinite within the limits of our own embodiment. Furthermore, the contingent situation of resonance within the frayed spatio-temporal configuration of networks relates to the concept of chaos. I have mentioned earlier the relation between computers and users as understood

[1] In signal processing terms, the sound of a voice might be approached with timbre stamps or vocoders, a type of Fourier-based filter in which "the spectrum of one sound is used to derive a filter for another" (M. Puckette, 2007).

in terms of complex systems (See 2.2). Considering databasing as chaotic systems brings yet another aspect to the contingency of style.

Database and Chaos Given that this style can be considered as an emergent singularity of databasing, we can hink of this singularity as deterministic. In mathematics, determinism refers to the capacity to predict results, specifically by solving differential equations. This is the case of dynamic systems studied within Chaos Theory. For example, the Lorenz attractor is a system of differential equations discovered by Edward N. Lorenz in 1963, following experiments on weather conditions prediction. The attractor is most famously recognized by the butterfly-like appearance of its visualization and, most interestingly, for certain fractal properties that are evienced in its graph (See Figure 6.1).[2] The Lorenz attractor is a dynamic system, which means that it can render very different and quite unpredictable results by minimal changes on their initial conditions, despite the fact that it is indeed a deterministic system. Considering databasing as a dynamic system two performances can be exactly the same if given the same initial conditions and states. In this case, databasing would be closer to the performance of digital fixed media works, in which at least at the sample level every 'bit' of it is exactly the same as the original.

[2]This butterfly shape also relates to what is known as the 'butterfly effect,' a concept of chaos brought forth by Lorenz in his 1972 lecture named: "Predictability: Does The Flap Of A Butterfly's Wings Brazil Set Off A Tornado In Texas?" (Lorenz, 1993, p. 181)

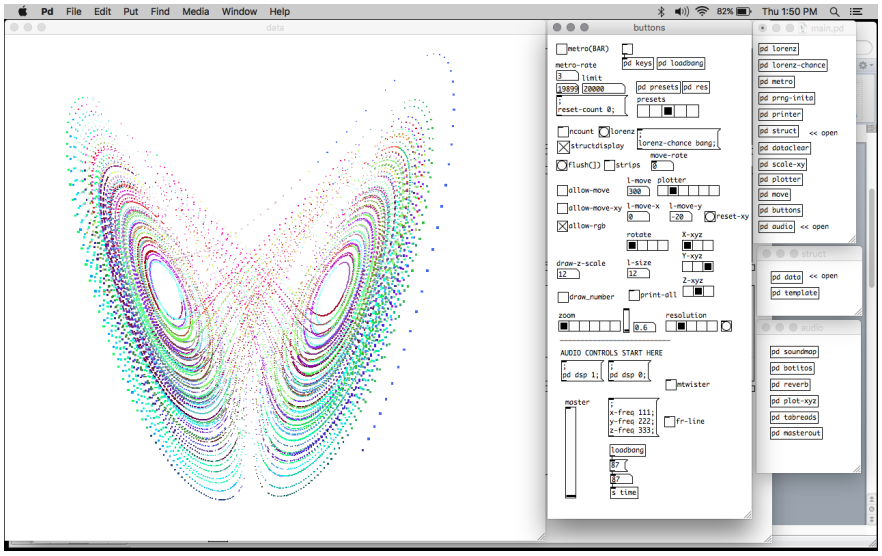


Figure 6.1: Lorenz Attractor
Plotting of the Lorenz system in Pure Data.

Fractality However, identifying predictability in this way means falling in a cybernetic trap, of which Hayles already warned about when considering Turing's Test. Hayles reads Turing's test as a game which, in order to play you are already part of its outcome because you accept its predicates as a condition for playing. In Turing's case, the moment you enter into the disembodied place where the screen is the only thing you see, you are already a cyborg, and the definition of the human and the nonhuman is already laid out in principle. On the one hand, by equating fidelity of data storage with fidelity of performance, one is already removing the human out of the concert stage, and the question of performance altogether, leaving only the idealist and romantic notion of the work of art in its pure and objective state. On the other, in order to allow for the style of databasing (skin) to emerge, one has to consider not only the actual staging of performance, also the staging of listening, allowing the resonant subject of the database to emerge as the communicative apparition of a skin. Therefore, the contingency of style (as chaotic state) can only emerge out of the unpredictable agency of the unfolding. This is how I consider databasing and the contingency of style: the unpredictability of databasing has the qualities of a fractal. Because of the fractal

dimension, it expands the definition of geometric figures to the infinite. In this sense, it presents an unfolding symmetry (self-similarity), which relates to their shapes being replicated nearly exactly in different scales.

A Music Work as a Singularity The nature of the aesthetic experience of database music slips through the cracks of traditional conceptualizations of the work of music as a result of stylistic or stipulated constraints on the part of the composer or of stochastic procedures applied to musical structures. For example, composer Horacio Vaggione goes to great lengths to prove that the musical work affirms itself as a singularity, in the particular sense that its rules are only prescribed from within, and always in an "action-perception loop" with the composer (Vaggione, 2001). What Vaggione is arguing against is the tendency of formalized musical processes that epitomize the black-box approach towards composition in Computer Aided Composition (CAC): "a composer [unlike a scientist] knows how to generate singular events, and how to articulate them in bigger and bigger chunks without losing the control of the singularities" (Vaggione, 1993, p. 97). However, there is a fundamental concern that needs to be addressed in relation to the contingency of style. For Vaggione, style comes to represent the reified status of the rules within a work, insofar as this reification is taken as the starting point from which to compose, and not the result of a composed thing. Consider this quote from an earlier text:

> Here lies what seems to be one of the sources of confusion regarding the nature of music composition processes: on the one hand, we must make as careful a distinction as possible between the collective rules and the composer's own constraints; on the other, this distinction seems irrelevant [because] any primitive (coming from a common practice or postulated ad hoc) is to be considered as a part of what is to be composed, *in order to produce a musical work affirming itself as a singularity, beyond an exercise in style*. [emphasis added] (Vaggione, 2001, p. 59)

Arbitrariness Distinguishing between rules and constraints, that is, between socially and historically established canons as stylistic conventions, and locally established postulates to be carried

out by the composer as constraints, is crucial, simultaneously, in defining style in composition and databasing. However, conventions and constraints collapse into the realm of compositional arbitrariness for, if any "primitive" of the composition is to be considered "part of what is to be composed," then style itself becomes a result. This is what Vaggione means by "beyond an *exercise* in style:" it is not an exercise in the sense of a draft, in the military context of training (practice for the sake of training). Style is not an exercise because it cannot be operative in the sense that it is considered a product of work. Style is contingent, emerging from the performative action of databasing. Style would only result in closure if considered teleologically: a closed object, stipulated from the start as a law to which every composable element abides. Vaggione calls these overarching laws 'global laws,' and he compares them with a marching army following a 'one-two' directive, where "no singularities... are... allowed" (Vaggione, 1993, p. 101).

Inoperative Style As mentioned earlier, inoperativity can be understood as a feature of the activity of work that allows the music work to distinguish from notions of production, product, and completion. An inoperative style can be understood, therefore, as a contingency that appears in the form of exposure, not as a closed object, but as an unclosed object; some *thing* that is exposed and bound to exposure; a thing that exposes us in the same resonance of its touch. Another word I have given to this 'thing' in exposition is 'infraskin,' which is where this inoperative style would be imprinted. Inoperative style does not mean it is a passive style. As I described before, activity is what defines style. Therefore, to speak of an inoperative style means to place ontology at the limit: whatever style databasers perform becomes the style that defines them but, this definition is never achieved, it simply leaves a trace. Like the marks on our skin, like its wounds; like the cracks of an old house, like debris, wreckages, or any form of residual mark that is the evidence of an event; with forensic intimacy, the contingent style of a musical unwork reveals itself as communication. This skin is what connects aesthetic experience of style with forensic or after-the-fact musical analysis as well as with an encounter with the spectral. Furthermore, this is how the spectral cannot

be but a result of the inoperative, of that which escapes the limits of the work. Like the timbre of Lucier's voice that, releasing from itself into the room, then returns back as the resonance of a self. This 'voice' of the unwork is what is 'invisible' to the work. Invisible, because neither the inner voice in one's head, nor the actual timbre of the voice as one hears it are accessible to us. We can only hear this voice transduced, and from the perspective of others. It is what we can never listen and yet, in being hidden or silenced from us is how it becomes available for listening, what begins listening at the first staging of the waves: the strength of the first 'I' in Lucier's work with no first breath. Severing the voice from the impulse of the body requires an insurmountable amount of activity, even if it means cutting a magnetic tape, or applying an offset when reading a sample buffer. An inoperative style depends on this excess of activity.

6.4 A Specter Of Authority

> Gender is instituted through the stylization of the body and, hence, must be understood as the mundane way in which bodily gestures, movements, and enactments of various kinds *constitute the illusion of an abiding gendered self.* [emphasis added] (Butler, 1988, p. 519)

The figure of the author (composer/databaser) is, to a certain extent, expanded through the network by the complexity of the system: the composer's agency and compositional authority is distributed to the various agents of the network (database, interface, sounds, etc). However, authority is reified into the 'name' of the author because of the interplay among work, productivity, and product. In this section I attempt an approach to the 'name' of the composer not by its work, but from the illusory perspective of authority. However composable all Vaggonian primitives can be, the structure of the database tree is so vast that any attempt to comprehend it as a whole would extend it even further (See 4.3). However, this determines neither the extent of the performativity of databasing, nor the agency of the human. Quite the contrary, expansion through the network can be considered as the trace of the author, or better, the elongation of the spectral shape of an author.

Further, with the performativity of databasing, the databaser too becomes incomplete.

The Name The infinitude in the fractality of databasing, however, is at some point reified in a figure or a name. This figure is the place where authority is condensed, and it responds to traditionally essentialist conceptualizations of the romantic author which, despite the many attempts during 20th century,[3] are still in effect today, specifically in the field of music composition. It is not the purpose of this section to criticize this tradition, namely because I don't consider it relevant for the purposes of databasing. Focusing on it would be missing the point. That is to say, in the case of databasing, such figure of an essential author is simply dislocated and forced upon the structure of the network, and it is anachronic because it constitutes a temporality set against the temporality of networks. Databasing, as resonant performativity already exists beyond this traditional figure of the author. However, in its spectrality that stems from the archontic (See 5.2), authority can be seen as the illusory resonance of an author. It is this illusion that I attempt to address here; a ghost that haunts music composition.

Dictionaries Consider how style is used in some cases of Computer Aided Composition (CAC). David Cope's Experiments in Music Intelligence (EMI) (Cope, 1987b), for example, can be considered a formalization of compositional authority. That is to say, intentional stylization "based on a large database of style descriptions, or rules, of different compositional strategies" (IV, 1999, p. 3). Written in the functional programming language LISt Processor (LISP), EMI's focus is "style imitation" in order to assist the composer when in front of a "composing block," provoking the "author into almost immediate action. Any blank moments along the way are immediately filled by simple queries…" (Cope, 1987a, p. 38). Cope's approach is inherently hierarchical, and thus based on the premise that music is a language. Therefore, Cope designed dictionaries (databases) of Musical Instrument Digital Interface (MIDI) scores representing the internal rela-

[3] See for example Roland Barthe's 1967 *Death of the Author*, or Michel Foucault's 1969 text *What is an author?*, both of them commented on in (Daniel, 2007).

tions between composed elements. From items in the dictionary, logically correct inferences are drawn (predicate calculus) (Cope, 1987b, p. 1). Thus, EMI is aimed at generalizations that reify the authority of the composer as style:

> Years of consistent interactive use have resulted in dictionaries which so complement the author's own style that compositions show little evidence of the origins (man/machine) of the music. (ibid., p. 179)

Artistry Vaggione, in response to a formalized approach to music —among many that exist in the literature (Hiller & Isaacson, 1959; Xenakis, 1992; Truax, 1976; Ariza, 2005a)—, proposes the equal role of the informal craftsmanship of the composer using computers. In a very different case of the use of databases, consider Roads' account of Vaggione's workflow when composing the work *SHALL*:

> These involved arranging microsounds[4] using a sound mixing program with a graphical time-line interface. He would load a *catalog of pre-edited microsound* into the program's library then select items and paste them onto a track at specific points on the timeline running from left to right across the screen. By pasting a single particle multiple times, it became a sound entity of a higher temporal order. Each paste operation was like a stroke of a brush in a painting, adding a touch more color over the blank space of the canvas. In this case, *the collection of microsounds in the library can be thought of as a palette*. Since the program allowed the user to zoom in or out in time, the composer could paste and edit on different time scales. The program offered multiple simultaneous tracks on which to paste, permitting a rich interplay of microevents. [emphasis added] (Curtis Roads, 2001, pp. 313–314)

While this workflow is only representative of certain aspect of the piece in question, it does serve as an example of his concept of craftsmanship. Craftsmanship refers to the manual and direct action of the hand of the composer. The hand, as Makis Solomos very well points out, is not to be understood as being without the tool (mouse) that it needs to use in order to precisely locate sounds on the timeline interface (Solomos, 2005, p. 4). Craftsmanship might be better understood, however, as 'artistry,' thus keeping its relation to hand-made crafts, while

[4]The word 'microsound' refers to sonic events shaped below the threshold of the 'note.' See (Curtis Roads, 2001)

maintaining a link with articulation, one of Vaggione's crucial concepts. While articulation relates to the composer's operativity on multiple time scales, artistry relates to the arbitrariness of choice. It is thus a reaction to the abundance of radical formalism and automation in CAC (ibid., p. 3). Therefore, Vaggione writes, "to write music 'manually', note by note, partial by partial, or grain by grain, is an approach proper to a composer, and he should not be embarrassed about using this aspect of his craftsmanship"' (ibid., p. 3). Vaggione built his terminology not in opposition, but in the spirit of reconfiguring CAC from an embodied stance coming from outside information theory. This stance is not only evident in Vaggione's writings and music. To a debatable extent, this stance is a point of departure to think of a branch of Argentinian electroacoustic identity that developed in France.[5]

The Work of Mice For Vaggione, instead of relying on the rule-based programming of formalization processes alone (keyboard-based input), the artistry of the composer resides in the use of the mouse. The timeline of the sequence interface, and its workflow depends on the mouse pointer. The presence of the composer's hand is evidenced by the trajectory or course of the pointer. The mouse, along with the history of clicks and drag-n-drop motions suggests the spectral presence of the author. The mouse pointer, the 'stilus', like the writing device, becomes that with which we resonate as listeners. Thus, we perceive the marks of an authorial skin in database music. The Vaggionian singularity-based approach to authority embeds composers and computers in a complex system or network, that renders the world of music with computers as a hybrid between human and nonhuman. This is how the specter of the author coexists with the specter of the database, and thus, how databasing and composition reveal themselves to be instances of a performativity that resonate aesthetically through the work of music.

[5]For example, in the work of Beatriz Ferreyra, Elsa Justel, Mario Mary, to name a few. For an approach to Justel's timeline-based spatialization techniques, see (Cámara Halac, 2018b).

Chapter 7

Rethinking Composition

7.1 Interlude: Hyperbolic Reactions

Imagining Composers In today's composition and databasing practices, the probabilities of a composer or a databaser working without computers are very slim. Databasing or composition outside the digital seems rather fictional. However, the very image of a 'composer,' which traditionally stems from romantic standards is, already, outside the world of computers. This image of composing can be painted as follows: the composer at work, quietly on a desk with pen and paper, transcribing, arranging, making parts, drawing line after line, dot after dot, notating instructions for the performance of an imagined music. Where is the computer in this image of composition? Certainly, placing a computer on this idyllic desk would be anachronistic and obtrusive; anachronistic, since the romantic quality of the scene would point to the fact that personal desktop computers were not available until late in the 20th century; obtrusive, in the sense that a computer would play against this romantic composer whose motivations were of a different nature than those we can find today. This reification of the composer already precludes not only the digital, but also the many technological devices that have entered music composition over the years, such as tape recorders, or electronics in general. These technological devices have redefined the composer in

many ways. In sum, a composer without computers cannot be imagined today, but this is not due to the practice of composition itself. My argument here is that in any given situation, it is hardly possible to imagine a human without computers at all. This is what media studies has to teach us about the posthuman condition in which we hybridly live, where humans and technology, humans and nonhumans, unfold as interminably networked traces.

Playing with Shadows Georgina Born's ethnography of Institut de Recherche et Coordination Acoustique/Musique (IRCAM) (Born, 1995) captured how the institutionalization of music composition and technology resulted in hierarchical structures of work dynamics, and how these were coated with false notions of collaboration. Inequalities of social, economical, and political status among technicians and composers within IRCAM became privately evident. Knowing how to use computers and knowing how to compose comprised two irreconcilable poles in the institutional structure. For example, Born described internal hierarchies such as 'superuser' password knowledge, source code access, software licences, and, in some cases, she showed how these hierarchies reflected on internal privacy issues: "workers concocted their various informal ways of protecting privacy and retaining secrecy: blocking the glass walls of their studies, working at night to prevent others from knowing what they were doing or even whether they were working at all" (ibid., p. 272). On the one hand, it is tempting to link this irreconciliation to the extreme reification of the name Pierre Boulez. The obscure dynamics behind this reification, however privately and secretly they were kept within the institution, can be nonetheless seen as the shadow of the more general specter of the music maker. Born's mysterious but telling anonymization of everyone but Boulez on her transcriptions might attest to this shadow. The music maker has been traditionally considered an outsider, marginalized by society, but simultaneously an integrator of society (Attali, 2009, p. 12).

On the other hand, this shadow might also be that of the computer, the structural presence of a fictional intelligence constructed upon first wave cybernetics. That is to say, precisely

because the computer projects an insurmountable power that comes from its calculation potential, the human is inevitably bound to be a subordinate, and with this subordination comes the subordination of the composer, and (perhaps) the end of music. This hyperbolic reaction would explain the need for privacy and secrecy of information, the undocumented "oral culture" of Born's IRCAM, as well as the reversal of the human-computer subordination evidenced in the social strata of the institution. To a certain extent, this impulse to protect the secret can be seen as an after-shock of the earthquake-type clash with which IRCAM is composed: a hierarchical archivization of music composition and technology.

Composers Without Computers Composing with and without computers cannot be seen as poles on a continuum. At the risk of drawing a straw man out of this computer-less composer, it is very unlikely in today's world to imagine a composer who has not googled 'clarinet multiphonics' for more than a few YouTube tutorials on the topic. Likewise for digitized music listening: in order to escape it, one has to go to great cult-like lengths to do so: going to instrumental performances, getting a vinyl record or a tape player, etc. To have a concert, therefore, a composer without computers today would need to whisper the score to the performers who would, in turn, play by ear. ('By ear', in the sense that they would need to play from memory, since no printed score would exist, for even if the composer wrote the parts, the score would have to be inscribed on a paper, and somewhere along paper networks there is at least one computer.) The composer should also whisper invitations to a few neighbors to be part of the audience. The composer should also demand no recordings whatsoever, while performing for an audience that has been kindly reminded not to bring their cellphones. Even then, the concert would need to take place on an amphitheater to avoid architectural networks, and Automatic Computer Assisted Design (AutoCAD); before the sun sets, to avoid electricity networks altogether while we are at it; away from cities, a car driving by would be unforgivable; so far away that we would, in fact, need to bring non-perishables for the pilgrimage, and even then, packaging networks or agriculture networks would be almost

impossible to avoid. And this is precisely the point: in attempting to avoid it, the pilgrimage exists not in space, but in time, and thus it enters into the realm of fiction. The same can be applied to the overloaded case of a composer totally *with* computers, that is, a computer-composer not needing a human to write music.

Databasing Without Computers The same applies to databasing: removing computers altogether from databasing takes us to the world of libraries, encyclopedias, collectors, gatherers. Most important, it takes us to the place databasing occupies within society, the dynamics of archivization and institutionalization. That is to say, it relates performance with the archontic, with the Oedipal drive to re-place (See 5.2), to an infinite return that the structure of the archive imposes upon us. So, if we imagine a computer-less census, we'd have to picture a gatherer of names walking around town, asking out loud for each person's name and place of residence. Getting rid of networks which might have computers —as in the case of the above painted computer-less composer—, it is clear that the only suitable person for the job would be Irineo Funes (See 1.7), and the only possible storage medium would be his memory.[1]

Hopefully, the reader would consider this resort to hyperbolic fictions less as a means of justification of the hybrid condition of composition and databasing, and more as an absurd parenthesis that brings no criticism to the —still valid— efforts of working 'outside' the digital. These efforts are not questioned in regards to their validity, only in terms of their definition, which, for the purposes of my dissertation, is understood as built upon a particular concept of man: man as a unity, as whole, and as the one.

[1] Although censuses are still carried out similarly —i.e., people coming into your house and filling out a form—, given the vast amounts of collected population data very little statistical information could be derived without the use of databases.

7.2 Working Composition

The Work Problem What does the problem of the music work consist of? and, why is it a problem? As I have already described in relation to community (See 4.4), work can be thought of in two ways: work as teleology or work as ontology. In both cases, the word 'work' uses its two meanings: the first meaning is that of the activity of working (labor, effort), which points to a series of meanings that I will explain below. The second meaning is that of the finished activity, what is traditionally referred to as a music 'work' (product, composition).

When a music work is understood teleologically, work acquires an aim, and it is measured in terms of this aim. Therefore, one can say 'this music works,' 'it sustains itself,' 'it is very well structured;' or 'this just doesn't work,' 'it fell apart.' These expressions generally refer to what the music 'proposes' and what it ultimately produces, and how these two (proposition, product) relate. We can understand this as what Peter Szendy (2008) calls the "modernist regime of listening," in which the music work shapes its listeners into an ideal listener: "the work listens to itself" (ibid., p. 127). What this "listening without listener" refers to is a certain gap between the listener and the listened: between the subject and the object. For Szendy, this modernist regime is based on a more fundamental aspect: the absorption of the listener by the work. In this dynamics of absorption, "distracted" listeners "fall away like a dead limb" and "bring nothing to the great *corpus* of the work" (ibid., p. 127). On the one hand, 'distraction' in the listener relates to an inability to listen structurally, to maintain and analyze the relations of the different elements of the musical discourse. However, this 'inability' is measured against the standards of the music work, which 'tells' you how to listen. Therefore, a distracted listener pays no attention to the way the work should be listened, and it is left 'outside.' On the other, the notion of a corpus of music work (understood broadly as the *oeuvre* throughout a composer's lifetime) relates to the concept of the archive. The fact that a listener would remain 'outside' speaks of the filtering activity of archives. In sum, the moment the music work begins to act as 'work,' its listening is predetermined neither

by the physicality of the waves in media, nor by the virtuality inherent in perception, but by a teleology of work.

Music Unwork When understood ontologically, work has no purpose other than being. 'Work,' in relation to music, begins to separate from itself. What is this 'itself' and how does work separate or withdraw from it? We can approach at least six channels that would help better define the constitution of work: productivity; objectification; convergence; completion, integrity, or organicity; unification; and consignation. First, the productivity of work stops, not in the sense that there is no longer any activity of 'production,' but rather because there is no longer a notion of a finished product. Second, the objectification of the music work (the work-as-object) loses its retaliation, that is, the 'object' stops 'absorbing' the listener. Third, instead of convergence into the 'one' of the work, we have a certain divergence that instead of being 'more' than one, it becomes always 'less.' That is to say, the music work behaves in a way similar to what I described of Latour's network: every node points to every other node, which leads to the network's expansion; but, this expansion is never realized entirely, and every node returns a 'size' of the network that is always less than the one immediately after. Fourth, the work of music is no longer a whole: it is not an 'organicity,' but an in-organicity that relates to the destruction drive I have described of archives. Both listener and work become incomplete (like Nancy's self, interrupted and suspended), an in such way we can understand their disintegration: the 'dead limb' is not the inattentive listener any more, but rather, the concept of work itself. Fifth, the music work stops pursuing unity, and it is instead segregated. What this entails is that, in its core, its disintegration is a way for the music work to sever itself from its own historical constitution. In this sense, we can speak of a break of music composition with its past. This severing constitutes a break because, at once, it erases and inscribes its consignation. That is to say, in being an anarchic breakage, the ontological understanding of work opens up an aesthetic space for imagination, while nonetheless still remaining under the spell of archives, under their constitution. In this sense, a certain nostalgia of the un-

work should not misguide us into inactivity. On the contrary, in music composition today we can still engage in resonance with this spectral 'feature' of the music work, we can still address the powerful force that drives the archive, and this addressing is something that occurs in databasing.

A Severed Work What constitutes, then, that moment when the music work becomes a work? How is it possible for the work to become a thing, for the object to become the ruler, for the regime to be built on the first place, if the resonant space is already an inoperative space, interrupted and suspended? I would like to invert Szendy's metaphor of the inattentive listener as a fallen limb, and propose that it is the music work itself what falls away, the moment that it becomes a finished thing. Like the human in Kittler's digitally converged apocalypse, redundancy is out of the question. Redundancy in terms of the human being absorbed by (nonhuman) technology. Uselessness is left at the gates of the majestic concert hall, with the rest of (useless) humans: it is literally and conceptually placed outside architecture itself. The created work, in its essential nature of being a cohesive, coherent whole, separates itself from the world of mechanical waves, and forms the one and only work: the piece of music. It is a 'piece' not because it is in itself incomplete, but because it is the piece of the whole of the work of a composer.

Absorption For Szendy, the ultimate aim of this modern regime of listening is the absorption of the listener by the work. Not surprisingly, 'absorption' is the key concept in Iannis Xenakis' narrative of the four stages of degradation of Western Music's "outside-time structures," in his article *Towards a Metamusic* (1967): "we can see a phenomenon of absorption of the ancient enharmonic by the diatonic. This must have taken place during the first centuries of Christianity, as part of the Church fathers' struggle against paganism and certain of its manifestations in the arts…" Later, referring to larger structural groupings: "this phenomenon of absorption is comparable to that of the scales (or modes) of the Renaissance by the major diatonic scale, which perpetuates the ancient syntonon diatonic…" Finally, "one can observe the phenomenon of the absorption of imperfect

octaves by the perfect octave by virtue of the basic rules of consonance" (Xenakis, 1992, pp. 189–190). The final stage of this process of absorption and degradation comes with atonalism, which "practically abandoned all outside-time structure" (ibid., p. 193). However, Xenakis' narrative contextualizes his sieve theory, devised as a means to "establish for the first time an axiomatic system, and to bring forth a formalization which will unify the ancient past, the present, and the future" (ibid., p. 182). Thus, Xenakis formulated this theory with computers in mind, that is, with its concrete application in computer programs, under the subtitle "suprastructures" (ibid., p. 200). The logic of absorption in Xenakis' sieves suggests that the 'suprastructures' of the computer can now contain the key to the ultimate absorption: not only the outside-time structures, but also the modernist 'regime' of listening itself. The music works made under this systematization, would prove to be extremely organic and based on an overly modern gesture towards unity, metastructure, and mechanization. If we can take this comparison to a hyperbolic extreme: if the computer (then) had this ability to return music structures, then, the movement of absorption would take place by the computer: the computer program would become the music work absorbing its listeners. These conjectures serve, if anything, as a gateway to understand the context in which Xenakis was embedded when writing his sieves program, which was built in reaction to the "poison that is discharged into our ears" as he witnessed the "industrialization of music [that] already floods our ears in many public places, shops, radio, TV, and airlines, the world over" (ibid., p. 200). In this sense, these overflooded ears would find a remedy by systematizing to the extreme the whole frequency range into a database (the sieves) that could be queried with algebraic expressions. I will return to this point later. Nevertheless, we can ask ourselves where is the poison that Xenakis the architect and composer was identifying with 'industrialized' music? Is Xenakis not a product of modernity itself, as the work that listened to itself to the point of shaping a Xenakis-listener-node?

7.3 The Composer As Navigator

> I am motivated to present this architecture, which is linked to antiquity and doubtless to other cultures, because it is an elegant and lively witness to what I have tried to define as an outside-time category, *algebra*, or structure of music, as opposed to its other two categories, in-time and temporal. [emphasis added] (ibid., p. 192)

> With [the relational] model any formatted data base is viewed as a collection of time-varying relations of assorted degrees... this collection is called a *relational algebra*... a query language could be directly based on it... The primary purpose of [relational] algebra is to provide a collection of operations on relations of all degrees... suitable for selecting data from a relational data base. [emphasis added] (Codd, 1972, pp. 1–5)

Querying the Sieves If we consider pitches as an outside-time (relational) database, one way of understanding Xenakis' sieve theory is as a query method, for which E.F. Codd's Relational model would fit perfectly. The nature of this consideration stems from the application of algebra as a programmable selection mechanism or simply, filters. Both concepts (sieves and relational algebra) have a common link which, not surprisingly, is the IBM-7090 mainframe computer.[2] While Xenakis' experiments were carried out on the IBM-7090 mainframe computer located at IBM-France in Paris, Codd himself worked at the IBM Research Laboratory in San Jose, California. Furthermore, this same computer was used by Hiller and Baker in their realization of MUsic SImulator-Interpreter for COMpositional Procedures (MUSICOMP), a pioneering language for algorithmic composition (Ariza, 2005a, p. 44). Most important, the IBM-7090 used the programming language FORTRAN IV, as can be seen by the printed FORTRAN routines for Xenakis' 1962 work *Atrées (ST/10-3 060962)* (Xenakis, 1992, p. 145).[4] Xenakis's work on sieves came a few years after his experiments on the IBM-7090, and his sieves program was written in Basic

[2] Among other things, the IBM-7090 computer was used in the computation of the first 100,000 digits of π (Shanks & W.jun. Wrench, 1962), Roger Shepard's computation of the homonymous 'shepard' tone (N. Shepard, 1964), Alexander Hurwitz's computation of the 19th and 20th mersenne prime numbers,[3] and Peter Sellers' plot-twisting moment in Stanley Kubrick's "Dr. Strangelove or: How I Learned to Stop Worrying and Love the Bomb:" https://en.wikipedia.org/wiki/Dr._Strangelove

[4] Interestingly, given that Ariza finds Xenakis' sieves code unusable (ibid., p. 1), chances are that the printed code for the ST/10-3 composition is likewise useless.

and then in C. However, the experience with FORTRAN IV at the IBM-7090 serves nonetheless as a common ancestor to both Xenakis and Codd. For example, Xenakis' transcriptions in early Computer Aided Composition (CAC) systems were performed with tables of outputted computer data. Further, Ariza (ibid.) writes how "the early systems of Hiller, Xenakis, and Koenig all required manual transcription of computer output into Western notation. The computer output of these early systems was in the form of *alpha-numeric data tables*: each row represents an event, each column represents an event parameter value" [emphasis added] (ibid., p. 94). In this sense, performance in CAC meant interpreting results out of a database.[5]

Sound Synthesis Parenthesis (Before continuing, a sound synthesis parenthesis must be opened. While Xenakis praised the speed at which the IBM-7090 could perform computations, Max Mathews (Mathews, 1963), then director of the Behavioral Research Laboratory at Bell Telephone Laboratories, wrote:

> A high-speed machine such as the IBM-7090, using the programs described later in this article, can compute *only about 5000 numbers per second* when generating a reasonably complex sound. However, the numbers can be temporarily stored on one of the computer's digital magnetic tapes, and this tape can subsequently be replayed at rates up to 30,000 numbers per second (each number being a 12-bit binary number). [emphasis added] (ibid., p. 553)

Mathews' concern for speed was grounded on the need to achieve sound synthesis, which meant fast computations of the sample theorem. Initially, the first synthesized sound was obtained in 1957, with the (assembly-code written) MUSIC 1 program with the IBM-704 (a predecessor of the IBM-7090). Later, when Bell Labs obtained the IBM-7094 —which "was a very, very effective machine" (C. Roads & Mathews, 1980, p. 16)—, and in combination with the (then) widely available FORTRAN compiler, Mathews could develop the MUSIC I program, into MUSIC V, which became the first portable computer music language designed for computer music synthesis.[6] I

[5]For further reference on the early uses of computers in CAC, I refer the reader to Ariza's PhD thesis (ibid.).

[6]As an example, I would refer the reader to James Tenney's work from 1962 "Five Stochastic Studies," which can

return to this discussion in 7.4)

Algebraic Abstractions for Freedom Xenakis' and Codd's papers came out around the same time: Xenakis' English publication of *Towards a Metamusic* was in 1970, Codd's papers were published in 1970 and 1972. Sieve theory was aimed at providing a plethora of computable sets (or relations) of pitches, according to different temperings of the smallest displacement unit and the selected value for the modulo operator. Contemporaneously, Codd's relational algebra was meant to be the internal structure of a query language for selecting elements based on their relations. Both of these can be considered algebraic abstractions of a selection process. In the case of Xenakis, the abstraction was outside-time, as the composer could make a snapshot, or a tomography of the pitch space in order to analyze it and extrapolate structural relations. In Codd's case, the abstraction was spatial: the query language would be separated from the database, allowing databasers to perform queries in the 'frontend' without worrying about internal data structures, memory allocation, and so on, since these operations would occur in the background of the 'backend.' By black-boxing and hardware-specific programming, both methods freed the human operator to devise any kind of algebraic queries. Thus, the operator worked on a higher level of abstraction that enabled a less problematic kind of envisioning. Xenakis wrote convinced: "freed from tedious calculations, the composer is able to devote himself to the general problems that the new musical form poses, and to explore the nooks and crannies of this form while *modifying the values of the input data…*" [emphasis added] (Xenakis, 1992, p. 144).

A Cosmic Vessel and an Armchair Therefore, the composer delegates to the computer the minutiae of arduous iterative computations: precisely what the computer is better at than the human. As a result, in Xenakis' view, and in resonance with programmer Charles Bachman's claim

be found on a YouTube account on his name: https://www.youtube.com/channel/UCEzSaoPnxCJVzXxA9obuRWg/videos. Roads, while interviewing Matthews recalls this piece to be named "Noise Studies" (C. Roads & Mathews, 1980, p. 18), which fades out the reference to Xenakis' music.

in *The Programmer as Navigator* (Bachman, 1973), the composer became a pilot:

> With the aid of electronic computers the composer becomes a sort of pilot: he presses the buttons, introduces coordinates, and supervises the controls of *a cosmic vessel sailing in the space of sound*, across sonic constellations and galaxies that he could formerly glimpse only as a distant dream. *Now he can explore them at his ease, seated in an armchair* [emphasis added] (Xenakis, 1992, p. 144)

Codd's and Xenakis' propositions were abstractions deeply rooted in and contextualized against a backdrop of their own fields. Xenakis wrote against the current state of Western Music with its "degradation of outside-time structures", the "followers of information theory" and the "intuitionists."[7] Codd wrote against the previously developed hierarchical and network database models. Most important, these tools and their development had the human operator's considerations in mind. The composer, like the databaser, would engage in a rudimentary and limited, but still present, feedback process at the *input* level. That is to say, unless rewriting the code, which consisted in a very long and economically expensive process combining punch cards and magnetic tapes, the composer and the databaser could change the input several times, achieving different outputs in a matter of hours.[8] For example, queries made on the relational model would appear on screen at a very fast rate, thus enabling better tuning of the input in relation to a wanted output. Likewise, the composer could modify the input values to highly complex calculations that would otherwise take a long time, or be error prone. The limitation, of course, is the level of intervention with the code, which the overall circuitry would thus complicate; criticism on account of this shortcoming of the circuit would thus be rendered anachronistic, but recalling these limitations places composition and databasing in perspective.

[7] Xenakis defined both groups as poles on an extreme. On the first extreme ("followers of information theory"), and on the other ("intuitionists"), Xenakis identified two groups: "graphists" and those involved with "happenings" (Xenakis, 1992, p. 180). We can only speculate that Xenakis was referring to two American music currents during the 1950s: Hiller and Isaacson can be linked to the information theorists, and the intuitionists can be represented by Cage and the New York School composers.

[8] As a reference, the computation of the first 100,000 values of π took about eight and a half hours (Shanks & W.jun. Wrench, 1962).

7.4 The Database As Performer

I would like to take an improvisational turn that would make Xenakis fall off his armchair. Xenakis' fall would be contemplated against the spirit of the later discussions on interaction that came with George Lewis and *Voyager* (G. Lewis, 2000; G. E. Lewis, 1999; Rowe et al., 1993). Lewis called his approach an "improvisational, nonhierarchical, subject-subject model of discourse, rather than a stimulus/response setup" (G. E. Lewis, 1999, p. 104). Thus, the activity of the composer was reconfigured in a networked relation *with* the computer. That is to say, Xenakis' metaphor of the composer as pilot turns upside down, altogether reconfiguring the navigational metaphor: the ship begins to navigate itself.

The Computer as a Musical Instrument It is now pertintent to bring back Max Mathews paper "The Digital Computer as a Musical Instrument" (Mathews, 1963). The architecture of MUSIC-V is founded upon the concept of the computer as an instrument that the composer executes by providing it a score. The three stages of data flow (reading, sorting, and executing) can be understood as modeled with three music concepts: score, conductor, and instrument. Therefore, it can be argued that MUSIC-V left composers, performers, and programmers on the margins of its data flow, and that by this elision a new identity was in the makings. On the one hand, the hybrid musical instrument that the computer represented already subsumed three concepts into one, resulting in a hybrid score/conductor/instrument. On the other, by this elision, the three missing human terms have now been subsumed anew, forming a hybrid definition of composer/performer/programmer. In any case, this hybridity is evident in the music works that I have cited in earlier sections (See 3.4).

> So far I have described use of the computer solely as a musical instrument. The composer writes one line of parameters for each note he wishes played and hence has complete control of the note. He is omnipotent, except for lack of control over the noise produced by the random-number unit generators. *Here a minor liberty is allowed the computer.* [emphasis added] (ibid., p. 557)

A Minor Liberty As can be read at the end of the introduction to MUSIC-V (ibid.), the extent of this "minor liberty" was measured against Hiller and Isaacson's previous work (Hiller & Isaacson, 1959), which Mathews describes as an extreme case of the computer as composer: "the computer can be given a set of rules, plus a random-number generator, and can simply be *turned on* to generate any amount of music" (Mathews, 1963, p. 557). On the one hand, Mathews' argument is based on the "omnipotence" of the composer in front of the computer. Control of the music work is not something that can be delegated to the computer, unless it comprises lengthy calculations of pseudorandomness. On the other hand, as Ariza has shown, the computer output of early Computer Aided Composition (CAC) has been often misconceived in the literature as directly musical output, disregarding the extensive transcription work on the part of composers (Ariza, 2005a). Nonetheless, Mathews' "minor liberty" can be considered as a reassurance for the reader that computers would not take control over music, let alone over the world. As I see it, arguing for control while granting some liberty relates to a negotiation between a composer's work and computer time. Because of the correlation between sonic complexity and parameter input, "the composer must make his own compromise between interest, cost, and work" (Mathews, 1963, p. 555). Pseudorandom generators introduced complexity in an efficient way (Cámara Halac, 2018a). Therefore, in an economical choice, arguing for omnipotence allowed for some aesthetics agency to come from computers.

The Computer as a Player Rowe (ibid.) identified two paradigms within interactive systems: *instrument* and *player*. The instrument paradigm comprises systems in which performance gestures are sensed (collecting gestural data), processed (reading and interpreting data), and then a sonic output is elaborated in the form of a response. The player paradigm comprises the creation of "an artificial player, a musical presence with a personality and behavior of its own…" (ibid., Chapter 1). Therefore, the system contains embedded processes that grant a great level of independence. This means that the composer intentionally relinquishes control of the artwork's structure to

the system. Like Vaggione's concept of the computer as a complex system in which the composer "is imbedded in a network within which he or she can act, design, and experience concrete tools and (meaningful) musical situations" (Vaggione, 2001), the human node breaks the traditionally hierarchical structure of composer-work, arriving at a distributed authority of the work among the elements of the system. For example, in George G. E. Lewis (1999)'s *Voyager*, "the computer system is not an instrument, and therefore cannot be controlled by a performer. Rather, the system is a multi-instrumental player with its own instrument" (ibid., p. 103). The computer becomes an improvisation partner. While the limitations of computer capabilities precluded more complex conceptualizations of the type of interactivity between computer and composer in MUSIC-V, as personal computers became affordable the type of negotiations no longer depended on economic decisions. For Lewis, this negotiation existed sonically between computer and improviser:

> There is no built-in hierarchy of human leader/computer follower, no 'veto' buttons, pedals, or cues. All communication between the system and the improviser takes place sonically. A performance of Voyager is in a very real sense the result of a process of negotiation between the computer and the improviser. (ibid., p. 104)

Programming Decisions However, in order to implement concepts coming from artificial intelligence such as machine listening and learning, the complexity of the program increases exponentially. In light of the difficulties arising from programming large software, and in response to Lewis' criticism of the Max patching paradigm rooted on trigger-based interactivity, Miller Puckette responds: "If you wish your computer to be more than just a musical instrument —if you want it to be an improvisation partner, for instance— you need a programming language. One thing people in this situation might want to do is write Max external C procedure" (Rowe et al., 1993, p. 8). As Rowe writes:

> To arrive at a more sophisticated interaction, or *cooperation*, the system must be able to understand the directions and goals of a human counterpart sufficiently to predict where those directions will lead and must know enough about composition to be able to reinforce the goals at the same moment as they are achieved in the human performance.

(Rowe, 1992, Chapter 8)

The player paradigm and its subsequence reconfiguration of compositional authority is possible by means of a database: the computer stores features during the course of the performance, which are then analyzed over time, and which serve as guides for the sonic outcome on the part of the computer. As I mentioned earlier, while the guidance of the database provides paths through uncharted territories, it also hides other paths (See 3.3.1). *Voyager* indeed brings interactivity between humans and nonhumans to another stage, and because of it, music composition can be seen differently. However, the intricacies of programming decisions still play a role in the musical outcome, specifically in the modelling of musical concepts within data structures.

Anachronistic Composers This notion of interactivity differs greatly from Xenakis' model of the (modern) composer. He is sitting quietly in his armchair pressing buttons in 1962. By pressing them and inputting certain values, he controls the output, since he knows beforehand the internal mechanisms that are embedded in the software. This image of the modern composer in front of computer technology can also be found in, for example, Edgar Varèse: "The computing machine is a marvelous invention and seems almost superhuman. But, in reality, it is as limited as the mind of the individual who feeds it material" (Varese, 2004, p. 20). Varèse's words, however, refer to the creative limit that a computer might have, which is always a function of the input and, by extension, of material itself. Furthermore, in relation to electronic technology, he writes: "like the computer, the machines we use for making music can only give back what we put into them" (ibid., p. 20). Therefore, from these images of Varese-composer and Xenakis-composer, two axioms can be extrapolated: first, that composers do not lose control of the output; second, that the way to interact with computers is precisely by telling them what and when to do it, so that the user is in total operative control. It is against these two axioms of computers and composition that Lewis' work in the late 1980s and 1990s can be contextualized. More precisely, it is because of the anachronic presence of the modern 'eurocentric' composer, and of its popularity among computer

music history, that Lewis brings into surface the question of interactivity.

[bang(Placing Max into perspective by commenting on the social and cultural environment of computer music of the late 1980s, Lewis writes:

> [Lewis:] 'interaction' in computer music has moved from being considered the province of kooks and charlatans (I'm proud to have been one of those), to a position where composers now feel obliged to 'go interactive' in order to stay abreast of newer developments in the field (Rowe et al., 1993, p. 11).

The way in which interactivity was conceived in the 'interactive' music made with Max was, for Lewis, determined by a fundamental feature of program —the 'trigger'—, which, in turn, was grounded on a more general programming concept: the conception of the patching window as a digital equivalent to the analog synthesizer's patching mechanism, where graphic cords are equivalent to cables, equating data flow with voltage flow. Nonetheless, the trigger ('bang') is a feature, not a bug, unless it is used as an extension of the stimulus/response paradigm of interactivity. In other words, in resonance with Vaggione (See 6.3), subordinating music events to triggers by a human operator brings out a certain military metaphor which Lewis calls "hear and obey" (ibid., p. 11). This metaphor can easily be extended to that of weaponry, and to 'bang,' the unfortunate naming of the method which (generally) triggers an object's core routine.[9] In order to address this shortcoming of interactivity, Lewis relates the trigger-based type of interactivity that was in vogue in the early 1990s to rudimentary mental processes or, as he puts it, to "amoeba- or roach-like automata" (ibid., p. 11). In this sense, not only interactivity is at stake, it is rather the empowering of the image of the composer by the presence of a simple model of interaction. This intentionally (very) simple automaton promotes two fundamentally hierarchical notions that Lewis attempts to deconstruct. On the one hand, the composer as controller who would never relinquish control of

[9]However unfortunate this 'bang' name is, the computer itself makes one think back to the 1946 setting of the UNIVAC computer, in the military context of the Manhattan Project, for which it was used to get closer to the 'H' bomb. That is to say, even if 'bang' was named differently, the computer itself would be inevitably linked to this particularly *big* bang.

the music work, that is, the modern (eurological) image of the composer, and the old ghost train that comes with it: "The social, cultural, and gender isolation of the computer music fraternity (for that is what it is)" (ibid., p. 11). This image leaves improvisation, together with non-eurological thinking out of the scope of contemporary music research. On the other hand, the human operator as the higher (architectural) mind who would not allow for the nonhuman to become an operational agent beyond the instructions for which it was designed. In this sense, the simple-level automaton is a symbolic restraint representing the classical concept of the human, which allows a non-threatening relation between man and machine that can be considered functional, productive, and operative.

Nonhuman composers One is tempted to claim that the first of these images —the reified composer— is determined by the second —the reified human—, and that their relation is a matter of depth or inheritance. Thus, in order to redefine the composer one would have to redefine the human. In turn, this depth would be measured against that which is nonhuman, and by extension, that which is non-composer. We can understand, therefore, Lewis' narrative as the redefinition of composition itself by making the non-composer (e.g., what was eurologically considered the 'improviser' or the 'performer') resound back into composition, regrouping the concept 'composer,' but not as a whole, since now the extent of its terms have found places within a networked system. This is precisely what he does in *Voyager*. The composer, like the human, became regrouped in a certain hybridity where interactive computer music is *not entirely* driven by (human) input, because the system proposes an 'input' of its own.

Fractured Works

> [Lewis:] The composer therewith relinquishes some degree of low-level control over every single bloop and bleep in order to obtain more complex macrostructural behavior from the total musical system. The output of such entities might be influenced by input, but *not entirely* driven by it. [emphasis added] (ibid., p. 11)

It is precisely this 'not entirely,' as a negation of wholeness, which begins to question the basis upon which our general concept of the human is built, and by extension, what begins to announce the presence and the agency of everything that falls outside of its definition. It is the beginning of a breakage, a crack on the foundation of Xenakis' (old) armchair, from which the state of suspension of the concept of the music work can also be understood:

> [Lewis:] With this in mind, it becomes easier to see that Voyager is *not really a 'work'* in the modernist sense —heroic, visionary, unique. . . I choose to explore allegory and metatextuality, the programmatic, the depictive— and through embedded indeterminacy [pseudorandom generators], the contingent. Ultimately, the subject of Voyager is not technology or computers at all, but musicality itself. [emphasis added] (G. E. Lewis, 1999, p. 110)

What this fracture in the constitution of the concept of the work reveals is the hybrid nature of reality and virtuality. Understood traditionally or under the stipulations of first wave cyberneticians, the composer, being the real factor in the constitution of the (modern) image of the composer, is faced with the virtuality of the computer. Upon this encounter, the virtual comes as a form of threat to that image, and thus to the reality of the composer: the computer's virtuality would replace first the imaginary and then the real. However, as Lewis claims, interactivity between the composer and the computer allows both the virtual and the real, "virtuality and physicality," to engage in the production of a hybrid that "strengthens on a human scale." "Seen in this light," Lewis continues, " virtuality should enhance, not interfere, with communication between us" (ibid., p. 110). Considering the role of virtuality after embodied new media theory, the computer reveals to the human —composer, improviser, performer— the very condition of its own capability for virtuality, and thus redefines reality for the composer, and in turn, a new image of the composer emerges. This new image is no longer a threat, but rather a reflection of a new agency within music composition. In the case of *Voyager*, this virtuality is sonic, it comes as the "emotional transduction" that Lewis aims for with this computer system. Therefore, Lewis is right in claiming that *Voyager* is not 'really' a work, because it is virtuality as such: a virtual composer, improviser,

and performer, and in sum, a virtual listener.

Databasing Vessel Understood as a listener, *Voyager* engages not only with signal processing at the lower level, it engages with the resonant process of the relation to self. Furthermore, the computer is not only listening, it is *databasing*, because it is keeping record of the listened features, and in so doing, it becomes empowered with the database. This database of actions, however, is the sonic trace of the performance, which is what is most surprising of its agency, and what resounds most in time. Therefore, far from being 'really a work', but also far from Lewis' notions of narrative in the sense of "allegory and metatextuality, the programmatic, the depictive" (ibid., p. 110), I consider *Voyager* an unwork of music, because it questions the operativity of the music work. However, certain notions of productivity and cohesion are still present within Lewis' music and texts, and thus *Voyager* is still considered a 'work;' a destiny that somehow manages to persist within the practice of composition, which is why the metaphor of the unwork comes not without resistance. Nonetheless, and without a doubt, Lewis' claim for a "non-eurocentric computer music" (ibid., p. 107) can be a starting point to the conceptualization of the unwork.

7.5 The Severed Object Of Music

I would like to refer once again to Jean-Luc Nancy's concept of inoperativity (See 4.4), this time in relation to what I call the severed music object. This object is different from Pierre Schaeffer's music or sound object, which comes to represent material with which to work. Neither is it related to Vaggione's concept of an object, which comes from object-oriented programming, meaning every composable primitive, from the micro to the macro. In both these authors, the object is used to provide, though not without their author's intervention, a notion of *coherence* to the work.

Remnants of Listening The object I am referring to resides in memory, as what remains after the event of an exposure. It is inherently linked to the fractured way in which our own memory works, and it is impossible to define, since it has no beginning and no end. Its dimensionality includes both beginning and ending simultaneously. This object is the spectral evidence of a musical event or, better, of the happening that takes place in listening. In being evidence, it becomes a topic for analysis; it is forensic. In being fractured, this object is the evidence of a destruction. In being severed, and this is the central aspect that I would like to focus on, risking simultaneously the severing of the object, it becomes the evidence of a sacrifice. If it can be said that the music object is a severed object, then the question of its severing necessarily relates to the question of listening. Therefore, by listening —and, by this, I mean entering in resonance with resonance, exposing the self to that which returns to itself— I participate in this severing, because I choose what to listen in spite of being already deprived from that choice.

Sources and Sorcerers The severed object of music is what we as listeners grab from the stage, what we choose to rip from the sounding waves, and also what we cannot help but feeling so much a part of us before noticing it is happening. Severing is yet another way of thinking the aesthetic experience of listening, but it is not as passive as it seems. Severing empowers the listener; it is the tool of listening; the reversed stilus; the inverted mouse; the part of the human that necessarily is nonhuman. With it, we can make the world appear, but only as a fraction, because 'it' can never be *completely*. As Brian Kane writes of Nancy's work of art: "a work that refuses to create itself as a total work" (Gratton & Morin, 2015, p. 29). The severed object of music is always severed, but never in the same way, since there are as many severings as there are listeners, and as many listenings as there are moments. Composers have been traditionally considered a 'source' of this object or, better, the one at the door, the key keeper that has access to the door that opens up the flow of inspiration. The composer, but also the programmer with access to the source code, which unless it is opened, is hidden to the rest; and, unless you know the language, it is complete pseudo-

linguistic nonsense with weird punctuation marks, sometimes closer to poetry than it is to extreme formalism. For example, consider the following piece of code that can be read as a simple poem, but when run from a terminal would repeat in a computerized voice a famous line from Gertrude Stein's poem "Sacred Emily:"[10]

```
#!/bin/bash

# Palabritas que hacen cosas

while true
do
        for ever in rose is a
        do
                say $ever
                sleep $((RANDOM/10000))
        done
done
```

Listing 7.1: Little words that do things.

In this access to the source, the programmer and the composer are traditionally kept at a distance, as if their listening were of some other sort, engaging with the very essence of the source, drinking the water from the originary fountain, satisfying an originary thirst. Therefore, if this is the role of the composer and the programmer, if this is their relation to the source, then, they are the first to perform the severing. In the hierarchy of the consequent severings, they are at the top. Further, if they are the first severers, they are the first who perform the first listening. They are the listeners at the top of the mountain, next to the source of all fountains. On the way in and out of the world, the sorcerers of condensation.

Naming I would like to point out now that it is not my intention here to sever the head of the sorcerer because it is an illusion that does not allow me to do so. It is not my illusion, although I

[10]To run the code (linux or macos shell) simply copy the text onto a plain text file you can name `gert.sh` and run it with `sh gert.sh`. Note: it will never stop on its own.

have described how I interpret it, and it comes as a product of a reification of the composer, but also of the human as the one and only owner of the world. In being in resonance, listeners become the resonant world, that is, the self begins to resonate as space. In this sense, it is the world what is listened to, and it is a world that has no apparent origin. However, the composition —the written score, like the written code— propose their own origin —the composer, the programmer. Thus, they give an origin to the world by providing an answer (a name) to the question of creation: Who created this music? *this* composer. The answer, therefore, has a 'this' that comes in the form of the name of the composer. This name becomes attached to the flowing of the source. Therefore, the name of the composer is like a timbre stamp that is applied to the listening experience. Furthermore, the severing style now can be named. How many different names or anagrams would it take for Click Nilson's style to dilute or to arrive at the unclaimable work of art?: "This is why for some years I have experimented with releasing music under other people's names, so as to dilute their style, and under multiple versions of my own name, to cast doubt on any claim to the future" (Nilson, 2016). The name of the composer becomes a synecdoche of the source, directly naming part of the source. This applies, quite literally in some cases, to the name of the program and the name of the programmer (the 'max' in Max Mathews and the 'smith' in 'msp').[11]

Dynamics Furthermore, the activity of the sorcerer lends itself to its signature. In other words, the manner in which the composer defines the music from beginning to end becomes the shape of the music. We can understand 'shape' or 'form' as something that is at once behind and in front of the singularity of the listened music. It is behind because it is the activity of sound sources —speakers, musical instruments, or media in general— the movement of air pressure. It is in front because it filters the memory of the activity of sound sources. However, a composed shape and a singularity act together in the moment of listening. The question is, then, regarding the dynamics

[11] 'Max' is named after the 'father' of computer music Max Mathews, and MAX/MSP contains Miller Smith Puckettes's initials. Friendly gestures, most probably, but also pointers to originary sources, sources of inspiration, historical references that contextualize computer music software within broader social and environmental structures.

of this activity. Given that this activity happens during listening, what I address now is precisely how the shape of the music interacts with the listening experience. Interaction, here, refers to the shared activity that occurs 'inside' listening, and it happens 'inside' because of the severing that needed to occur prior —or immediately at— the resonant oscillation of air pressure. However, once this severing has occurred, and within its momentum, it is the internal dynamics that enter into play, and it is the shape of the music what begins to delineate the shape of the listened.

Masterwork The shape of the music is a force that produces a certain listening experience. Therefore, the internal dynamics are already prescribed. The singularity of the listened becomes (almost) one and the same with the shape of the music. 'Almost,' because it is not that the listened brings no resistance to this 'ideal' force. The singularity of the listened is resistance, it acts as resistance, but its force is not enough to resist the command of an excellent work. This is the very presence of the masterwork: the work of a master that requires a slave. This 'slave' is not the other music works that have not reached the necessary status of a master. Slavery exists among the outshunned singularities that have been muted by the very presence of a master. 'Almost,' in the hope that the work of this masterwork can be relativized and disarticulated; disentangled from the source of sources; brought down the stream to the place where singularities can resonate in endless forms of matter. The problem is now of a different sort. Even if resisting forces match those of the masterwork, then, like Derrida's paralysis of memory, we can encounter a paralysis of listening as such. This paralysis might (also) be what Szendy means by the inattentive listener that falls away. But, it is not a paralysis caused by distraction, it is a paralysis caused by the very effort that is needed to match the force of the masterwork. Thus, it is a paralysis that is directly called for from 'outside,' preventing any further listening. This is what is called for by the work of the masterwork: pure —and utterly ideal— silence.

Architecture of Obedience Therefore, within these dynamics of work, what results is a function of the predicates, it is an architecture of obedience that is written in the form of a music work, with the one and only aim which is for it to 'work.' Thus, the composer engaging with this dynamics of the work, becomes the architect of the listened, the creator of a listening of which he himself is the only chief. The sorcerer in charge of quenching a thirst that is only there because it instantiates with its creation. The question now is how can this dynamics be approached once we have recognized it. How can music composition continue? A composition not participating in this dynamics? A composition that is not a force? A composition that is not 'really' or 'entirely' a composition? A composition that does not impose its shape? A music work that is not a work but that still resonates within listening?

7.6 Anarchy And The Unwork

Inoperativity characterizes the aesthetic dimension in the severed music object of the composition that does not impose its own listening. In this sense, the practice of music composition can be understood in terms of Nancy's positive, active force of unworking. The condition of unworking in relation to works of art is exposed by a certain resistance present in the unwork of art. This resistance is a force of interruption and suspension that prevents the notion of a whole to reach completion.

Place in Common An unwork differs radically from the notion of an open work as is the case, for example, of Umberto Eco's famous formulation that "the work of art is a complete and closed form in its uniqueness as a balanced organic whole, while at the same time constituting an open product on account of its susceptibility to countless different interpretations…" (Eco, 2004). Instead of openness being located in the interpretation, the openness is inherent to the hybridity of its construction. The construction, in turn, is a result of the reticulated and fragmented state of

exposure between the human and the nonhuman.

Disintegrated Imperative I would like to analyze the inoperativity of the music in relation to the dynamics of the shape of the unwork and the singularity of the listened. The former, in being a disintegrated imperative —i.e., without the integrity that is required of the imperative for it to work as command and instruction—, cannot behave as a force in its own right. This is not to mean that it 'fails' as a force. At this point it would be useful to revise Kim Cascone's consideration of the aesthetics of failure (Cascone, 2000). In his analysis of the 'post-digital' culture of the late 1990s, Cascone identified electronic music outside academia as one related to the unintended uses of computer music software, also known as glitch art:

> It is from the 'failure' of digital technology that this new work has emerged: glitches, bugs, application errors, system crashes, clipping, aliasing, distortion, quantization noise, and even the noise floor of computer sound cards are the raw materials composers seek to incorporate into their music. (ibid., p. 13)

Blind Experimentation Within what he called the "cultural feedback loop in the circuit of the Internet" —where artists engage with download and upload of software tools and artworks— Cascone describes a 'modular' approach regarding music creation as being grounded in the use of (recorded) samples and later mixing (ibid., p. 17). His argument is that "electronica DJs typically view individual tracks as *pieces* that can be layered and mixed freely" [emphasis added] (ibid., p. 17). In atomizing this use of samples, glitch art descended to the micro-level, but precisely by this descent, it sacrificed the whole for the parts, that is, it became a case of extreme modularity that "affected the listening habits of electronica aficionados" (ibid., p. 17). Cascone's conclusion is to call for new tools "built with an educational bent in mind" (ibid., p. 17), bridging the gap between academic and non-academic electronic music and, therefore, illuminating glitch music "past its initial stage of blind experimentation" (ibid., p. 17).

Doctoring the Glitch It must be noted that Cascone's inclination towards bringing academic knowledge to the academy of the Internet refers not only to computer music software. Professors, generally of computer music, in several universities across the USA have been openly uploading class materials, patches, softwares, and many other highly useful technical information; not to mention the free and open online publishing of conference proceedings that have spawned in the last 20 years. Cascone's rendering of this educational turn can be understood with an authoritative and dated tilt on his end. Particularly, consider what he writes in relation to the form of glitch music, which is the last arguing moment before his claim for education: '...contemporary computer music has become fragmented, it is composed of stratified layers that intermingle and defer meaning until the listener takes an active role in the production of meaning" (ibid., p. 17). How are we to interpret Cascone's call for education? What is the center of this education: music technology, composition, or listening? If fragmentation, modularity, stratification, and deferred meaning are affecting listening habits, are these habits themselves what needs to be taught? Or is it the structure of the music that is under distress? I understand the ambiguities in his argument as coming out of the main premise of the text, that of extending the concept of failure from technology to the analysis of the artwork. Thus, in Cascone's view, the aesthetics of failure of the late 1990s are still failing to enter academia because they fail to achieve the same standards of formal cohesion that are required by the modern conception of the music work. Therefore, instead of finding an academic cure for blind experimentalism, I claim that failure is itself an unnecessary blindfold. Failure is only possible within the structure of teleology. That is to say, both failure and success are measured against the expected outcome of a project. In success, the aim is accomplished. In failure, there is nothing achieved since the task at hand is neither neglected nor omitted. Therefore, if our project is a music work and we consider it outside the arrow of teleology, then failure has no place in it.

Spectral Traces The unwork cannot behave like a force, but it can be considered the spectral traces of a force. In this sense, if there is an illusion of a force, it must appear as wreckage; an after dream; a mirror that shows us a skin of the past; the ruins of an empire; or the humidity creeping through the cracks of an old house. However, and this is a big however, these allusions to vessels, to the psyche, to architecture, and to the presence of the past altogether, must be addressed with the same strength as one would address a phantom. The unwork makes us feel the uncanny presence of the past in the now, of the overpowering ghost that brings with it the archontic, in the shape of our own selves that has been revealed to us as 'not us,' but as yet again us. This is the moment that the unwork carries with it the most crucial aspect of all: it has nothing to give. It expresses nothing. And this is when listening finds us without anything to hold on to but our resonance. Our very own listening to ourselves listening. The moment where we realize it is our own self that is returning to us. This is our resistance.

Macroforma The resonance of a return. This is why the unwork depends so extremely on its very state of fragility: even the softest sounds have this self-referencial power. The moment this fragility is forgotten is when composers, performers, improvisors, programmers —humans and nonhuman listeners, in the broadest sense possible— enable an operative `macro` that has a political agency in the shaping of singularities. In order to provide some insight into the difficulties that arise from this conceptualization of the unwork, I would like to bring Vaggione again. When he writes of the shaping of singularities, he refers to the arbitrariness of the composer. However, he intentionally maintains formal coherence by extending the singularity of a grain (conceptually) to the singularity of a work. Therefore, in expanding this singularity he is ultimately arriving at a very unique and delimited shape that is the work. The contradiction I see here is that, in an attempt to propose a bottom-up approach in which, like Lewis' work, local actions percolate up to global behavior, Vaggione grants his work with an inevitable global behavior that is extremely operative: Vaggione himself. Without a doubt Vaggione (self) *is* singular, and the value of his music is not put into

question. I bring this as an example of the name of the composer and its impression on the music. In this case, the singularity of the composer impresses its own shape, style, and trace on the music. The problem is that the work now engages with its own operativity, with its integrity, and begins to dictate the shape of its own listening.

Overfitting Anarchy is a paradoxically productive force. As I have outlined before, databasing and composition bring forth their relation to the archive, and by doing so, they are bound to the origin and the rule. Like the name of the composer which is written on the shape of the music, the database has too the potential of becoming a source. Databasing becomes an activity of this source, and thus embeds the databaser with a specter of authority. Claiming, therefore, that composition can be identified with, or understood as, databasing means translating the 'archic' not only to the performativity of composition, but also to the product of composing, to the composer and the composed; to the shape of the music and to the singularity of the listened. An unwork, therefore, would be necessarily an an-archic work. It is still a work, however, in the sense that it demands from the composer, from the databaser, and from every node in the scope of its network, an incessant operativity. That is to say, the 'un' of the unwork does not come from inactivity, from passivity, from an escape of any form of action. Quite the contrary, it is a result of the constant impression of the work, the accumulated efforts towards the 'un' of the thing; an extreme operativity that goes beyond the threshold of its own making so that it reaches a point of inflexion, a bent, an overflow. There is a point in statistics where learning algorithms, given a dataset, tend to adapt themselves too closely to the dataset, thus failing to render future predictions reliably. This is known as overfitting. Despite its uselessness (or, better, because of it) I believe this to be a suitable metaphor for the thinking about the unwork: precisely by overworking the work, one can find some insight into the 'un,' and thus, one can begin to approach the anarchic in music composition. However, this approach comes not without its warnings, since it means at once, to eradicate the archic with the 'an', which means to introduce a bug in the Oedipal loop that could result in unheard-of musical

behaviors.

7.7 [Wip] Work In Progress

```
// code for the "working" pd class.
// it does nothing.

#include <stdio.h>
#include "m_pd.h"

t_class *working_class;

typedef struct working {
        t_object  *x_obj;
        t_symbol *work
        union {
                t_symbol *product;
                t_symbol  *music_piece;
                t_symbol  *music_work;
                t_symbol  *opera;
        } music_work;
        t_symbol  *something_done;
        t_float  *physical_labor, *skill;
        t_atom  *the_work_of_an_author, *oeuvre;
        t_symbol *the_operativity_of_the_composer;
        t_atom  *matrix_operations;
        t_symbol *operetta, *opera_prima, *obra, *open_work;
        t_symbol *a_work_of_art;
        t_symbol *artistic_creation, *techne;
        t_float  *fullTime, *partTime;
        t_symbol *clockwork, *officiate, *office, *act;
        t_symbol *produce, *make_it_work;
        t_float  *magic_work, *work_of_angels;
        t_symbol *blueCollar, *whiteCollar, *slavework, *masterwork;
        t_symbol *Work_as_in_the_application_of_forces;
        //V:"But applied to whom?"
        t_symbol *working_a_field;
        t_symbol *the_internal_workings_of_structures;
        t_symbol *work_in_an_app, *worked_out;
        t_symbol *work_your_hat_off, *workflow, *workspace;
        t_symbol *working_for_food, *hardworking, *labour, *giving_birth;
        t_symbol *all_that_is_remunerated_after_efforts_have_been_given;
        t_symbol *achieve_a_goal, *your_task, *to_work_to_live;
        t_symbol *to_have_a_working_body, *functioning;
        t_symbol *operative, *working_like_a_bee;
        union {
                t_symbol *like_a_bee;
```

```
                t_symbol *like_a_member_of_the_hive;
                t_symbol *like_an_ant;
                t_symbol *like_a_worker;
                t_symbol *like_a_coworker;
                t_atom *organized_labour;
        } workers_union;
        char   work["for","to","after","by"];
        unsigned char *hours;
        t_symbol *working_as_an_extension_of_truth_as_well_as_lies;
        t_symbol *out_of_work, *at_work, *work_in_progress;
        t_symbol *working_for_the_man, *freelancing, *working_under_the_table;
        t_symbol *working_past_a_deadline, *working_in_pairs;
        t_symbol *teamwork, *collaborate, *co-operate;
        t_symbol *paperwork, *networking, *prototyping, *worked-up;
        char  *work_the_crowd, *work_the_system;
        t_symbol *work_a_miracle, *work_your_workers;
        t_symbol *social_worker;
        t_float  *a_ship_works_in_a_heavy_sea, *work_the_levers;
        t_float  *work_for_Facebook, *future_work, *framework;
} t_working;
```

Listing 7.2: Pure Data working class

Afterword

In this dissertation, I embarked on an adventure throughout a sonic history of databases. Database music has been sounding in computer music, in sonification practices, and in Music Information Retrieval (MIR), in a similar way that has been shining in new media theory in the past decades. This similarity is not only interdisciplinary, and not only at the level of computers, algorithms, and data structures; it is a similarity of a different nature, one that we can call spectral, and that places us and our experience of art at an intersection. It is interdisciplinary because it exists along the edges of our practices, as a shared practice that we have inherited, one that continues to progress as we understand and reconfigure its performativity. It is a technological reflection of ourselves that changes us just as well as we reflect it back and change it. It is spectral because it reconfigures our own notions of what is human and what is nonhuman.

Since computers have changed communications and science in extreme ways in recent decades, our aesthetic experiences have also changed. Music made with databases cannot simply be considered music. If there is something we can take from new media theory is that human experience is mediated by technology. With this in mind, database music has been used as an experiment throughout this dissertation. On one hand, I tested how much of the database we can find in art, in computers, and in sound. On the other, I delved into mediation with terms such as listening, memory, and performance. Both of these experiments found a way to collide towards the end, with a musical approach that focused on how what once was cutting-edge now has become a common practice; not to disregard the latter for the former, but to point to what is in common and how it sounds. This project takes on a granularity that I have tried to hold back, only to find that the more I was finding, the more I needed to write. For this reason, I can suggest that the topic of database music is still largely unexplored. Nevertheless, I covered some of the central questions that I have found in the literature, and tried to pose some of my own. What is the role of the database in art? How has this role been contextualized? Much of the literature focuses on Internet art, digital art, and also on visual and virtual reality, topics of great interest that had to be left out of this text. Instead, I emphasized the structures underlying databases, their

histories, and how these have reconfigured our own sense of corporeality in art. In this sense, I have explored how databases have been present and evolved in music practices, and how these practices have undergone significant changes in return. Namely, these changes appear at the level of music software development as well as in the resulting artworks and in the different approaches to music composition.

Due to the variety of shapes that database music has taken over the years, I focused on what I found to be central questions with which to approach a framework for discussing aesthetics of database music. What is it about database music that we find so fascinating? I ask this question and try to answer it, but I have warned already about my condition of composer in the introduction. A 'condition' that began as an obsession for musical acoustic instruments, and for digital instruments, but that has never separated from listening, imagining, and performing sound. In this way, I evaluated how these three terms relate to database music. Listening brought the discussion of databases to resonance, and to a resonance that redefines our notion of self, as well as our notion of community. Imagination brought the database and its relation to memory, which reconfigures how we dream database music, how we remember it, and how we document it. Performance brought the database to a stage, which reconfigures the surfaces upon which we transform at every moment, in every act. With this threefold approach, I suspend a framework for an aesthetic discussion of database music that I contextualize within new media theory as well as music composition; and that I, at once, begin to delineate, but leave incomplete and untethered to the limits of its constitution. Perhaps the reader might find new ways of approaching this discussion, and I hope that by then, the inexhaustibility of the topic would raise new questions about the aesthetics of database music. The final step of this adventure takes on the topic of work within database music, contextualizing musical work in light of this strange hybrid database music. How has the database changed music composition? and, How can we think of database music composition? I have begun answering these questions, but I have not arrived to any conclusions, so this text might perhaps be "an attempt to incite... a provocation before the question" but —and I cannot stress this enough—: this

is only a incitement in terms of acoustic laws, an aesthetic provocation, and a question that perhaps remains unasked.

Appendix

Chapter 8

Multimodal Database: mmdb

8.1 Overview

`mmdb`[1] is a multimodal database system geared towards live querying of image an audio. It consists of a series of bash and python scripts that surround and aid the performance of Pure Data patches. A multimodal database combines two sensing modes. In this sense, the camera sensor and the microphone.

The system enables you to load a folder with images with various formats and sizes, analyze them, and output a database describing the images with some useful keywords (descriptors). The images can be either taken and collected by you, obtained from the web, or generated by some other means. The analysis is done after normalizing these images to the same size and format in a pre-processing step.

After the analysis is done, the database obtained is divided into two types. The first type is a very small text file with only a handful of values that describe a few things of the image. This file is useful to sort all images based on some or all of these values. The second type is a semi-structured JavaScript Object Notation (JSON) file that includes a lot of data referring to each

[1] This program is available at: https://github.com/fdch/mmdb

image. This second file is then used for a set of purposes. On the one hand, we can use this database to perform queries based on those values and obtain desired images. For example, we can ask for bright images, or images with faces or bodies, or images with lots of blobs, etc. On the other, from this database we obtain a set of color words (English color names) of the most present colors on each image. These color words become the link between image and audio in a process that goes as follows. First, we use these color words to query related nouns to those colors using an online database called Datamuse.com. Then, from this query, we obtain another database that has all of these colors and nouns. Finally, this intermediary database that has only text is used to query Freesound.org, to match and download sounds related to those words. Once we have our folder with downloaded audio files from Freesound, we concatenate all of these sounds in sound files named with their respective color words.

Now we can use our audio and image databases to perform a simultaneous query to both, and display this live as an audio/visual stream. The live query is made with a matching matrix that equates certain image descriptors with some audio descriptors. For example, images with faces and bodies will match with audios with pitches on them, and images with many blobs will match with noisier sounds.

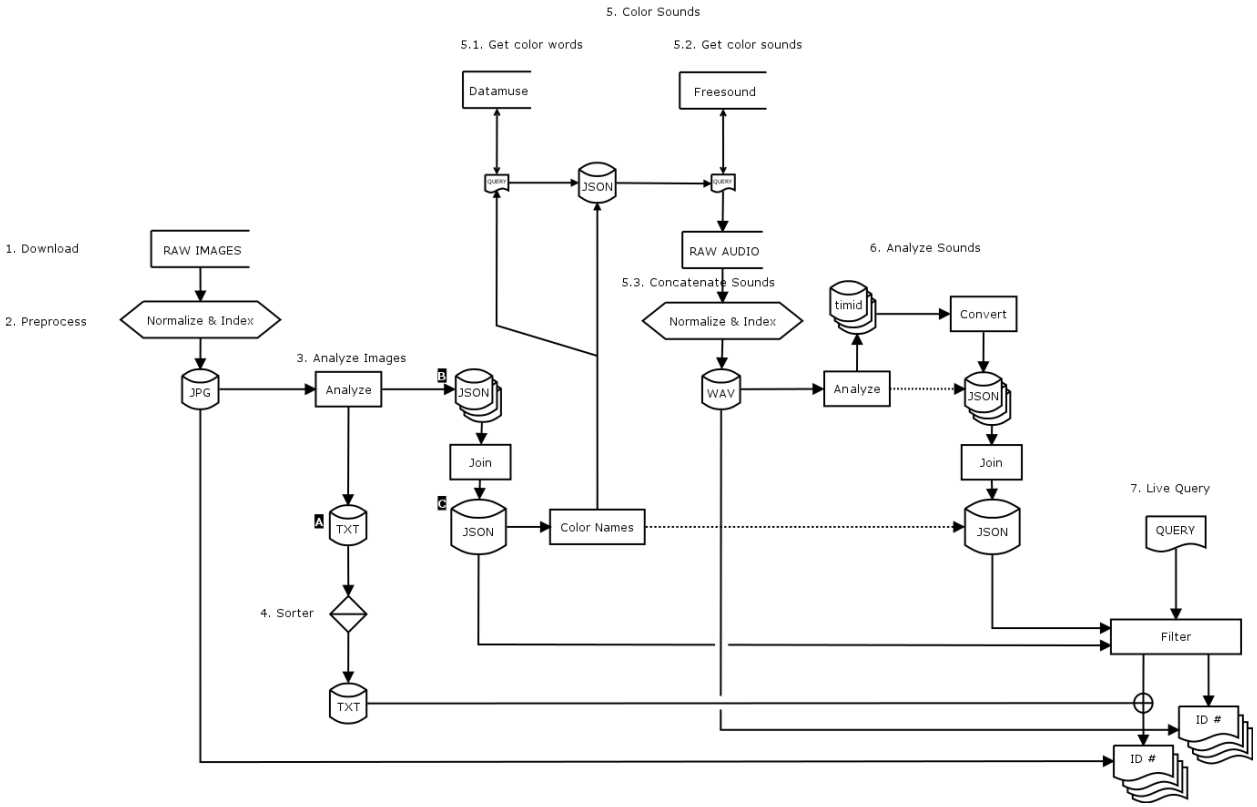


Figure 8.1: Diagram representing each step
Data flow needed to obtain a live querying system for images and audio based on feature equivalencies.

8.2 Steps

The program is broken down into independent modules that enable you to perform a sequence of tasks that are needed for the live querying system to work. While these steps may be done systematically with an automated script, I chose to keep each module separated by design, so as to provide fine tuning on each part of the process. In this sense, you can better adjust parameters in each step of the way according to the particular datasets that you would use. Most of these steps are run from the command line and they take several arguments. See their help for further information on how to use them. Nonetheless, some of these programs are Pure Data patches run on batch mode, meaning that you can open the respective patches to edit or to change internal parameters according to your needs.

These are the main steps:

1. Download
2. Preprocess
3. Analyze Images
4. Sorter
5. Color Sounds
6. Analyze Sounds
7. Live Query

8.2.1 Download

The way this system works is that you must start with a folder with images that you then preprocess and analyze for useful descriptions. Later versions of this program will enable you to start with a folder with audio instead. For now, you need to first download an image dataset of raw, original files into the `raw` directory. This dataset can be of any type, format, and sizes. In this sense, this directory can contain various images from different sources, either from the web, or made by you.

8.2.2 Preprocess

The program `preprocess.sh` preprocesses a directory holding an image/video dataset and outputs the following into three directories:

- <img> : resized images to defined `WIDTH`, `FORMAT`, base name, and `EXT`ension
- <vid> : extracted frames resized to defined `WIDTH`, base name, and `EXT`ension
- <aud> : extracted audio to defined format in `AUDEXT`

Additionally, it outputs a Comma Separated Values (CSV) file holding original files and converted files for later use, since filenames would otherwise be lost. This file is useful for making high quality renderings of visual streams with the original images.

This step needs ffmpeg and sips

8.2.3 Analyze Images

The program `analyze.sh` contains a single function `analyze()` which takes three arguments:

1. files: like multiimage 'open' message: path/to/file/base-*.extension
2. target: prefix to name analysis files
3. roi-flag 1/0 to set roi defined in roi text

The core of this program is the Pure Data file `analyze.pd`, which is run in batch mode, analyzing the specified images, and returning one JSON file per analyzed image. Optionally, you can define a Region of Interest (ROI) by opening the `roi.pd` patch, then adjusting your desired ROI, and finally setting the ROI flag to `1` before running the command line program.

The output of this program is in two types: an entry file **A** (See Table 8.1) and a data file **B** (See Table 8.2)

Field	Descriptor	Brief explanation
1	brightness	variance of the image histogram
2	bodies	number of bodies found (haarcascades)
3	faces	number of faces found (haarcascades)
4	cvblobs	number of cvblobs found
5	lines	number of hough lines found
6	circles	number of hough circles found
7	keypoints	number of keypoints (corners) found

Table 8.1: Description of the 'entries' text file. This file is a plain text file that contains one entry (rows) per image file holding the data described above, one value per field (columns)

8.2.4 Sorter

The program `sorter.sh` has only one function named `sort_it()`, which takes a variable number of arguments. The arguments are the field (columns) of a given text file with which to sort the text file. Thus, it enables you to sort the entries file **A** (`<entries_file>`) based on any given field, as well as with pairs or sets of fields. It outputs the sorted files into a file with the same name of the original, appended with the `-sorted.txt` flag to the name. Each line of the `-sorted.txt` file contains the sorted indices of each image filename. This is useful to load within Pure Data as a text file for use with the live querying system, as an alternative way to generate image sequences besides the query results.

8.2.5 Color Sounds

The purpose of this section is to obtain a sound database based on the analyzed images. It uses the color clusters for this purpose. Since it is a rather large process, this section is broken down in three steps:

Get color words

The `colors.py` script places first all data objects **B** inside an array of objects in one JSON object **C** (`<image_data_file>`). Thus, it concatenates a series of JSON files into one. Then,

Descriptor	Brief explanation	Structure
`mean_col`	the array of color clusters of the image	`pct` (percentage over the total image), `b` (blue), `g` (green), `r` (red)
`histo`	the histogram of the image	array of fixed size (64)
`bodies`	the sequence of blob information pertaining to found bodies on screen	`x`, `y`, `r` (radius), `id`
`faces`	the sequence of blob information pertaining to found faces on screen	(same as `bodies`)
`blobcount`	the number of blobs detected	one value
`blobpoints`	the total number of points accross all blobs	one value
`cvblobs`	the array of blobs recognized in the image	`Ycentroid`, `Xcentroid`, `length`, `points`, `corners` (4 x-y pairs), `area`, `angle`, `Ysize`, `Xsize`, `Ycenter`, `Xcenter`, `id`,
`lines`	the array of lines found on the image	`y1`, `x1`, `y2`, `x2`, `d` (length of line), `id`
`circles`	the array of circles found on the image	`r` (radius), `y`, `x`, `id`
`keypoints`	the array of keypoints (corners) found on the image	`y`, `x`, `id`
`id`	the index number of the image file name	one value

Table 8.2: Description of the JSON data files. These files are JSON files containing one object per image file holding the data mentioned above.

Key	Value
`name`	English name of the color, e.g. 'blue'
`idlist`	all the image ids that have that color
`words`	nouns related to such color after querying datamuse.com, e.g. 'sky,' 'eyes,' etc.

Table 8.3: Description of the 'colorwords' JSON file holding one object accessed with the key `data` which has an array of objects as described above, one object per color name.

this script gets English names of the clustered colors in the JSON data base **C**, and outputs a file (<color_words_file>) containing one entry per unique color. The structure is like this: name, idlist, and words (See Table 8.2.5).

Get color sounds

The <color_words_file> file is then used to query Freesound and download sounds related to all words and names using the python script: fs_download.py. NOTE: Some colors may not result in words that have a related sound to them.

Concatenate sounds

The script concat_sounds.py concatenates all downloaded sounds sounds into single files named after their respective colors. This script runs ffprobe to ignore files that might not be audio, or that might be malformed. It then runs ffmpeg to concatenate all the audio related to a color name into a file named with that same color name.

8.2.6 Analyze Sounds

The script analyze_sounds.sh runs the analyze_sounds.pd patch in batch mode. It analyzes sounds in a source directory, and places all *.timid files in a target directory. It takes two arguments: a source and a target directory. Optionally, you can analyze only one file by passing an index into the source directory with a 3rd argument. By default, the analysis is outputted both in *.timid and in *.json (using a custom converter found in timid2json.py), and it concatenates all JSON files into one database (<audio_data_file>)

Instance Structure

The first nine features are single-valued, so one float each. The last two features default to 50 values each, representing the bins of the bark scale with a filterbank spaced at 0.5. You can edit this and other parameters on the parameters file (open `analyze_sounds.pd` to do this). The instance length would change accordingly. The output analysis file is one per each audio file, with the following instance structure:[2]

1. barkSpecSlope
2. barkSpecKurtosis
3. barkSpecSkewness
4. barkSpecBrightness
5. barkSpecFlatness
6. barkSpecRolloff
7. barkSpecCentroid
8. barkSpecSpread
9. barkSpecIrregularity
10. bfcc (bark frequency cepstral coefficients)
11. barkSpec (used for all of the above, internal window size is 512)

The default analysis window size is 4096, so in one second of file at 44100, you will have around 10 instances, which is ok for many purposes, but you can change this. On the one hand, you can specify overlaps (default 1, no overlap). On the other, you can define an analysis average factor **f** (default 8). This factor is used to average several smaller sized analysis into one. To do this, we simply take the mean of **f** consecutive analysis frames within the larger analysis window size.

[2] All of these features come from **timbreID**, so their longer explanation I leave to their respective helpfiles.

8.2.7 Live Query

This enables you to perform live queries to both images and audio simultaneously, using the same query parameters and a matching matrix (See Table 8.2.7).

Instructions

To open this patch, run the following on four separate terminals:

1. `bash audio` (for sound playback)
2. `bash display` (for image display)
3. `python live_query.py <arguments>` (for the live database to load)
4. `pd live_query.pd` (for the live querying system interface)

The arguments for the `live_query.py` script are:

1. `<entries_file>`
2. `<image_data_file>`
3. `<audio_data_file>`
4. `<color_words_file>`
5. `<port number>` [default: `5011`]
6. `<host name>` [default: `localhost`]

8.3 Extra

8.3.1 Reader / Visualizer

The `reader.pd` patch can be used to visualize the JSON data files **B**. This patch is useful to edit your parameters for analysis depending on your particular image dataset. You can isolate each visual feature interactively. For it to load, you need to have already analyzed the images at least once.

Image Feature	Audio Feature Equivalency
thres_{R,G,B}	audio database
thres_C	audio database
{bodies, faces}	Kurtosis
{bodies, faces}[size]	Skewness
brightness	Slope
smoothness	grain size (for concatenation)
cutness	grain size (for concatenation)
blobiness	Brightness, Flatness, Rolloff
skewness	grain location (for spatialization)
boundedness	Centroid, Spread
kontrastedness	Irregularity

Table 8.4: Matching Matrix: image and sound descriptor equivalencies.

8.3.2 Image Query (non-live)

The `query.pd` patch is a gui for `query.py` and it is a non-live version of the live query patch. It works in a very similar way, therefore, it can be used to perform a query to the JSON database **C** to get indices, based on multiple descriptors (color, brightness, smoothness, blobiness, etc.), and to visualize the queries for live editing with the `display` program. Both input query and its results are stored on JSON files for later use.

8.4 Dependencies

8.4.1 Externals

I have not included binaries within the repository holding **mmdb**, but you can download the following externals available via `deken`: **Gem**, **pix_opencv**, **purest_json**, **ggee**, **timbreID**, **zexy/repack**. Additionally, you can get **fd_lib**[3] for **iterate** and **counter**.

[3] https://github.com/fdch/fd_lib

8.4.2 Abstractions

In the `bin/lib` directory there are some abstractions made for this repo (prepended with a _). I also have included these together with some other abstractions as well in the `pdbin` directory that are taken from **`fd_lib`** and other places. `pdbin` might not be necessary if you have already installed all the external libraries mentioned above. NOTE: the `pdbin` directory is not necessary to load the patches, it is just placed there for convenience. Just declare it with **`declare -path ../pdbin`** if you need to use it.

Glossary

Access Microsoft Access is a database management system from Microsoft that combines the relational Microsoft Jet Database Engine with a graphical user interface and software-development tools. See also: https://products.office.com/en/access. 46

AFLOWLIB A globally available database of 2,118,033 material compounds with over 281,698,389 calculated properties See also: http://aflowlib.org/. 59

Apache Lauded among the most successful influencers in Open Source, The Apache Software Foundation (ASF)'s commitment to collaborative development has long served as a model for producing consistently high quality software that advances the future of open development.. 46

API In computer programming, an application programming interface is a set of subroutine definitions, communication protocols, and tools for building software. See also: https://en.wikipedia.org/wiki/Application_programming_interface. 84

ArangoDB ArangoDB is a native multi-model database system developed by ArangoDB Inc. The database system supports three data models with one database core and a unified query language AQL. The query language is declarative and allows the combination of different data access patterns in a single query. ArangoDB is a NoSQL database system but AQL is similar in many ways to SQL.. 46

ASE SAP ASE, originally known as Sybase SQL Server, and also commonly known as Sybase DB or Sybase ASE, is a relational model database server developed by Sybase Corporation, which later became part of SAP AG. ASE is predominantly used on the Unix platform, but is also available for Microsoft Windows. See also: https://en.wikipedia.org/wiki/Adaptive_Server_Enterprise. 46

AudioGuide AudioGuide is a program for concatenative synthesis that I'm currently developing with Norbert Schnell, Philippe Esling and Diemo Schwarz. Work began in 2010 at IRCAM when I was composer in residence for musical research. Written in python, it analyzes databases of sound segments and arranges them to follow a target sound according to audio descriptors. The program outputs soundfile event lists that can be either synthesized (in csound or Max MSP/Pure Data) or translated into symbolic musical notation.. 79

AutoCAD AutoCAD is a commercial computer-aided design and drafting software application. Developed and marketed by Autodesk, AutoCAD was first released in December 1982 as a desktop app running on microcomputers with internal graphics controllers. See also: https://en.wikipedia.org/wiki/AutoCAD. 142

BOOST Boost is a set of libraries for the C++ programming language that provide support for tasks and structures such as linear algebra, pseudorandom number generation, multithreading, image processing, regular expressions, and unit testing. See also: https://www.boost.org. 46

CAAC Unlike software for sequencing, mixing, or notation, these systems are often diverse and innovative, breaking with traditional musical paradigms of meter, part, or score. These systems expand compositional resources, and offer diverse models of compositional design. See also: . 38

CAMP A general purpose composition and performance software environment originally based upon William Buxton's SSSP. See also: . 36, 64

Cassandra The Apache Cassandra database is the right choice when you need scalability and high availability without compromising performance. Linear scalability and proven fault-tolerance on commodity hardware or cloud infrastructure make it the perfect platform for mission-critical data. Cassandra's support for replicating across multiple datacenters is best-in-class, providing lower latency for your users and the peace of mind of knowing that you can survive regional outages.. 46

CataRT The concatenative real-time sound synthesis system CataRT plays grains from a large corpus of segmented and descriptor-analysed sounds according to proximity to a target position in the descriptor space. This can be seen as a content-based extension to granular synthesis providing direct access to specific sound characteristics. See also: http://imtr.ircam.fr/imtr/CataRT. 79

CC An American non-profit organization devoted to expanding the range of creative works available for others to build upon legally and to share. See also: https://en.wikipedia.org/wiki/Creative_Commons. 87

CCRMA A multi-disciplinary facility where composers and researchers work together using computer-based technology both as an artistic medium and as a research tool. See also: https://ccrma.stanford.edu/. 68, 81

CDIP An extensive network for monitoring waves and beaches along the coastlines of the United States. Since its inception in 1975, the program has produced a vast database of publicly-accessible environmental data for use by coastal engineers and planners, scientists, mariners, and marine enthusiasts. See also: https://cdip.ucsd.edu/. 55

CERL A research center based in the University of Illinois See also: https://en.wikipedia.org/wiki/PLATO_(computer_system). 71

CMAM The Centre for Arab and Mediterranean Music — Centre des Musiques Arabes et Mediterraneennes (CMAM) — is an institution operating under the authority of the Ministry of Cultural Affairs. It was established on 20 December 1991and its statutes were enacted in October 1994. See also: http://cmam.tn/. 49

Cmix A computer music software designed and developed by Paul Lansky. Belongs to the Music N family, although it was designed for a more specific context of concrete music.. 75

CNMAT A multidisciplinary research center within University of California, Berkeley Department of Music. The Center's goal is to provide a common ground where music, cognitive science, computer science, and other disciplines meet to investigate, invent, and implement creative tools for composition, performance, and research. It was founded in the 1980s by composer Richard Felciano. See also: https://en.wikipedia.org/wiki/Center_for_New_Music_and_Audio_Technologies. 85

COBOL A compiled English-like computer programming language designed for business use. See also: https://en.wikipedia.org/wiki/COBOL. 39

CODASYL A consortium formed in 1959 to guide the development of a standard programming language that could be used on many computers. This effort led to the development of the programming language COBOL and other technical standards. See also: https://en.wikipedia.org/wiki/CODASYL. 39

CouchDB Apache CouchDB is open-source database software that focuses on ease of use and having a scalable architecture. See also: http://couchdb.apache.org/. 34, 46

CPU The electronic circuitry within a computer that carries out the instructions of a computer program by performing the basic arithmetic, logic, controlling, and input/output (I/O) operations specified by the instructions. See also: https://en.wikipedia.org/wiki/Central_processing_unit. 18, 57, 76

CSV A delimited text file that uses a comma to separate values. See also: https://en.wikipedia.org/wiki/Comma-separated_values. 59, 180

CUIDADO A new chain of applications through the use of audio/music content descriptors, in the spirit of the MPEG-7 standard See also: . 79

DARMS The DARMS project started in 1963 by Stefan Bauer-Mengelberg and it is one of the first programming languages for music engraving See also: . 67

DBMS A computer program (or more typically, a suite of them) designed to manage a database, a large set of structured data, and run operations on the data requested by numerous users. Typical examples of DBMS use include accounting, human resources and customer support systems. See also: https://en.wikipedia.org/wiki/Category:Database_management_systems. 31, 36, 85

DDL A data definition or data description language (DDL) is a syntax similar to a computer programming language for defining data structures, especially database schemas. See also: https://en.wikipedia.org/wiki/Data_definition_language. 37

DJ A person who plays existing recorded music for a live audience. Most common types of DJs include radio DJ, club DJ who performs at a nightclub or music festival and turntablist who uses record players, usually turntables, to manipulate sounds on phonograph records. See also: https://en.wikipedia.org/wiki/Disc_jockey. 80

DML A data manipulation language is a computer programming language used for adding, deleting, and modifying data in a database. A DML is often a sublanguage of a broader database language such as SQL, with the DML comprising some of the operators in the language See also: https://en.wikipedia.org/wiki/Data_manipulation_language. 37

DOM A cross-platform and language-independent application programming interface that treats an HTML, XHTML, or XML document as a tree structure wherein each node is an object representing a part of the document See also: https://en.wikipedia.org/wiki/Document_Object_Model. 40

DSL A computer language specialized to a particular application domain. This is in contrast to a general-purpose language, which is broadly applicable across domains. See also: https://en.wikipedia.org/wiki/Domain-specific_language. 59

EDM A broad range of percussive electronic music genres made largely for nightclubs, raves and festivals See also: https://en.wikipedia.org/wiki/Electronic_dance_music. 87

EMI "Experiments in Music Intelligence (1984) was developed in order to create an interactive tool for composing.... By applying an augmented transition network parser and an object oriented approach, intervals, through inheritance and message passing, have both local and global impact (non-linear composition)" See also: http://artsites.ucsc.edu/faculty/Cope/experiments.htm. 137

EsAC Essen Associative Code (EsAC) was developed for one-part music, especially for European folksong databases. The code itself was inspired by the Chinese notation JIANPU and consists of cyphers (meaning pitches related to the declared tonic of the mode) and underlines and dots (standing for rhythmic durations). In addition to the code, several programs have been written for PC to analyze, listen, transpose and represent melodies. Conversions are possible into TEX, PCX, MIDI and other formats. See also: http://www.esac-data.org/. 50

Essentia Open-source library and tools for audio and music analysis, description and synthesis. 50

FDN "Structures well suited for artificial reverberation. These structures are characterized by a set of delay lines connected in a feedback loop through a 'feedback matrix"' See also: https://ccrma.stanford.edu/~jos/cfdn/Feedback_Delay_Networks.html. 97

FM In telecommunications and signal processing, frequency modulation is the encoding of information in a carrier wave by varying the instantaneous frequency of the wave. See also: https://en.wikipedia.org/wiki/Frequency_modulation. 57

Freesound Freesound is a collaborative database of Creative Commons Licensed sounds. Browse, download and share sounds.. 49, 50, 79, 87, 183

GB A unit of information equal to one thousand million (109) or, strictly, 230 bytes. See also: https://en.wikipedia.org/wiki/Gigabyte. 55

GDIF A format for storing, retrieving and sharing information about music-related gestures. See also: . 86

GEM GEM stands for Graphics Environment for Multimedia and is an external (plugin) for the computer-music software Pure Data. See also: http://gem.iem.at/. 81

GPU A specialized electronic circuit designed to rapidly manipulate and alter memory to accelerate the creation of images in a frame buffer intended for output to a display device. See also: https://en.wikipedia.org/wiki/Graphics_processing_unit. 57

GSR GSR is another name for Electrodermal Activity (EDA) is the property of the human body that causes continuous variation in the electrical characteristics of the skin. See also: https://en.wikipedia.org/wiki/Electrodermal_activity. 83

GTTM A theory of music conceived by American composer and music theorist Fred Lerdahl and American linguist Ray Jackendoff and presented in the 1983 book of the same title. See also: https://en.wikipedia.org/wiki/Generative_theory_of_tonal_music. 51

GUI The graphical user interface is a form of user interface that allows users to interact with electronic devices through graphical icons and visual indicators such as secondary notation, instead of text-based user interfaces, typed command labels or text navigation. See also: https://en.wikipedia.org/wiki/Graphical_user_interface. 44, 61

GUIDO The GUIDO Music Notation Format is a formal language for score level music representation. It is a plain-text, i.e. readable and platform independent format capable of representing all information contained in conventional musical scores. See also: http://guidolib.sourceforge.net/GUIDO/. 66

HCI A field that researches the design and use of computer technology, focused on the interfaces between people (users) and computers. See also: https://en.wikipedia.org/wiki/Human-computer_interaction. 61

HMDB A comprehensive, high-quality, freely accessible, online database of small molecule metabolites found in the human body. Created by the Human Metabolome Project funded by Genome Canada. See also: https://en.wikipedia.org/wiki/Human_Metabolome_Database. 56

HMSL HMSL is a programming language for experimental music composition and performance. It was popular between 1986 to 1996. HMSL is an object oriented set of extensions to the Forth language for the Amiga and the Macintosh. See also: http://www.softsynth.com/hmsl/. 72

HP An American multinational information technology company headquartered in Palo Alto, California. See also: https://en.wikipedia.org/wiki/Hewlett-Packard. 46

HTTP An application protocol for distributed, collaborative, hypermedia information systems. See also: https://en.wikipedia.org/wiki/Hypertext_Transfer_Protocol. 56

IBM An American multinational information technology company headquartered in Armonk, New York, with operations in over 170 countries. See also: https://en.wikipedia.org/wiki/IBM. 38

IBM-7090 The IBM 7090 is a second-generation transistorized version of the earlier IBM 709 vacuum tube mainframe computer that was designed for 'large-scale scientific and technological applications.' The first 7090 installation was in November 1959.. 148, 149

ICMC A yearly international conference for computer music researchers and composers. It is the annual conference of the International Computer Music Association (ICMA). See also: https://en.wikipedia.org/wiki/International_Computer_Music_Conference. 56, 67

IDMS A network model database management system for mainframes See also: https://en.wikipedia.org/wiki/IDMS. 46

IDS An early network database management system largely used by industry, known for its high performance. IDS became the basis for the CODASYL Data Base Task Group standards. See also: https://en.wikipedia.org/wiki/Integrated_Data_Store. 40

IEM A multidisciplinary research center within the University of Music and Performing Arts, Graz, (Austria). See also: https://en.wikipedia.org/wiki/Institute_of_Electronic_Music_and_Acoustics. 54

IMS The first database model ever created. It was created by IBM during the early 1960s, in conjunction with two other American manufacturing conglomerates (Rockwell and Caterpillar) for NASA's Project Apollo See also: https://en.wikipedia.org/wiki/IBM_Information_Management_System. 38

iOS A mobile operating system created and developed by Apple Inc. exclusively for its hardware. See also: https://en.wikipedia.org/wiki/IOS. 84

IP The principal communications protocol in the Internet protocol suite for relaying datagrams across network boundaries. Its routing function enables internetworking, and essentially establishes the Internet. See also: https://en.wikipedia.org/wiki/Internet_Protocol. 34, 56

IR The activity of obtaining information system resources relevant to an information need from a collection of information resources. Searches can be based on full-text or other content-based indexing. See also: https://en.wikipedia.org/wiki/Information_retrieval. 48

IRCAM A French institute for science about music and sound and avant garde electro-acoustical art music. See also: https://www.ircam.fr/. 70, 77, 141

IRIS IRIS is a consortium of over 120 US universities dedicated to the operation of science facilities for the acquisition, management, and distribution of seismological data. See also: https://www.iris.edu. 57

ISMIR An international forum for research on the organization of music-related data. See also: http://ismir.net. 49

IXD This is an index file in text form that shows closed captions and file offset information from a MPEG or WM video file. The .IXD file is used by VBrick Systems, Videoalive, Discovervideo.com, and other vendors. See also: https://filext.com/file-extension/IXD. 86

JSON An open-standard file format that uses human-readable text to transmit data objects consisting of attribute-value pairs and array data types (or any other serializable value) See also: https://www.json.org/. 42, 86, 176

LISP A family of computer programming languages with a long history and a distinctive, fully parenthesized prefix notation. Originally specified in 1958, Lisp is the second-oldest high-level programming language in widespread use today. Linked lists are one of Lisp's major data structures, and Lisp source code is made of lists. See also: https://en.wikipedia.org/wiki/Lisp_(programming_language). 46, 75, 137

Looperman Looperman is a Free pro audio community for musicians, film and video producers, djs and multi media designers. See also: https://www.looperman.com/. 49, 79

LPC A tool used mostly in audio signal processing and speech processing for representing the spectral envelope of a digital signal of speech in compressed form, using the information of a linear predictive model See also: https://en.wikipedia.org/wiki/Linear_predictive_coding. 81

madBPM A modular C++ software platform serving as a data-ingestion engine suitable for database perceptualization (i.e. sonification and visualization). 59

Marsyas Marsyas is an open source software framework for audio processing with specific emphasis on Music Information Retrieval applications. It has been designed and written by George Tzanetakis with help from students and researchers from around the world. Marsyas has been used for a variety of projects in both academia and industry. See also: http://marsyas.info/. 50

Max Max programming language. Named after Max Mathews.. 70–74, 154, 156

MAX/MSP Also known as Max/MSP/Jitter, is a visual programming language for music and multimedia developed and maintained by San Francisco-based software company Cycling '74. Max (Mathews) + ("Max Signal Processing", or the initials Miller Smith Puckette). 42, 59, 72, 76, 79, 80, 162

MetriXML A computer music synthesis programming language based in the Music-N type. See also: https://xamat.github.io/Thesis/html-thesis/. 86

MIDI A technical standard that describes a communications protocol, digital interface, and electrical connectors that connect a wide variety of electronic musical instruments, computers, and related audio devices. See also: https://en.wikipedia.org/wiki/MIDI. 65, 137

MIMO The world's largest freely accessible database for information on musical instruments held in public collections. See also: http://www.mimo-international.com. 50

MIR The interdisciplinary science of retrieving information from music. MIR is a small but growing field of research with many real-world applications. Those involved in MIR may have a background in musicology, psychoacoustics, psychology, academic music study, signal processing, informatics, machine learning, optical music recognition, computational intelligence or some combination of these. See also: https://en.wikipedia.org/wiki/Music_information_retrieval. 2, 47, 48, 79, 172

MIT Private research university in Cambridge, Massachusetts. See also: http://www.mit.edu/. 50, 62

MongoDB A cross-platform document-oriented database program. Classified as a NoSQL database program, MongoDB uses JSON-like documents with schemata.. 34, 46

MuseData The MuseData database is a project of the Center for Computer Assisted Research in the Humanities (CCARH). The database was created by Walter Hewlett. Data entry has been primarily done by Frances Bennion, Edmund Correia, Walter Hewlett, and Steve Rasmussen.. 50

MUSIC-11 Barry Vercoe's development of MUSIC-IV which later grew into the still widely used Csound.. 68, 70

MUSIC-N MUSIC-N refers to a family of computer music programs and programming languages descended from or influenced by MUSIC, a program written by Max Mathews in 1957 at Bell Labs.. 68

MUSIC-V See MUSIC-N. 67, 72, 152–154

Music21 Music21 is a set of tools for helping scholars and other active listeners answer questions about music quickly and simply. It is a python module.. 50

MUSICOMP “Development of MUSICOMP began in the late 1950s, and is considered by Loy as ‘the granddaddy of all programming systems for automatic music generation’ (1989, p. 368) and by Roads as the ‘first composition language’ (1996, p. 815).... From 1967 to 1969 these tools were used [with John Cage] in the production of HPSCHD” See also: . 148

MusicXML MusicXML is an Extensible Markup Language (XML)-based file format for representing Western musical notation. The format is open, fully documented, and can be freely used. See also: https://en.wikipedia.org/wiki/MusicXML. 66

MySQL An open source relational database management system (RDBMS). Its name is a combination of "My", the name of co-founder Michael Widenius's daughter, and "SQL", the abbreviation for Structured Query Language.. 34, 46, 83

NASA An independent agency of the United States Federal Government responsible for the civilian space program, as well as aeronautics and aerospace research.NASA was established in 1958, succeeding the National Advisory Committee for Aeronautics (NACA). See also: https://www.nasa.gov/. 38

NC2IF The Center for Network-Centric Cognition and Information Fusion (NC2IF) explores the information chain from energy detection via sensors and human observation to physical modeling, signal and image processing, pattern recognition, knowledge creation, information infrastructure, and human decision-making-all in the context of organizations and the nation. See also: https://ist.psu.edu/research/centers_labs/nc2if. 56

Neo4j A graph database management system developed by Neo4j, Inc. . 46

NetworkX NetworkX is a Python package for the creation, manipulation, and study of the structure, dynamics, and functions of complex networks. See also: http://networkx.github.io. 46

NeXT An American computer and software company founded in 1985 by Apple Computer cofounder Steve Jobs.. 69

NIME An international conference dedicated to scientific research on the development of new technologies and their role in musical expression and artistic performance. Researchers and musicians from all over the world gather to share their knowledge and late-breaking work on new musical interface design. See also: https://en.wikipedia.org/wiki/New_Interfaces_for_Musical_Expression. 56

NMR A physical phenomenon in which nuclei in a strong static magnetic field are perturbed by a weak oscillating magnetic field (in the near field and therefore not involving electromagnetic waves) and respond by producing an electromagnetic signal with a frequency characteristic of the magnetic field at the nucleus. See also: https://en.wikipedia.org/wiki/Nuclear_magnetic_resonance. 56

NoSQL A mechanism for storage and retrieval of data that is modeled in means other than the tabular relations used in relational databases. See also: https://en.wikipedia.org/wiki/NoSQL. 34, 41

Objective-C Objective-C is a programming language combining the Smalltalk messaging system and the C programming language, which enables an object-oriented approach to the latter.. 69

OFX openFrameworks is an open source toolkit designed for creative coding. It is written in C++ and built on top of OpenGL. It runs on Microsoft Windows, macOS, Linux, iOS, Android and Emscripten. See also: https://openframeworks.cc. 59

Oracle An American multinational computer technology corporation headquartered in Redwood Shores, California. See also: https://en.wikipedia.org/wiki/Oracle_Corporation. 19, 46

OS 2200 The OS 2200 database managers are all part of the Universal Data System (UDS). UDS provides a common control structure for multiple different data models. Flat files (sequential, multi-keyed indexed sequential - MSAM, and fixed-block), network (DMS), and relational (RDMS) data models all share a common locking, recovery, and clustering mechanism. OS 2200 applications can use any mixtures of these data models along with the high-volume transaction file system within the same program while retaining a single common recovery mechanism. See also: https://en.wikipedia.org/wiki/Unisys_OS_2200_databases. 46

OSC A protocol for networking sound synthesizers, computers, and other multimedia devices for purposes such as musical performance or show control. OSC's advantages include interoperability, accuracy, flexibility and enhanced organization and documentation. See also: https://en.wikipedia.org/wiki/Open_Sound_Control. 58, 85

PostgreSQL An open source object-relational database management system with an emphasis on extensibility and standards compliance. It can handle workloads ranging from small single-machine applications to large Internet-facing applications with many concurrent users.. 34, 46, 85

QCD-Audio QCD-audio works on data of computer physics, stemming from the Institute for Physics. Our interdisciplinary proposal QCD-Audio will develop sonification techniques for data of numerical models in physics.. 54

QED In particle physics, quantum electrodynamics is the relativistic quantum field theory of electrodynamics See also: https://en.wikipedia.org/wiki/Quantum_electrodynamics. 54

RDM An ACID-compliant embedded database management system designed for use in embedded systems applications. RDM has been designed to utilize multi-core computers, networking (local or wide area), and on-disk or in-memory storage management. See also: https://en.wikipedia.org/wiki/Raima_Database_Manager. 46

Redis Redis is an open source (BSD licensed), in-memory data structure store, used as a database, cache and message broker. 46

RISM The International Inventory of Musical Sources —Repertoire International des Sources Musicales (RISM)— is an international, non-profit organization which aims for comprehensive documentation of extant musical sources worldwide. . . Nearly all of the records may be downloaded as open data and linked open data in MARCXML and RDF format under a Creative Commons CC-BY license. See also: https://opac.rism.info. 50

ROI Samples within a data set identified for a particular purpose. For example, in medical imaging, the boundaries of a tumor may be defined on an image or in a volume, for the purpose of measuring its size. See also: https://en.wikipedia.org/wiki/Region_of_interest. 54, 180

RTcmix A real-time software "language" for doing digital sound synthesis and signal-processing. It is written in C/C++, and is distributed open-source, free of charge. See also: http://rtcmix.org/. 75

SAP HANA An in-memory, column-oriented, relational database management system developed and marketed by SAP SE See also: https://en.wikipedia.org/wiki/SAP_HANA. 46

SCORE A scorewriter program, written in FORTRAN for DOS by Stanford Professor Leland Smith (1925-2013) with a reputation for producing very high-quality results.. 67–69

SCRIVA Notation software for the Structured Sound Synthesis Project (SSSP). 61, 64

SDIF A standard for the well-defined and extensible interchange of a variety of sound descriptions. See also: https://en.wikipedia.org/wiki/SDIF. 86

Sedna Sedna is a free native XML database which provides a full range of core database services - persistent storage, ACID transactions, security, indices, hot backup. Flexible XML processing facilities include W3C XQuery implementation, tight integration of XQuery with full-text search facilities and a node-level update language.. 46

Smalltalk Smalltalk is an object-oriented, dynamically typed reflective programming language.. 46, 68

SonART A flexible, multi-purpose multimedia environment that allows for networked collaborative interaction with applications for art, science and industry. It provides an open ended framework for integration of powerful image and audio processing methods with a flexible network features.. 58

SonData SonData is an Interactive Data Sonification toolkit, targeted at all practitioners interested in sonifying data. However, it also provides a set of tools that are useful to the academic and scientific Data Sonification community. See also: https://github.com/JoaoMenezes/SonData. 59

Sparksee Sparksee is a high-performance and scalable graph database management system written in C++. Its development started in 2006 and its first version was available on Q3 - 2008. The fourth version is available since Q3-2010.. 46

Sparsity High-performance human solutions for Extreme Data. 46

SPEAR An application for audio analysis, editing and synthesis. The analysis procedure (which is based on the traditional McAulay-Quatieri technique) attempts to represent a sound with many individual sinusoidal tracks (partials), each corresponding to a single sinusoidal wave with time varying frequency and amplitude. See also: http://www.klingbeil.com/spear/. 86

SQL A domain-specific language used in programming and designed for managing data held in a relational database management system, or for stream processing in a relational data stream management system. See also: https://en.wikipedia.org/wiki/SQL. 41, 80

SQLite A relational database management system contained in a C programming library. In contrast to many other database management systems, SQLite is not a client-server database engine. Rather, it is embedded into the end program. See also: https://en.wikipedia.org/wiki/SQLite. 46

SQLObject SQLObject is a popular Object Relational Manager for providing an object interface to your database, with tables as classes, rows as instances, and columns as attributes. SQLObject includes a Python-object-based query language that makes SQL more abstract, and provides substantial database independence for applications.. 41, 46

SSSP An interdisciplinary project whose aim is to conduct research into problems and benefits arising from the use of computers in musical composition See also: https://www.billbuxton.com/SSSP.html. 36, 61

STFT A Fourier-related transform used to determine the sinusoidal frequency and phase content of local sections of a signal as it changes over time. See also: https://en.wikipedia.org/wiki/Short-time_Fourier_transform. 78

TCP/IP The Internet protocol suite is the conceptual model and set of communications protocols used in the Internet and similar computer networks. It is commonly known as TCP/IP because the foundational protocols in the suite are the Transmission Control Protocol (TCP) and the Internet Protocol (IP). See also: https://en.wikipedia.org/wiki/Internet_protocol_suite. 44, 75

Telemeta Telemeta is a free and open source collaborative multimedia asset management system (MAM) which introduces fast and secure methods to archive, backup, transcode, analyse, annotate and publish any digitalized video or audio file with extensive metadata. It is dedicated to collaborative media archiving projects, research laboratories and digital humanities — especially in ethno-musicological use cases — who need to easily organize and publish documented sound collections of audio files, CDs, digitalized vinyls and magnetic tapes

over a strong database, through a smart and secure platform, in accordance with open web standards. Telemeta stands for Tele for “remote access” and meta for “metadata.”. 49

TurboIMAGE A database developed by Hewlett Packard and included with the HP3000 minicomputer.. 46

UbuWeb UbuWeb is a large web-based educational resource for avant-garde material available on the internet, founded in 1996 by poet Kenneth Goldsmith. It offers visual, concrete and sound poetry, expanding to include film and sound art mp3 archives.. 50

Unisys Unisys Corporation is an American global information technology company based in Blue Bell, Pennsylvania, that provides a portfolio of IT services, software, and technology. It is the legacy proprietor of the Burroughs and UNIVAC line of computers, formed when the former bought the latter. See also: https://www.unisys.com. 46

XML A markup language that defines a set of rules for encoding documents in a format that is both human-readable and machine-readable. The W3C’s XML 1.0 Specification and several other related specifications-all of them free open standards-define XML See also: https://en.wikipedia.org/wiki/XML. 41, 86

XPlain A semantic database model developed by J.H. ter Bekke. 46

YAML A human-readable data serialization language. It is commonly used for configuration files, but could be used in many applications where data is being stored or transmitted. See also: https://en.wikipedia.org/wiki/YAML. 41, 86

Acronyms

Access Microsoft Access. 46, *Glossary:* Access

AFLOWLIB Automatic Flow for Materials Discovery. 59, *Glossary:* AFLOWLIB

API Application Programming Interface. 84, *Glossary:* API

ASE Adaptive Server Enterprise. 46, *Glossary:* ASE

ASF Apache Software Foundation. *Glossary:* ASF

AT Actor-Network Theory. 98–100, 105–107

AutoCAD Automatic Computer Assisted Design. 142, *Glossary:* AutoCAD

BFCC Bark Frequency Cepstrum Coefficient. *Glossary:* BFCC

BOOST Boost Software Library. 46, *Glossary:* BOOST

CAAC Computer-Aided Algorithmic Composition. 38, *Glossary:* CAAC

CAC Computer Aided Composition. 31, 69, 134, 137, 139, 149, 153

CAD Computer Aided Design. *Glossary:* CAD

CADDC Computer Aided Data Driven Composition. 59

CAMP Computer Assisted Music Project. 36, 64, *Glossary:* CAMP

CASE Computer Aided Software Engineering. *Glossary:* CASE

CataRT Real-Time Corpus-Based Concatenative Synthesis. 79, *Glossary:* CataRT

CC Creative Commons. 87, *Glossary:* CC

CCRMA Center for Computer Research in Music and Acoustics. 68, 81, *Glossary:* CCRMA

CDIP Coastal Data Information Program. 55, *Glossary:* CDIP

CERL Computer Based Education Research Laboratory. 71, *Glossary:* CERL

CMAM Centre des Musiques Arabes et Mediterraneennes. 49, *Glossary:* CMAM

CNMAT Center for New Music and Audio Technologies. 85, *Glossary:* CNMAT

COBOL Common Business Oriented Language. 39, *Glossary:* COBOL

CODASYL Conference/Committee on Data Systems Languages. 39, 46, *Glossary:* CODASYL

COMPath Composition Path. *Glossary:* COMPath

CouchDB Appache Couch Database. 34, 46, *Glossary:* CouchDB

CPU Central Processing Unit. 18, 57, 76, *Glossary:* CPU

CSV Comma Separated Values. 59, 180, *Glossary:* CSV

CUIDADO Content Based Unified Interfaces and Descriptors for Audio/music Databases available Online. 79, 80, *Glossary:* CUIDADO

CWM Cluster Weighted Modeling. *Glossary:* CWM

DARMS Digital Alternate Representation of Musical Scores. 67, 68, *Glossary:* DARMS

DBMS Database Managemet System. 31, 33, 34, 36, 37, 85, *Glossary:* DBMS

DDL Data Definition Language. 37, *Glossary:* DDL

DJ Disk Jockey. 80, *Glossary:* DJ

DML Data Manipulation Language. 37, *Glossary:* DML

DOM Document Object Model. 40, *Glossary:* DOM

DSL Domain Specific Language. 59, *Glossary:* DSL

EDM Electronic Dance Music. 87, *Glossary:* EDM

EMI Experiments in Music Intelligence. 137, 138, *Glossary:* EMI

ER Entity Relationship. *Glossary:* ER

EsAC Essen Associative Code. 50, *Glossary:* EsAC

FDN Feedback Delay Network. 97, *Glossary:* FDN

FM Frequency Modulation. 57, *Glossary:* FM

FORTRAN Formula Translation. *Glossary:* FORTRAN

GB Gigabyte. 55, *Glossary:* GB

GDIF Gesture Description Interchange Format. 86, *Glossary:* GDIF

GEM Graphics Environment for Multimedia. 81, *Glossary:* GEM

GPU Graphics Processing Unit. 57, *Glossary:* GPU

GSR Galvanic Skin Response. 83, *Glossary:* GSR

GTTM Generative Theory of Tonal Music. 51, *Glossary:* GTTM

GUI Graphical User Interface. 44, 61, 64, *Glossary:* GUI

GUIDO GUIDO Music Notation Format. 66, *Glossary:* GUIDO

HCI Human Computer Interaction. 61–63, *Glossary:* HCI

HMDB Human Metabolome Database. 56, *Glossary:* HMDB

HMSL Hierarchical Music Specification Language. 72, *Glossary:* HMSL

HP Hewlett Packard. 46, *Glossary:* HP

HTML Hypertext Markup Language. *Glossary:* HTML

HTTP Hypertext Transfer Protocol . 56, *Glossary:* HTTP

IBM International Business Machines Corporation. 38, 46, *Glossary:* IBM

ICMC International Computer Music Conference. 56, 67, 70, 74, 75, *Glossary:* ICMC

IDMS Integrated Database Management System. 46, *Glossary:* IDMS

IDS Integrated Data Store. 40, 46, *Glossary:* IDS

IEM Institute of Electronic Music and Acoustics. 54, *Glossary:* IEM

IFT Inverse Fourier Transform. 56, 58

IMDB Internet Movie Database. 44

IMDBs In Memory Data Bases. *Glossary:* IMDBs

IMS Information Management System. 38, 46, *Glossary:* IMS

iOS I Operating System. 84, *Glossary:* iOS

IP Internet Protocol. 34, 56, 57, *Glossary:* IP

IR Information Retrieval. 48, *Glossary:* IR

IRCAM Institut de Recherche et Coordination Acoustique/Musique. 70, 75, 77, 79, 141, 142, *Glossary:* IRCAM

IRIS Incorporated Research Institutions for Seismology. 57, *Glossary:* IRIS

ISMIR International Society for Music Information Retrieval. 49, *Glossary:* ISMIR

IXD Videoalive Indexer Extracted Closed Captions and Metadata. 86, *Glossary:* IXD

JSON JavaScript Object Notation. 41, 43, 86, 176, 180–183, 185, 186, *Glossary:* JSON

LBDM Local Boundaries Detection Model. *Glossary:* LBDM

LFCC Log Frequency Cepstral Coefficients. 80

LISP LISt Processor. 46, 75, 137, *Glossary:* LISP

Looperman Looperman.com. 49, 50, 79, *Glossary:* Looperman

LPC Linear Predictive Coding. 81, *Glossary:* LPC

Marsyas Music Analysis, Retrieval and Synthesis for Audio Signals. 50, *Glossary:* Marsyas

MetriXML MetriX in Extensible Markup Language. 86, *Glossary:* MetriXML

MIDI Musical Instrument Digital Interface. 65, 66, 70, 75, 137, *Glossary:* MIDI

MIMO Musical Instrument Museums Online. 50, *Glossary:* MIMO

MIR Music Information Retrieval . 2, 47–50, 79, 88, 172, *Glossary:* MIR

MIT Massachusetts Institute of Technology . 50, 62, 70, 74, *Glossary:* MIT

MUSICOMP MUsic SImulator-Interpreter for COMpositional Procedures. 148, *Glossary:* MUSICOMP

MusicXML MusicXML. 66, *Glossary:* MusicXML

NASA National Aeronautics and Space Administration. 38, *Glossary:* NASA

NC2IF Center for Network-Centric Cognition and Information Fusion. 56, *Glossary:* NC2IF

NetworkX NetworkX. 46, *Glossary:* NetworkX

NIME New Interfaces for Musical Expression. 56, *Glossary:* NIME

NMR Nuclear magnetic resonance. 56, *Glossary:* NMR

NoSQL Non or Not only SQL. 34, 41, 46, *Glossary:* NoSQL

NYU New York University. *Glossary:* NYU

OFX OpenFrameworks. 59, *Glossary:* OFX

Oracle Oracle Corporation. 19, 46, *Glossary:* Oracle

OS 2200 Unisys OS 2200 databases. 46, *Glossary:* OS 2200

OSC Open Sound Control . 58, 85, *Glossary:* OSC

PVC Phase Vocoder. *Glossary:* PVC

QED Quantuum Electrodynamics. 54, *Glossary:* QED

RAM Random Access Memory . *Glossary:* RAM

RDM Raima Database Manager . 46, *Glossary:* RDM

RISM Repertoire International des Sources Musicales. 50, *Glossary:* RISM

RM/T Relational Model/Tasmania. *Glossary:* RM/T

ROI Region of Interest. 54, 180, *Glossary:* ROI

RTcmix Realtime Cmix. 75, 76, *Glossary:* RTcmix

RWC Real World Computing Music Database. *Glossary:* RWC

SAP HANA High Performance Analytics Appliance. 46, *Glossary:* SAP HANA

SDIF Sound Description Interchange Format. 86, *Glossary:* SDIF

SMC Sound and Music Computing Conference. 56

SMIL Synchronized Multimedia Integration Language. *Glossary:* SMIL

SonData Sonifying Data. 59, *Glossary:* SonData

SPEAR Sinusoidal Partial Editing Analysis and Resynthesis. 86, *Glossary:* SPEAR

SQL Structured Query Language. 41, 80, *Glossary:* SQL

SQLite Structured Query Language Lite. 46, *Glossary:* SQLite

SSSP Structured Sound Synthesis Project. 36, 61, 62, 64, 68, 74, *Glossary:* SSSP

STFT Short Time Fourier Transform. 78, *Glossary:* STFT

STRAIGHT Speech Transformation and Representation using Adaptive Interpolation of weiGHTed spectrogram. *Glossary:* STRAIGHT

TCP/IP Transmission Control Protocol / Internet Protocol. 44, 75, *Glossary:* TCP/IP

timbreID Timbre Identification. *Glossary:* timbreID

Unisys United Information Systems. 46, *Glossary:* Unisys

URI Uniform Resource Identifier. *Glossary:* URI

URL Uniform Resource Locator . *Glossary:* URL

XML Extensible Markup Language. 41, 43, 46, 86, *Glossary:* XML

YAML YAML Ain't Markup Language. 41, 43, 86, *Glossary:* YAML

Bibliography

Abiteboul, S. (1996). *Querying semi-structured data* (Technical Report No. 1996-19). Stanford InfoLab. Stanford InfoLab. Retrieved from http://ilpubs.stanford.edu:8090/144/

Abiteboul, S., Hull, R., & Vianu, V. (1995). *Foundations of databases*. Addison-Wesley.

Amatriain, X. (2004). *An object-oriented metamodel for digital signal processing with a focus on audio and music* (Doctoral dissertation, Universitat Pompeu Fabra). Retrieved from https://xamat.github.io//Thesis/html-thesis/node51.html

Ames, C. (1985). Applications of linked data structures to automated composition. In *Proceedings of the international computer music conference, ICMC 1985*, Michigan Publishing. Retrieved from https://quod.lib.umich.edu/i/icmc/bbp2372.1985.040/1

Angles, R., & Gutierrez, C. (2008). Survey of graph database models. *ACM Computing Surveys*, *40*(1).

Antila, C., & Cumming, J. (2014). The VIS framework: Analyzing counterpoint in large datasets. In H. Wang, Y. Yang, & J. H. Lee (Eds.), *Proceedings of the 15th international society for music information retrieval conference, ISMIR 2014, taipei, taiwan, october 27-31, 2014* (pp. 71–76). Retrieved from http://www.terasoft.com.tw/conf/ismir2014/proceedings/T014_162_Paper.pdf

Ariza, C. (2003). Ornament as data structure: An algorithmic model based on micro-rhythms of csng laments and funeral music. In *Proceedings of the international computer music conference, ICMC 2003*, Michigan Publishing. Retrieved from https://quod.lib.umich.edu/i/icmc/bbp2372.2003.030/1

Ariza, C. (2005a). *An open design for computer-aided algorithmic music composition: Athenacl* (Doctoral dissertation). Copyright - Database copyright ProQuest LLC; ProQuest does not claim copyright in the individual underlying works; Last updated - 2017-02-23. Retrieved from http://proxy.library.nyu.edu/login?url=https://search-proquest-com.proxy.library.nyu.edu/docview/305469710?accountid=12768

Ariza, C. (2005b). The xenakis sieve as object: A new model and a complete implementation. *Computer Music Journal*, *29*(2), 40–60. Retrieved from https://www.jstor.org/stable/3681712

Assayag, G., Agón, C., Fineberg, J., & Hanappe, P. (1997). An object oriented visual environment for musical composition. In *Proceedings of the 1997 international computer music conference, ICMC 1997, thessaloniki, greece, september 25-30, 1997*, Michigan Publishing. Retrieved from http://hdl.handle.net/2027/spo.bbp2372.1997.097

Assayag, G., Dubnov, S., & Delerue, O. (1999). Guessing the composer's mind: Applying universal prediction to musical style. In *Proceedings of the 1999 international computer music con-*

ference, ICMC 1999, beijing, china, october 22-27, 1999, Michigan Publishing. Retrieved from http://hdl.handle.net/2027/spo.bbp2372.1999.454

Attali, J. (2009). *Noise: The political economy of music*. University of Minnesota Press.

Bachman, C. W. (1973). The programmer as navigator. *Commun. ACM*, *16*(11), 653–658. doi:10.1145/355611.362534

Ballora, M. (2000). *Data analysis through auditory display: Applications in heart rate variability* (Doctoral dissertation, McGill University). Retrieved from http://www.markballora.com/publications/diss.pdf

Ballora, M., Panulla, B., Gourley, M., & Hall, D. (2010). Sonification of web log data. In *Proceedings of the international computer music conference, ICMC 2010*, Michigan Publishing. Retrieved from https://quod.lib.umich.edu/i/icmc/bbp2372.2010.117/1

Barrett, N. (2000a). A compositional methodology based on data extracted from natural phenomena. In *Proceedings of the international computer music conference, ICMC 2000*, Michigan Publishing. Retrieved from https://quod.lib.umich.edu/i/icmc/bbp2372.2000.123/1

Barrett, N. (2000b). Viva la selva. Retrieved from http://www.natashabarrett.org/viva.html

Barthes, R., Lavers, A., & Smith, C. (1968). *Elements of semiology*. Hill and Wang, New York.

Beilharz, K., & Ferguson, S. (2009). Aesthetic sonification toolkit for real-time interaction with data. (pp. 401–408).

Ben-Tal, O., Berger, J., Cook, B., Daniels, M., & Scavone, G. (2002). Sonart: The sonification application research toolbox. In *Presented at the 8th international conference on auditory display (icad), kyoto, japan, july 2-5, 2002*, Georgia Institute of Technology. Retrieved from http://hdl.handle.net/1853/51376

Bertin-Mahieux, T., Ellis, D. P. W., Whitman, B., & Lamere, P. (2011). The million song dataset. In A. Klapuri & C. Leider (Eds.), *Proceedings of the 12th international society for music information retrieval conference, ISMIR 2011, miami, florida, usa, october 24-28, 2011* (pp. 591–596). University of Miami. Retrieved from http://ismir2011.ismir.net/papers/OS6-1.pdf

Bittner, R. M., Salamon, J., Tierney, M., Mauch, M., Cannam, C., & Bello, J. P. (2014). Medleydb: A multitrack dataset for annotation-intensive MIR research. In H. Wang, Y. Yang, & J. H. Lee (Eds.), *Proceedings of the 15th international society for music information retrieval conference, ISMIR 2014, taipei, taiwan, october 27-31, 2014* (pp. 155–160). Retrieved from http://www.terasoft.com.tw/conf/ismir2014/proceedings/T028_322_Paper.pdf

Bloch, G., & Dubnov, S. (2008). Introducing video features and spectral descriptors in the omax improvisation system. In *Proceedings of the 2008 international computer music conference, ICMC 2008, belfast, ireland, august 24-29, 2008*, Michigan Publishing. Retrieved from http://hdl.handle.net/2027/spo.bbp2372.2008.090

Bogdanov, D., Wack, N., Gómez, E., Gulati, S., Herrera, P., Mayor, O., ... Serra, X. (2013). Essentia: An audio analysis library for music information retrieval. In A. de Souza Britto Jr., F. Gouyon, & S. Dixon (Eds.), *Proceedings of the 14th international society for music information retrieval conference, ISMIR 2013, curitiba, brazil, november 4-8, 2013* (pp. 493–498). Retrieved from http://www.ppgia.pucpr.br/ismir2013/wp-content/uploads/2013/09/177%5C_Paper.pdf

Boie, R., Mathews, M., & Schloss, A. (1989). The radio drum as a synthesizer controller. In *Proceedings of the 1989 international computer music conference, ICMC 1989, columbus, ohio, usa, november 2-5, 1989*, Michigan Publishing. Retrieved from http://hdl.handle.net/2027/spo.bbp2372.1989.010

Borges, J. L. (1942). Funes el memorioso. *Ficciones*.

Borges, J. L. (1994). *Ficciones* (A. Kerrigan, Ed.). Grove Press.

Born, G. (1995). *Rationalizing culture*. University of California Press.

Bortz, B., Jaimovich, J., & Knapp, R. (2015). Emotion in motion: A reimagined framework for biomusical/emotional interaction. In E. Berdahl & J. Allison (Eds.), *Proceedings of the international conference on new interfaces for musical expression* (pp. 44–49). Baton Rouge, Louisiana, USA: Louisiana State University. Retrieved from http://www.nime.org/proceedings/2015/nime2015_291.pdf

Boynton, L., Duthen, J., Potard, Y., & Rodet, X. (1986). Adding a graphical user interface to FORMES. In *Proceedings of the 1986 international computer music conference, ICMC 1986, den haag, the netherlands, october 20-24, 1986*, Michigan Publishing. Retrieved from http://hdl.handle.net/2027/spo.bbp2372.1986.025

Brent, W. (2010). A timbre analysis and classification toolkit for pure data. In *Proceedings of the international computer music conference, ICMC 2010*, Michigan Publishing. Retrieved from https://quod.lib.umich.edu/i/icmc/bbp2372.2010.044/1

Bresson, J., & Agon, C. (2004). Sdif sound description data representation and manipulation in computer assisted composition. In *Proceedings of the international computer music conference, ICMC 2004*, Michigan Publishing. Retrieved from https://quod.lib.umich.edu/i/icmc/bbp2372.2004.004/1

Bresson, J., & Agon, C. (2010). Processing sound and music description data using openmusic. In *Proceedings of the international computer music conference, ICMC 2010*, Michigan Publishing. Retrieved from https://quod.lib.umich.edu/i/icmc/bbp2372.2010.129/1

Brinkman, A. R. (1981). Data structures for a music-11 preprocessor. In *Proceedings of the international computer music conference, ICMC 1981*, Michigan Publishing. Retrieved from https://quod.lib.umich.edu/i/icmc/bbp2372.1981.018/1

Brinkman, A. R. (1982). Original version of the score11 manual. *Score11 Manual*. Copyright <c> 1982 by Alexander R. Brinkman. Retrieved from http://ecmc.rochester.edu/ecmc/docs/score11/index.html#Introduction

Brinkman, A. R. (1983). A design for a single pass scanner for the darms music coding language. In *Proceedings of the international computer music conference, ICMC 1980*, Michigan Publishing. Retrieved from https://quod.lib.umich.edu/i/icmc/bbp2372.1983.002/1

Brinkman, A. R. (1984). A data structure for computer analysis of musical scores. In *Proceedings of the international computer music conference, ICMC 1984*, Michigan Publishing. Retrieved from https://quod.lib.umich.edu/i/icmc/bbp2372.1984.033/1

Brzezinski-Spiczak, M., Dobosz, K., Lis, M., & Pintal, M. (2013). Music files search system. *CoRR*, *abs/1309.4345*. arXiv: 1309.4345. Retrieved from http://arxiv.org/abs/1309.4345

Bullock, J., Beattie, D., & Turner, J. (2011). Integra live : A new graphical user interface for live electronic music. In *Proceedings of the international conference on new interfaces for*

musical expression (pp. 387–392). Oslo, Norway. Retrieved from http://www.nime.org/proceedings/2011/nime2011_387.pdf

Bullock, J., & Coccioli, L. (2009). Towards a humane graphical user interface for live electronic music. In *Proceedings of the international conference on new interfaces for musical expression* (pp. 266–267). Pittsburgh, PA, United States. Retrieved from http://www.nime.org/proceedings/2009/nime2009_266.pdf

Bullock, J., & Frisk, H. (2009). An object oriented model for the representation of temporal data in the integra framework. In *Proceedings of the international computer music conference, ICMC 2009*, Michigan Publishing. Retrieved from https://quod.lib.umich.edu/i/icmc/bbp2372.2009.012/1

Buneman, P. (1997). Semistructured data. In *Proceedings of the sixteenth acm sigact-sigmod-sigart symposium on principles of database systems* (pp. 117–121). PODS '97. doi:10.1145/263661.263675

Butler, J. (1988). Performative acts and gender constitution: An essay in phenomenology and feminist theory. *Theatre Journal*, *40*(4).

Buxton, W. (1977). A composer's introduction to computer music. *Interface*, *6*, 57–72.

Buxton, W. (2016a). Objed: The sssp sound editing tool. *Youtube*. Retrieved from https://www.youtube.com/watch?v=pUoHc_2wUjY

Buxton, W. (2016b). Socializing technology for the mobile human. keynote, the next web conference, amsterdam/europe. *Youtube*. Retrieved from https://youtu.be/rEeEofRShAQ

Buxton, W., Fedorkow, G., Baecker, R., Reeves, W. T., Smith, K. C., Ciamaga, G., & Mezei, L. (1978). An overview of the structured sound synthesis project. In *Proceedings of the 1978 international computer music conference, ICMC 1978, evanston, illinois, usa, 1978*, Michigan Publishing. Retrieved from http://hdl.handle.net/2027/spo.bbp2372.1978.031

Buxton, W., Patel, S., Reeves, W. T., & Baecker, R. (1980). "objed" and the design of timbral resources. In *Proceedings of the 1980 international computer music conference, ICMC 1980, new york city, usa, 1980*, Michigan Publishing. Retrieved from http://hdl.handle.net/2027/spo.bbp2372.1980.006

Buxton, W., Reeves, W., Baecker, R., & Mezei, L. (1978). The use of hierarchy and instance in a data structure for computer music. In *Proceedings of the international computer music conference, ICMC 1978*, Michigan Publishing. Retrieved from https://quod.lib.umich.edu/i/icmc/bbp2372.1978.012/1

Cádiz, R. F., de la Cuadra, P., Montoya, A., Marín, V., Andia, M. E., Tejos, C., & Irarrazaval, P. (2015). Sonification of medical images based on statistical descriptors. In *Proceedings of the international computer music conference, ICMC 2015*, Michigan Publishing. Retrieved from https://quod.lib.umich.edu/i/icmc/bbp2372.2015.072/1

Cámara Halac, F. (2018a). *A spectral experience: Self convolution and face tracking*. Accepted at ICMC 2018 and SEAMUS 2019 conferences.

Cámara Halac, F. (2018b). This is for young ears: A response to Elsa Justel's Marelle... *Open Space*, (21), 339–350.

Caramiaux, B., Bevilacqua, F., & Schnell, N. (2011). Sound selection by gestures. In *Proceedings of the international conference on new interfaces for musical expression* (pp. 329–330). Oslo, Norway. Retrieved from http://www.nime.org/proceedings/2011/nime2011_329.pdf

Caraty, M. J., Richard, J. C., & Rodet, X. (1989). "vowel recognition in a data base of continuous speech: Experiments with local and global identification principles". *EUROSPEECH*, 2272.

Carlile, S. (2011). Psychoacoustics. In T. Hermann, A. Hunt, & J. G. Neuhoff (Eds.), *The sonification handbook* (Chap. 3, pp. 41–61). Berlin, Germany: Logos Publishing House. Retrieved from http://sonification.de/handbook/chapters/chapter3/

Carpentier, G., Tardieu, D., Rodet, X., & Saint-James, E. (2006). Imitative and Generative Orchestrations Using Pre- analysed Sounds Databases. doi:10.5281/zenodo.849343

Carter, R. (2017). On the expressive potential of suboptimal speakers. audience participation on mobile devices.

Cartwright, M., & Pardo, B. (2012). Building a Music Search Database Using Human Computation. doi:10.5281/zenodo.850060

Cartwright, M., & Pardo, B. (2014). Synthassist: Querying an audio synthesizer by vocal imitation. In *Proceedings of the international conference on new interfaces for musical expression* (pp. 363–366). London, United Kingdom: Goldsmiths, University of London. Retrieved from http://www.nime.org/proceedings/2014/nime2014_446.pdf

Cascone, K. (2000). The aesthetics of failure: 'post-digital' tendencies in contemporary computer music. *Computer Music Journal*, *24*(4), 12–18. Retrieved from https://doi.org/10.1162/014892600559489

Casey, M. A., & Grierson, M. (2007). Soundspotter / remix-tv: Fast approximate matching for audio and video performance. In *Proceedings of the 2007 international computer music conference, ICMC 2007, copenhagen, denmark, august 27-31, 2007*, Michigan Publishing. Retrieved from http://hdl.handle.net/2027/spo.bbp2372.2007.205

Casey, M. A., & Slaney, M. (2006). Song intersection by approximate nearest neighbor search. In *ISMIR 2006, 7th international conference on music information retrieval, victoria, canada, 8-12 october 2006, proceedings* (pp. 144–149).

Choi, I. (2000). Voices in ruins — composition with residuals. Retrieved from https://vimeo.com/23086026

Choi, I., Zheng, G., & Chen, K. (2000). Embedding a sensory data retrieval system in a movement-sensitive space and a surround sound system. In *Proceedings of the international computer music conference, ICMC 2000*, Michigan Publishing. Retrieved from https://quod.lib.umich.edu/i/icmc/bbp2372.2000.146/1

Ciardi, F. C. (2004). Real time sonification of stock market data with smax. In *Proceedings of the international computer music conference, ICMC 2004*, Michigan Publishing. Retrieved from https://quod.lib.umich.edu/i/icmc/bbp2372.2004.124/1

Clements, P. J. (1980). Musical data structures in a multi-use environment. In *Proceedings of the international computer music conference, ICMC 1980*, Michigan Publishing. Retrieved from https://quod.lib.umich.edu/i/icmc/bbp2372.1980.020/1

Codd, E. F. (1970). A relational model of data for large shared data banks. *Commun. ACM*, *13*(6), 377–387. doi:10.1145/362384.362685

Codd, E. F. (1972). Relational completeness of data base sublanguages. In *Database systems* (pp. 65–98). Prentice-Hall.

Collins, N. (2006). *Towards autonomous agents for live computer music: Realtime machine listening and interactive music systems* (Doctoral dissertation, University of Cambridge).

Collins, N. (2007). Audiovisual concatenative synthesis. In *Proceedings of the 2007 international computer music conference, ICMC 2007, copenhagen, denmark, august 27-31, 2007*, Michigan Publishing. Retrieved from http://hdl.handle.net/2027/spo.bbp2372.2007.187

Collins, N. (2015). The ubuweb electronic music corpus: An mir investigation of a historical database. *Organised Sound*, *20*(1), 122–134. doi:10.1017/S1355771814000533

Collins, N., Mclean, A., Rohrhuber, J., & Ward, A. (2003). Live coding in laptop performance. *Organised Sound*, *8*, 321–329. doi:10.1017/S135577180300030X

Connes, A. (2012). The music of shapes. Retrieved from http://www.alainconnes.org/en/videos.php

Cope, D. (1987a). An expert system for computer-assisted composition. *Computer Music Journal*, *11*(4), 30–46. Retrieved from http://www.jstor.org/stable/3680238

Cope, D. (1987b). Experiments in music intelligence (EMI). In *ICMC*, Michigan Publishing.

Corona, H., & O'Mahony, M. P. (2015). An Exploration of Mood Classification in the Million Songs Dataset. doi:10.5281/zenodo.851021

Correa, D. C., Saito, J. H., & Costa, L. d. F. (2010). Musical genres: beating to the rhythms of different drums. *New Journal of Physics*, *12*, 053030. doi:10.1088/1367-2630/12/5/053030. arXiv: 0911.3842 [physics.data-an]

Correia, N. N. (2010). AV Clash - Online Tool for Mixing and Visualizing Audio Retrieved From freesound.org Database. doi:10.5281/zenodo.849729

Crestel, L., Esling, P., Heng, L., & McAdams, S. (2017). A database linking piano and orchestral MIDI scores with application to automatic projective orchestration. In S. J. Cunningham, Z. Duan, X. Hu, & D. Turnbull (Eds.), *Proceedings of the 18th international society for music information retrieval conference, ISMIR 2017, suzhou, china, october 23-27, 2017* (pp. 592–598). Retrieved from https://ismir2017.smcnus.org/wp-content/uploads/2017/10/235_Paper.pdf

Crowley, C. (1998). Data structures for text sequences. In . Retrieved from https://www.cs.unm.edu/~crowley/papers/sds.pdf

Daniel, S. (2007). The database: An aesthetics of dignity. *Database Aesthetics: Art in the Age of Information Overflow*.

Defferrard, M., Benzi, K., Vandergheynst, P., & Bresson, X. (2017). FMA: A dataset for music analysis. In S. J. Cunningham, Z. Duan, X. Hu, & D. Turnbull (Eds.), *Proceedings of the 18th international society for music information retrieval conference, ISMIR 2017, suzhou, china, october 23-27, 2017* (pp. 316–323). Retrieved from https://ismir2017.smcnus.org/wp-content/uploads/2017/10/75_Paper.pdf

Dehkordi, M. B., & Banitalebi-Dehkordi, A. (2018). Music genre classification using spectral analysis and sparse representation of the signals. *CoRR*, *abs/1803.04652*. arXiv: 1803.04652. Retrieved from http://arxiv.org/abs/1803.04652

Delbouys, R., Hennequin, R., Piccoli, F., Royo-Letelier, J., & Moussallam, M. (2018). Music mood detection based on audio and lyrics with deep neural net. *CoRR*, *abs/1809.07276*. arXiv: 1809.07276. Retrieved from http://arxiv.org/abs/1809.07276

Depalle, P., Rodet, X., Galas, T., & Eckel, G. (1993). Generalized diphone control. In *Opening a new horizon: Proceedings of the 1993 international computer music conference, ICMC 1993, tokio, japan, september 10-15, 1993*, Michigan Publishing. Retrieved from http://hdl.handle.net/2027/spo.bbp2372.1993.038

Derrida, J. (1978). *Writing and difference*. The University of Chicago.

Derrida, J. (1982). *Margins of philosophy*. The Harvester Press.

Derrida, J., & Prenowitz, E. (1995). Archive fever: A freudian impression. *Diacritics*, *25*(2).

Derrida, J., & Spivak, G. C. (1976). *Of grammatology.* The Johns Hopkins University Press.

Devaney, J., Arthur, C., Condit-Schultz, N., & Nisula, K. (2015). Theme and variation encodings with roman numerals (TAVERN): A new data set for symbolic music analysis. In M. Müller & F. Wiering (Eds.), *Proceedings of the 16th international society for music information retrieval conference, ISMIR 2015, málaga, spain, october 26-30, 2015* (pp. 728–734). Retrieved from http://ismir2015.uma.es/articles/261_Paper.pdf

Didkovsky, N., & Burk, P. L. (2001). Java music specification language, an introduction and overview. In *Proceedings of the 2001 international computer music conference, ICMC 2001, havana, cuba, september 17-22, 2001*, Michigan Publishing. Retrieved from http://hdl.handle.net/2027/spo.bbp2372.2001.007

Diener, G. (1985). *Formal languages in music theory* (Master's thesis, McGill University, Faculty of Music).

Diener, G. (1988). Ttrees: An active data structure for computer music. In *Proceedings of the international computer music conference, ICMC 1988*, Michigan Publishing. Retrieved from https://quod.lib.umich.edu/i/icmc/bbp2372.1988.020/1

Diener, G. (1989). Ttrees: A tool for the compositional environment. *Computer Music Journal*, *13*(2), 77–85. Retrieved from http://www.jstor.org/stable/3680043

Diener, G. R. (1992). A visual programming environment for music notation. In *Proceedings of the 1992 international computer music conference, ICMC 1992, san jose, california, usa, october 14-18, 1992*, Michigan Publishing. Retrieved from http://hdl.handle.net/2027/spo.bbp2372.1992.030

Dinuzzo, F., Pillonetto, G., & Nicolao, G. D. (2008). Client-server multi-task learning from distributed datasets. *CoRR*, *abs/0812.4235*. arXiv: 0812.4235. Retrieved from http://arxiv.org/abs/0812.4235

Donahue, C., Mao, H. H., & McAuley, J. (2018). The NES music database: A multi-instrumental dataset with expressive performance attributes. In E. Gómez, X. Hu, E. Humphrey, & E. Benetos (Eds.), *Proceedings of the 19th international society for music information retrieval conference, ISMIR 2018, paris, france, september 23-27, 2018* (pp. 475–482). Retrieved from http://ismir2018.ircam.fr/doc/pdfs/265_Paper.pdf

Donahue, C., McAuley, J., & Puckette, M. (2018). Adversarial Audio Synthesis. *arXiv e-prints*, arXiv:1802.04208. arXiv: 1802.04208 [cs.SD]

Dunn, J. W. (2000). Beyond VARIATIONS: creating a digital music library. In *ISMIR 2000, 1st international symposium on music information retrieval, plymouth, massachusetts, usa, october 23-25, 2000, proceedings*. Retrieved from http://ismir2000.ismir.net/papers/invites/dunn_invite.pdf

Dydo, J. S. (1987). Data structures in the note processor. In *Proceedings of the international computer music conference, ICMC 1987*, Michigan Publishing. Retrieved from https://quod.lib.umich.edu/i/icmc/bbp2372.1987.045/1

Eco, U. (2004). The poetics of the open work. *Audio Culture: Readings in Modern Music*.

Emmerson, S. (1986). *The language of electroacoustic music*. doi:10.1007/978-1-349-18492-7

Eremenko, V., Demirel, E., Bozkurt, B., & Serra, X. (2018). Audio-aligned jazz harmony dataset for automatic chord transcription and corpus-based research. In E. Gómez, X. Hu, E. Humphrey, & E. Benetos (Eds.), *Proceedings of the 19th international society for music information retrieval conference, ISMIR 2018, paris, france, september 23-27, 2018* (pp. 483–490). Retrieved from http://ismir2018.ircam.fr/doc/pdfs/206_Paper.pdf

Erickson, R. F. (1975). "the darms project": A status report. *Computers and the Humanities*, *9*(6), 291–298. Retrieved from http://www.jstor.org/stable/30204239

Ernst, W. (2013). *Digital memory and the archive*. University of Minnesota Press.

Flusser, V. (2011). *Into the universe of technical images*. University of Minnesota Press.

Fonseca, E., Pons, J., Favory, X., Font, F., Bogdanov, D., Ferraro, A., ... Serra, X. (2017). Freesound datasets: A platform for the creation of open audio datasets. In S. J. Cunningham, Z. Duan, X. Hu, & D. Turnbull (Eds.), *Proceedings of the 18th international society for music information retrieval conference, ISMIR 2017, suzhou, china, october 23-27, 2017* (pp. 486–493). Retrieved from https://ismir2017.smcnus.org/wp-content/uploads/2017/10/161_Paper.pdf

Fox, M. K., Stewart, J., & Hamilton, R. (2017). Madbpm: A modular multimodal environment for data-driven composition and sonification. In *Proceedings of the international computer music conference, ICMC 2017*, Michigan Publishing. Retrieved from http://hdl.handle.net/2027/bbp2372.2017.087

Free, J. (1987). Towards an extensible data structure for the representation of music on computers. In *Proceedings of the international computer music conference, ICMC 1987*, Michigan Publishing. Retrieved from https://quod.lib.umich.edu/i/icmc/bbp2372.1987.046/1

Free, J., & Vytas, P. (1986). What ever happened to sssp? In *Proceedings of the 1986 international computer music conference, ICMC 1986, den haag, the netherlands, october 20-24, 1986*, Michigan Publishing. Retrieved from http://hdl.handle.net/2027/spo.bbp2372.1986.004

Free, J., & Vytas, P. (1988). The CAMP music configuration database. In *Proceedings of the 1988 international computer music conference, ICMC 1988, cologne, germany, september 20-25, 1988*, Michigan Publishing. Retrieved from http://hdl.handle.net/2027/spo.bbp2372.1988.014

Frid, E. (2017). Sonification of women in sound and music computing - the sound of female authorship in icmc, smc and nime proceedings. In *ICMC* (pp. 233–238). Michigan Publishing.

Frisson, C. (2015). *Designing interaction for browsing media collections (by similarity)* (Doctoral dissertation, Universit de Mons). Retrieved from http://tcts.fpms.ac.be/publications/phds/frisson/phd-frisson.pdf

García, F., Vinceslas, L., Tubau, J., & Maestre, E. (2011). Acquisition and study of blowing pressure profiles in recorder playing. In *Proceedings of the international conference on new interfaces for musical expression* (pp. 124–127). Oslo, Norway. Retrieved from http://www.nime.org/proceedings/2011/nime2011_124.pdf

Garton, B., & Topper, D. (1997). Rtcmix - using CMIX in real time. In *Proceedings of the 1997 international computer music conference, ICMC 1997, thessaloniki, greece, september 25-30, 1997*, Michigan Publishing. Retrieved from http://hdl.handle.net/2027/spo.bbp2372.1997.106

Good, M. (2000). Representing music using XML. In *ISMIR 2000, 1st international symposium on music information retrieval, plymouth, massachusetts, usa, october 23-25, 2000, proceedings*. Retrieved from http://ismir2000.ismir.net/posters/good.pdf

Goto, M., Hashiguchi, H., Nishimura, T., & Oka, R. (2002). RWC music database: Popular, classical and jazz music databases. In *ISMIR 2002, 3rd international conference on music information retrieval, paris, france, october 13-17, 2002, proceedings*. Retrieved from http://ismir2002.ismir.net/proceedings/03-SP04-1.pdf

Goto, M., Hashiguchi, H., Nishimura, T., & Oka, R. (2003). RWC music database: Music genre database and musical instrument sound database. In *ISMIR 2003, 4th international conference on music information retrieval, baltimore, maryland, usa, october 27-30, 2003, proceedings*. Retrieved from http://ismir2003.ismir.net/papers/Goto1.PDF

Gratton, P., & Morin, M.-E. (2015). *The nancy dictionary*. Edinburgh University Press.

Guedes, C., Trochidis, K., & Anantapadmanabhan, A. (2018). Modeling Carnatic Rhythm Generation: a Data Driven Approach Based on Rhythmic Analysis. doi:10.5281/zenodo.1422615

Hamanaka, M., Hirata, K., & Tojo, S. (2014). Musical structural analysis database based on GTTM. In H. Wang, Y. Yang, & J. H. Lee (Eds.), *Proceedings of the 15th international society for music information retrieval conference, ISMIR 2014, taipei, taiwan, october 27-31, 2014* (pp. 325–330). Retrieved from http://www.terasoft.com.tw/conf/ismir2014/proceedings/T059_257_Paper.pdf

Hamilton, R. (2006). Bioinformatic response data as a compositional driver. In *Proceedings of the international computer music conference, ICMC 2006*, Michigan Publishing. Retrieved from https://quod.lib.umich.edu/i/icmc/bbp2372.2006.123/1

Hansen, M. B. N. (2002). Cinema beyond cybernetics, or how to frame the digital image. *Configurations*, *10*(1).

Hansen, M. B. N. (2004). *New philosophy for new media*. The MIT Press.

Hashida, M., Matsui, T., & Katayose, H. (2008). A new music database describing deviation information of performance expressions. In J. P. Bello, E. Chew, & D. Turnbull (Eds.), *ISMIR 2008, 9th international conference on music information retrieval, drexel university, philadelphia, pa, usa, september 14-18, 2008* (pp. 489–494). Retrieved from http://ismir2008.ismir.net/papers/ISMIR2008_173.pdf

Hashida, M., Nakamura, E., & Katayose, H. (2017). Constructing PEDB 2nd Edition: A Music Performance Database with Phrase Information. doi:10.5281/zenodo.1401963

Hashida, M., Nakamura, E., & Katayose, H. (2018). CrestMusePEDB 2nd EDITION: MUSIC PERFORMANCE DATABASE WITH PHRASE INFORMATION. doi:10.5281/zenodo.1422503

Hauger, D., Schedl, M., Kosir, A., & Tkalcic, M. (2013). The million musical tweet dataset - what we can learn from microblogs. In A. de Souza Britto Jr., F. Gouyon, & S. Dixon (Eds.), *Proceedings of the 14th international society for music information retrieval conference, ISMIR 2013, curitiba, brazil, november 4-8, 2013* (pp. 189–194). Retrieved from http://www.ppgia.pucpr.br/ismir2013/wp-content/uploads/2013/09/85_Paper.pdf

Haus, G., & Pinto, A. (2005). Mx structural metadata as mir tools. doi:10.5281/zenodo.849297

Hayles, N. K. (1993). The materiality of informatics. *Configurations*, *1*(1).

Hayles, N. K. (1999). *How we became posthuman: Virtual bodies in cybernetics, literature, and informatics*. The University of Chicago Press.

Hildebrandt, T., Hermann, T., & Rinderle-Ma, S. (2014). A Sonification System for Process Monitoring as Secondary Task. In *Proceedings of the 5th ieee conference on cognitive infocommunication (coginfocom 2014)* (pp. 191–196). Vietri sul Mare, Italy: IEEE.

Hiller, L. A., & Isaacson, L. M. (1959). *Experimental music: Composition with an electronic computer*. McGraw-Hill Book Company, Inc.

Hochenbaum, J., Kapur, A., & Wright, M. (2010). Multimodal musician recognition. In *Proceedings of the international conference on new interfaces for musical expression* (pp. 233–237). Sydney, Australia. Retrieved from http://www.nime.org/proceedings/2010/nime2010_233.pdf

Homburg, H., Mierswa, I., Möller, B., Morik, K., & Wurst, M. (2005). A benchmark dataset for audio classification and clustering. In *ISMIR 2005, 6th international conference on music information retrieval, london, uk, 11-15 september 2005, proceedings* (pp. 528–531). Retrieved from http://ismir2005.ismir.net/proceedings/2117.pdf

Hu, X., & Yang, Y.-H. (2014). A Study on Cross-cultural and Cross-dataset Generalizability of Music Mood Regression Models. doi:10.5281/zenodo.850795

Humphrey, E., Durand, S., & McFee, B. (2018). Openmic-2018: An open data-set for multiple instrument recognition. In E. Gómez, X. Hu, E. Humphrey, & E. Benetos (Eds.), *Proceedings of the 19th international society for music information retrieval conference, ISMIR 2018, paris, france, september 23-27, 2018* (pp. 438–444). Retrieved from http://ismir2018.ircam.fr/doc/pdfs/248_Paper.pdf

IV, J. A. M. (1999). *A brief history of algorithmic composition*. Online. Retrieved from https://ccrma.stanford.edu/~blackrse/algorithm.html

Jaimovich, J., & Knapp, R. (2015). Creating biosignal algorithms for musical applications from an extensive physiological database. In E. Berdahl & J. Allison (Eds.), *Proceedings of the international conference on new interfaces for musical expression* (pp. 1–4). Baton Rouge, Louisiana, USA: Louisiana State University. Retrieved from http://www.nime.org/proceedings/2015/nime2015_163.pdf

Jaimovich, J., Ortiz, M., Coghlan, N., & Knapp, R. B. (2012). The emotion in motion experiment: Using an interactive installation as a means for understanding emotional response to music. In *Proceedings of the international conference on new interfaces for musical expression*, Ann

Arbor, Michigan: University of Michigan. Retrieved from http://www.nime.org/proceedings/2012/nime2012_254.pdf

Jones, R., Lagrange, M., & Schloss, W. A. (2007). A hand drumming dataset for physical modeling. In *Proceedings of the 2007 international computer music conference, ICMC 2007, copenhagen, denmark, august 27-31, 2007*, Michigan Publishing. Retrieved from http://hdl.handle.net/2027/spo.bbp2372.2007.083

Jr., C. N. S., Koerich, A. L., & Kaestner, C. A. A. (2008). The latin music database. In J. P. Bello, E. Chew, & D. Turnbull (Eds.), *ISMIR 2008, 9th international conference on music information retrieval, drexel university, philadelphia, pa, usa, september 14-18, 2008* (pp. 451–456). Retrieved from http://ismir2008.ismir.net/papers/ISMIR2008_106.pdf

Kamde, P. M., & Algur, S. P. (2011). A survey on web multimedia mining. *CoRR*, *abs/1109.1145*. arXiv: 1109.1145. Retrieved from http://arxiv.org/abs/1109.1145

Kane, B. (2014). *Sound unseen: Acousmatic sound in theory and practice*. Oxford University Press. Retrieved from http://www.oxfordscholarship.com/view/10.1093/acprof:oso/9780199347841.001.0001/acprof-9780199347841

Karaosmanoglu, M. K. (2012). A turkish makam music symbolic database for music information retrieval: Symbtr. In F. Gouyon, P. Herrera, L. G. Martins, & M. Müller (Eds.), *Proceedings of the 13th international society for music information retrieval conference, ISMIR 2012, mosteiro s.bento da vitória, porto, portugal, october 8-12, 2012* (pp. 223–228). FEUP Edições. Retrieved from http://ismir2012.ismir.net/event/papers/223-ismir-2012.pdf

Karydis, I., Nanopoulos, A., Papadopoulos, A., Cambouropoulos, E., & Manolopoulos, Y. (2007). Horizontal and Vertical Integration/Segregation in Auditory Streaming: A Voice Separation Algorithm for Symbolic Musical Data. doi:10.5281/zenodo.849469

Kernighan, B. W. (1978). *The c programming language*. Englewood Cliffs, N.J.: Prentice-Hall.

Kirlin, P. B. (2014). A data set for computational studies of schenkerian analysis. In H. Wang, Y. Yang, & J. H. Lee (Eds.), *Proceedings of the 15th international society for music information retrieval conference, ISMIR 2014, taipei, taiwan, october 27-31, 2014* (pp. 213–218). Retrieved from http://www.terasoft.com.tw/conf/ismir2014/proceedings/T039_344_Paper.pdf

Klein, J. (1998). The wolves of bays mountain. Published in 2004. Open Space.

Klein, J. (2017). *On my compositional approach*. Lecture given at New York University's Waverly Project, on February 2nd, 2017.

Klein, N. M. (2007). Waiting for the world to explode: How data convert into a novel. *Database Aesthetics: Art in the Age of Information Overflow*.

Knees, P., Faraldo, Á., Herrera, P., Vogl, R., Böck, S., Hörschläger, F., & Goff, M. L. (2015). Two data sets for tempo estimation and key detection in electronic dance music annotated from user corrections. In M. Müller & F. Wiering (Eds.), *Proceedings of the 16th international society for music information retrieval conference, ISMIR 2015, málaga, spain, october 26-30, 2015* (pp. 364–370). Retrieved from http://ismir2015.uma.es/articles/246_Paper.pdf

Kobayashi, R. (2003). Sound clustering synthesis using spectral data. In *Proceedings of the international computer music conference, ICMC 2003*, Michigan Publishing. Retrieved from https://quod.lib.umich.edu/i/icmc/bbp2372.2003.052/1

Kroher, N. (2011). *Acoustic feedbacks in sound reinforcement systems: Investigating the larsen effect*. AV Akademikerverlag.

Lansky, P. (1990). The architecture and musical logic of cmix. In *Proceedings of the 1990 international computer music conference, ICMC 1990, glasgow, scotland, september 10-15, 1990*, Michigan Publishing. Retrieved from http://hdl.handle.net/2027/spo.bbp2372.1990.023

Laske, O. E. 1.-., & Tabor, J. (1999). *Otto laske : Navigating new musical horizons*. Westport, Conn.: Greenwood Press.

Latour, B. (1990). On actor-network theory. a few clarifications plus more than a few complications. *Philosophia*, *25*(3).

Latour, B. (1993). *We have never been modern*. Harvard University Press Cambridge, Massachusetts.

Lewis, G. (2000). Too many notes: Computers, complexity, and culture in voyager. *Leonardo Music Journal*, *10*.

Lewis, G. E. (1999). Interacting with latter-day musical automata. *Contemporary Music Review*, *18*(3), 99–112. doi:10.1080/07494469900640381

Lindborg, P. (2017). Pacific bell tower, a sculptural sound installation for live sonification of earthquake data. In *Proceedings of the international computer music conference, ICMC 2017*, Michigan Publishing. Retrieved from https://quod.lib.umich.edu/i/icmc/bbp2372.2017.033/1

Lindemann, E. (1990). ANIMAL-A rapid prototyping environment for computer music systems. In *Proceedings of the 1990 international computer music conference, ICMC 1990, glasgow, scotland, september 10-15, 1990*, Michigan Publishing. Retrieved from http://hdl.handle.net/2027/spo.bbp2372.1990.068

Liu, Q., Han, Y. C., Kuchera-Morin, J., & Wright, M. (2013). Cloud bridge: A data-driven immersive audio-visual software interface. In *Proceedings of the international conference on new interfaces for musical expression* (pp. 431–436). Daejeon, Republic of Korea: Graduate School of Culture Technology, KAIST. Retrieved from http://nime.org/proceedings/2013/nime2013_250.pdf

Lodha, S., Beahan, J., Joseph, A., & Zane-ulman, B. (1998). Muse: A musical data sonification toolkit.

Long, R., Harrington, M., Hain, R., & Nicholls, G. (2000). *Ims primer*. International Business Machines Corporation. Retrieved from http://www.redbooks.ibm.com/redbooks/pdfs/sg245352.pdf

Lorenz, E. N. (1993). *The essence of chaos*. Seattle: University of Washington Press.

Loviscach, J. (2008). Programming a music synthesizer through data mining. In *Proceedings of the international conference on new interfaces for musical expression* (pp. 221–224). Genoa, Italy. Retrieved from http://www.nime.org/proceedings/2008/nime2008_221.pdf

Loy, G. (1985). Musicians make a standard: The midi phenomenon. *Computer Music Journal*, *9*(4), 8–26. Retrieved from http://www.jstor.org/stable/3679619

Lucier, A. (1970). I am sitting in a room. For voice and electronic tape. Retrieved from http://alucier.web.wesleyan.edu/discography.html

m zmölnig, I., & Eckel, G. (2015). *Live coding: An overview*.

Manovich, L. (2001). *The language of new media*. MIT Press.

Mathews, M. V. (1963). The digital computer as a musical instrument. *Science, New Series*, *142*(3592), 553–557. Retrieved from http://www.jstor.org/stable/1712380

Maxwell, J. B., & Eigenfeldt, A. (2008). A music database and query system for recombinant composition. In J. P. Bello, E. Chew, & D. Turnbull (Eds.), *ISMIR 2008, 9th international conference on music information retrieval, drexel university, philadelphia, pa, usa, september 14-18, 2008* (pp. 75–80). Retrieved from http://ismir2008.ismir.net/papers/ISMIR2008_158.pdf

Mazzoni, D., & Dannenberg, R. B. (2001). A fast data structure for disk-based audio editing. In *Proceedings of the international computer music conference, ICMC 2001*, Michigan Publishing. Retrieved from https://quod.lib.umich.edu/i/icmc/bbp2372.2001.051/1

McCartney, J. (1996). Supercollider, a new real time synthesis language. In *Proceedings of the 1996 international computer music conference, ICMC 1996, hong kong, august 19-24, 1996*, Michigan Publishing. Retrieved from http://hdl.handle.net/2027/spo.bbp2372.1996.078

McCartney, J. (1998). Continued evolution of the supercollider real time synthesis environment. In *Proceedings of the 1998 international computer music conference, ICMC 1998, ann arbor, michigan, usa, october 1-6, 1998*, Michigan Publishing. Retrieved from http://hdl.handle.net/2027/spo.bbp2372.1998.262

McCurdy, I., Heintz, J., Joaquin, J., & Knevel, M. (2015). Methods of writing csound scores. *FLOSS Manuals*. Retrieved from http://write.flossmanuals.net/csound/methods-of-writing-csound-scores/

Melucci, M., & Orio, N. (1999). The use of melodic segmentation for content-based retrieval of musical data. In *Proceedings of the international computer music conference, ICMC 1999*, Michigan Publishing. Retrieved from https://quod.lib.umich.edu/i/icmc/bbp2372.1999.355/1

Meseguer-Brocal, G., Cohen-Hadria, A., & Peeters, G. (2018). DALI: A large dataset of synchronized audio, lyrics and notes, automatically created using teacher-student machine learning paradigm. In E. Gómez, X. Hu, E. Humphrey, & E. Benetos (Eds.), *Proceedings of the 19th international society for music information retrieval conference, ISMIR 2018, paris, france, september 23-27, 2018* (pp. 431–437). Retrieved from http://ismir2018.ircam.fr/doc/pdfs/35_Paper.pdf

Miron, M., & Janer, J. (2017). Generating Data to Train Convolutional Neural Networks for Classical Music Source Separation. doi:10.5281/zenodo.1401923

Mital, P. K., & Grierson, M. (2013). Mining unlabeled electronic music databases through 3D interactive visualization of latent component relationships. In *Proceedings of the international conference on new interfaces for musical expression* (pp. 227–232). Daejeon, Republic of Korea: Graduate School of Culture Technology, KAIST. Retrieved from http://nime.org/proceedings/2013/nime2013_132.pdf

Mitra, J., & Saha, D. (2014). An efficient feature selection in classification of audio files. *CoRR*, *abs/1404.1491*. arXiv: 1404.1491. Retrieved from http://arxiv.org/abs/1404.1491

Morawitz, F. (2016). Molecular sonification of nuclear magnetic resonance data as a novel tool for sound creation. In *Proceedings of the international computer music conference, ICMC 2016*, Michigan Publishing. Retrieved from https://quod.lib.umich.edu/i/icmc/bbp2372.2016.002/1

Morin, P. (2019). *Open data structures*. Creative Commons. Retrieved from opendatastructures.org

Morton, T. (2013). *Hyperobjects: Philosophy and ecology after the end of the world*. University of Minnesota Press.

N. Shepard, R. (1964). Circularity in judgments of relative pitch. *The Journal of the Acoustical Society of America*, *36*, 2346. doi:10.1121/1.1919362

Nagavi, T. C., & Bhajantri, N. U. (2013). An extensive analysis of query by singing/humming system through query proportion. *CoRR*, *abs/1301.1894*. arXiv: 1301.1894. Retrieved from http://arxiv.org/abs/1301.1894

Nagavi, T. C., & Bhajantri, N. U. (2014). Progressive filtering using multiresolution histograms for query by humming system. *CoRR*, *abs/1401.2516*. arXiv: 1401.2516. Retrieved from http://arxiv.org/abs/1401.2516

Nakamoto, M., & Kuhara, Y. (2007). Circle canon chorus system used to enjoy a musical ensemble singing "frog round". In *Proceedings of the international conference on new interfaces for musical expression* (pp. 409–410). New York City, NY, United States. Retrieved from http://www.nime.org/proceedings/2007/nime2007_409.pdf

Nancy, J.-L. (1991). *The inoperative community*. University of Minnesota Press, Minneapolis and Oxford.

Nancy, J.-L. (2007). *Listening*. Fordham University Place.

Nardelli, M. B. (2015). Materialssoundmusic: A computer-aided data-driven composition environment for the sonification and dramatization of scientific data streams. In *Proceedings of the international computer music conference, ICMC 2015*, Michigan Publishing. Retrieved from https://quod.lib.umich.edu/i/icmc/bbp2372.2015.072/1

Nilson, C. (2007). Live coding practice. In *Proceedings of the international conference on new interfaces for musical expression* (pp. 112–117). New York City, NY, United States. Retrieved from http://www.nime.org/proceedings/2007/nime2007_112.pdf

Nilson, C. (2016). *Collected rewritings: Live coding thoughts, 1968-2015.* Retrieved from https://composerprogrammer.com

Norman, A., & Amatriain, X. (2007). Data jockey, a tool for meta-data enhanced digital djing and active listening. In *Proceedings of the international computer music conference, ICMC 2007*, Michigan Publishing. Retrieved from https://quod.lib.umich.edu/i/icmc/bbp2372.2007.117/1

Nort, D. V., Jarvis, I., & Palumbo, M. (2016). Towards a mappable database of emergent gestural meaning. In *Proceedings of the international conference on new interfaces for musical expression* (Vol. 16, pp. 46–50). 2220-4806. Brisbane, Australia: Queensland Conservatorium Griffith University. Retrieved from http://www.nime.org/proceedings/2016/nime2016_paper0010.pdf

Nuanàin, C. Ó., 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* (Vol. 16, pp. 383–387). 2220-4806. Brisbane, Australia: Queensland Conservatorium Griffith University. Retrieved from http://www.nime.org/proceedings/2016/nime2016_paper0076.pdf

Nymoen, K., & Jensenius, A. R. (2011). A Toolbox for Storing and Streaming Music-related Data. doi:10.5281/zenodo.849865

Oliver, J. (2010). The mano controller: A video based hand tracking system. In *Proceedings of the 2010 international computer music conference, ICMC 2010, new york, usa, 2010*, Michigan Publishing. Retrieved from http://www.jaimeoliver.pe/instrumentos/mano/pub-mano

Oliver, J., & Jenkins, M. (2008). The silent drum controller: A new percussive gestural interface. In *Proceedings of the 2008 international computer music conference, ICMC 2008, belfast, ireland, august 24-29, 2008*, Michigan Publishing. Retrieved from http://hdl.handle.net/2027/spo.bbp2372.2008.118

Osaka, N., Sakakibara, K.-I., & Hikichi, T. (2002). The sound synthesis system "otkinshi": Its data structure and graphical user interface. In *Proceedings of the international computer music conference, ICMC 2002*, Michigan Publishing. Retrieved from https://quod.lib.umich.edu/i/icmc/bbp2372.2002.039/1

Oviedo, M. B. (2019). *Memoria, olvido y narración: Funes como antítesis del escritor*. Lecture given at Cornell University.

Paul, C. (2007). The database as system and cultural form: Anatomies of cultural narratives. *Database Aesthetics: Art in the Age of Information Overflow*.

Pauletto, S., & Hunt, A. (2004). A toolkit for interactive sonification. In *Proceedings of icad 04. tenth meeting of the international conference on auditory display, sydney, australia, july 6-9, 2004. ed. barrass, s. and vickers, p. international community for auditory display, 2004.* Retrieved from http://hdl.handle.net/1853/50809

Peron, T., Rodrigues, F. A., & Costa, L. d. F. (2018). Pattern Recognition Approach to Violin Shapes of MIMO database. *arXiv e-prints*, arXiv:1808.02848. arXiv: 1808.02848 [stat.AP]

Pesek, M., Godec, P., Poredos, M., Strle, G., Guna, J., Stojmenova, E., ... Marolt, M. (2014). Introducing a dataset of emotional and color responses to music. In H. Wang, Y. Yang, & J. H. Lee (Eds.), *Proceedings of the 15th international society for music information retrieval conference, ISMIR 2014, taipei, taiwan, october 27-31, 2014* (pp. 355–360). Retrieved from http://www.terasoft.com.tw/conf/ismir2014/proceedings/T064_307_Paper.pdf

Poddar, A., Zangerle, E., & Yang, Y.-H. (2018). nowplaying-RS: A New Benchmark Dataset for Building Context-Aware Music Recommender Systems. doi:10.5281/zenodo.1422565

Poster, M. (2011). Introduction. *Into the Universe of Technical Images*.

Price, R., & Rebelo, P. (2008). Database and mapping design for audiovisual prepared radio set installation. In *Proceedings of the international conference on new interfaces for musical expression* (pp. 311–314). Genoa, Italy. Retrieved from http://www.nime.org/proceedings/2008/nime2008_311.pdf

Proutskova, P., Rhodes, C., Wiggins, G. A., & Crawford, T. (2012). Breathy or resonant - A controlled and curated dataset for phonation mode detection in singing. In F. Gouyon, P. Herrera, L. G. Martins, & M. Müller (Eds.), *Proceedings of the 13th international society for music information retrieval conference, ISMIR 2012, mosteiro s.bento da vitória, porto, portugal, october 8-12, 2012* (pp. 589–594). FEUP Edições. Retrieved from http://ismir2012.ismir.net/event/papers/589-ismir-2012.pdf

Puckette, M. (1986). Interprocess communication and timing in real-time computer music performance. In *Proceedings of the 1986 international computer music conference, ICMC 1986, den haag, the netherlands, october 20-24, 1986*, Michigan Publishing. Retrieved from http://hdl.handle.net/2027/spo.bbp2372.1986.008

Puckette, M. (1991). Something digital. *Computer Music Journal*, *15*(4), 65–69. Retrieved from http://www.jstor.org/stable/3681075

Puckette, M. (2002a). Max at seventeen. *Computer Music Journal*, *26*(4), 31–43. doi:10.1162/014892602320991356

Puckette, M. (2002b). Using pd as a score language. In *Proceedings of the 2002 international computer music conference, ICMC 2002, gothenburg, sweden, september 16-21, 2002*, Michigan Publishing. Retrieved from http://hdl.handle.net/2027/spo.bbp2372.2002.038

Puckette, M. (2007). On timbre stamps and other frequency-domain filters. In *ICMC*, Michigan Publishing.

Puckette, M. S. (1997). Pure data. In *Proceedings of the international computer music conference, ICMC 1997*, Michigan Publishing. Retrieved from https://quod.lib.umich.edu/i/icmc/bbp2372.1997.060/1

Puckette, M., Vercoe, B., & Stautner, J. P. (1981). A real-time music 11 emulator. In *Proceedings of the 1981 international computer music conference, ICMC 1981, denton, texas, usa, 1981*, Michigan Publishing. Retrieved from http://hdl.handle.net/2027/spo.bbp2372.1981.036

Queiroz, M., & Yoshimura, G. J. (2018). Relative DTW Embedding for Binary Classification of Audio Data. doi:10.5281/zenodo.1422585

Roads, C. [C.], & Mathews, M. (1980). Interview with max mathews. *Computer Music Journal*, *4*(4), 15–22. Retrieved from http://www.jstor.org/stable/3679463

Roads, C. [Curtis]. (2001). *Microsound*. MIT Press.

Roberts, C., Wright, M., Kuchera-Morin, J., & Höllerer, T. (2014). Rapid creation and publication of digital musical instruments. In *Proceedings of the international conference on new interfaces for musical expression* (pp. 239–242). London, United Kingdom: Goldsmiths, University of London. Retrieved from http://www.nime.org/proceedings/2014/nime2014_373.pdf

Rodet, X., Barrière, J., Cointe, P., & Potard, Y. (1982). The CHANT project: Modelization and production, an environment for composers including the FORMES language for describing and controlling sound and musical processes. In *Proceedings of the 1982 international computer music conference, ICMC 1982, venice, italy, september 27 - october 1, 1982*, Michigan Publishing. Retrieved from http://hdl.handle.net/2027/spo.bbp2372.1982.035

Rodet, X., Depalle, P., & Poirot, G. (1988). Diphone sound synthesis based on spectral envelopes and harmonic/noise excitation functions. In *Proceedings of the 1988 international computer music conference, ICMC 1988, cologne, germany, september 20-25, 1988*, Michigan Publishing. Retrieved from http://hdl.handle.net/2027/spo.bbp2372.1988.033

Rodet, X., & Lefèvre, A. (1996). Macintosh graphical interface and improvements to generalized diphone control and synthesis. In *Proceedings of the 1996 international computer music conference, ICMC 1996, hong kong, august 19-24, 1996*, Michigan Publishing. Retrieved from http://hdl.handle.net/2027/spo.bbp2372.1996.102

Rodet, X., & Lefèvre, A. (1997). The diphone program: New features, new synthesis methods and experience of musical use. In *Proceedings of the 1997 international computer music conference, ICMC 1997, thessaloniki, greece, september 25-30, 1997*, Michigan Publishing. Retrieved from http://hdl.handle.net/2027/spo.bbp2372.1997.111

Rosenboom, D., & Polansky, L. (1985). HMSL (hierarchical music specification language): A real-time environment for formal, perceptual and compositional experimentation. In *Proceedings of the 1985 international computer music conference, ICMC 1985, burnaby, bc, canada, august 19-22, 1985*, Michigan Publishing. Retrieved from http://hdl.handle.net/2027/spo.bbp2372.1985.039

Rossiter, D., & Ng, W.-Y. (1996). A system for the musical investigation and expression of levels of self-similarity in an arbitrary data stream. In *Proceedings of the international computer music conference, ICMC 1996*, Michigan Publishing. Retrieved from https://quod.lib.umich.edu/i/icmc/bbp2372.1996.085/1

Rowe, R. (1992). *Interactive music systems: Machine listening and composing*. Cambridge, MA, USA: MIT Press. Retrieved from https://wp.nyu.edu/robert_rowe/text/interactive-music-systems-1993

Rowe, R., Garton, B., Desain, P., Honing, H., Dannenberg, R., Jacobs, D., ... Lewis, G. (1993). Editor's notes: Putting max in perspective. *Computer Music Journal*, *17*(2), 3–11. Retrieved from http://www.jstor.org/stable/3680864

Sanden, C., Befus, C. R., & Zahng, J. (2010). Perception based multi-genre labeling on music data. In *Proceedings of the international computer music conference, ICMC 2010*, Michigan Publishing. Retrieved from https://quod.lib.umich.edu/i/icmc/bbp2372.2010.003/1

Sapp, C. S. (2005). Online database of scores in the humdrum file format. In *ISMIR 2005, 6th international conference on music information retrieval, london, uk, 11-15 september 2005, proceedings* (pp. 664–665). Retrieved from http://ismir2005.ismir.net/proceedings/3123.pdf

Scaletti, C. A. (1987). Kyma: An object-oriented language for music composition. In *Proceedings of the 1987 international computer music conference, ICMC 1987, champaign/urbana, illinois, usa, august 23-26, 1987*, Michigan Publishing. Retrieved from http://hdl.handle.net/2027/spo.bbp2372.1987.007

Schlei, K., & Yoshikane, R. (2016). The things of shapes: Waveform generation using 3d vertex data. In *Proceedings of the international computer music conference, ICMC 2016*, Michigan Publishing. Retrieved from https://quod.lib.umich.edu/i/icmc/bbp2372.2016.056/1

Schloss, W., Driessen, A., & Peter, F. (2001). Towards a virtual membrane: New algorithms and technology for analyzing gestural data. In *Proceedings of the international computer music conference, ICMC 2001*, Michigan Publishing. Retrieved from https://quod.lib.umich.edu/i/icmc/bbp2372.2001.103/1

Schmeder, A. (2009). Efficient gesture storage and retrieval for multiple applications using a relational data model of open sound control. In *Proceedings of the international computer music conference, ICMC 2009*, Michigan Publishing. Retrieved from https://quod.lib.umich.edu/i/icmc/bbp2372.2009.005/1

Schöner, B., Cooper, C., Douglas, C., & Gershenfeld, N. (1998). Data-driven modeling and synthesis of acoustical instruments. In *Proceedings of the 1998 international computer music*

conference, ICMC 1998, ann arbor, michigan, usa, october 1-6, 1998, Michigan Publishing. Retrieved from http://hdl.handle.net/2027/spo.bbp2372.1998.265

Schwarz, D. (2000). A system for data-driven concatenative sound synthesis. In *Proceedings of the cost g-6 conference on digital audio effects (dafx-00), verona, italy, december 7-9*.

Schwarz, D. (2003). New developments in data-driven concatenative sound synthesis. In *Proceedings of the international computer music conference, ICMC 2003*, Michigan Publishing. Retrieved from https://quod.lib.umich.edu/i/icmc/bbp2372.2003.099/1

Schwarz, D. (2006a). Concatenative sound synthesis: The early years. *Journal of New Music Research*, *35*, 3–22. doi:10.1080/09298210600696857

Schwarz, D. (2006b). Real-time corpus-based concatenative synthesis with catart. (pp. 18–21).

Schwarz, D. (2012). The sound space as musical instrument: Playing corpus-based concatenative synthesis. In *Proceedings of the international conference on new interfaces for musical expression*, Ann Arbor, Michigan: University of Michigan. Retrieved from http://www.nime.org/proceedings/2012/nime2012_120.pdf

Schwarz, D., & Schnell, N. (2009). Sound Search by Content-based Navigation in Large Databases. doi:10.5281/zenodo.849679

Selfridge-Field, E. (Ed.). (1997a). *Beyond midi: The handbook of musical codes*. Cambridge, MA, USA: MIT Press.

Selfridge-Field, E. (1997b). The score music publishing system. *SCORE*. From Beyond MIDI, The Handbook of Musical Codes. Retrieved from http://scoremus.com/score.html

Serafin, S., Smith, J., III, O., Thornburg, H., Mazzella, F., Tellier, A., & Thonier, G. (2001). Data driven identification and computer animation of bowed string model. In *Proceedings of the international computer music conference, ICMC 2001*, Michigan Publishing. Retrieved from https://quod.lib.umich.edu/i/icmc/bbp2372.2001.071/1

Serizel, R., Bisot, V., Essid, S., & Richard, G. (2016). Machine listening techniques as a complement to video image analysis in forensics. In *IEEE International Conference on Image Processing* (pp. 948–952). doi:10.1109/ICIP.2016.7532497

Shanks, D., & W.jun. Wrench, J. (1962). Calculation of pi to 100,000 decimals. *Mathematics of Computation*, *16*. doi:10.2307/2003813

Silberschatz, A., Stonebraker, M., & Ullman, J. (1995). *Database research: Achievements and opportunities into the 21st century* (Technical Report No. 1995-15). Stanford InfoLab. Stanford InfoLab. Retrieved from http://ilpubs.stanford.edu:8090/81/

Simonelli, L., Delgadino, M., & Halac, F. C. (2017). Hearing the self: A spectral experience. For 2 PS3 Eyecams, multichannel audio, screens and participants. 1440 minutes. Xuhui Art Museum, Shanghai, China: International Computer Music Conference.

Skinner, R. (1990a). Music software. *Notes*, *46*(3), 660–684. Retrieved from http://www.jstor.org/stable/941442

Skinner, R. (1990b). Music software. *Notes*, *47*(1), 91–101. Retrieved from http://www.jstor.org/stable/940555

Smith, J. B. L., Burgoyne, J. A., Fujinaga, I., Roure, D. D., & Downie, J. S. (2011). Design and creation of a large-scale database of structural annotations. In A. Klapuri & C. Leider (Eds.), *Proceedings of the 12th international society for music information retrieval conference,*

ISMIR 2011, miami, florida, usa, october 24-28, 2011 (pp. 555–560). University of Miami. Retrieved from http://ismir2011.ismir.net/papers/PS4-14.pdf

Smith, L. (1972). Score: A musician's approach to computer music. *Journal of the Audio Engineering Society*, *20*(1), 7–14. Retrieved from https://ccrma.stanford.edu/~aj/archives/docs/all/649.pdf

Solomos, M. (2005). An introduction to horacio vaggione musical-theoretical thought. *Contemporary Music Review*, *25*(4), 311–326. Retrieved from https://hal.archives-ouvertes.fr/hal-00770212

Sterne, J. (2003). *The audible past : Cultural origins of sound reproduction*. Duke University Press. Retrieved from https://ebookcentral-proquest-com.proxy.library.nyu.edu/lib/nyulibrary-ebooks/detail.action?docID=1167834

Sterne, J. (2012). *Mp3: The meaning of a format*. Duke University Press.

Stoller, D., Ewert, S., & Dixon, S. (2017). Adversarial semi-supervised audio source separation applied to singing voice extraction. *CoRR*, *abs/1711.00048*. arXiv: 1711.00048. Retrieved from http://arxiv.org/abs/1711.00048

Sturm, B. (2004). Matconcat: An application for exploring concatenative sound synthesis using matlab.

Sturm, B. L. (2002). Water music: Sonification of ocean buoy spectral data. In *Proceedings of the international computer music conference, ICMC 2002*, Michigan Publishing. Retrieved from https://quod.lib.umich.edu/i/icmc/bbp2372.2002.056/1

Szendy, P. (2008). *Listen: A history of our ears*. Fordham University. Retrieved from https://www.jstor.org/stable/j.ctt13x002m

Taylor, B., Allison, J., Conlin, W., Oh, Y., & Holmes, D. (2014). Simplified expressive mobile development with nexusui, nexusup, and nexusdrop. In *Proceedings of the international conference on new interfaces for musical expression* (pp. 257–262). London, United Kingdom: Goldsmiths, University of London. Retrieved from http://www.nime.org/proceedings/2014/nime2014_480.pdf

Thiebaut, J.-B., Bello, J., & Schwarz, D. (2007). How musical are images? from sound representation to image sonification: An eco systemic approach.

Truax, B. D. (1973). The computer composition: Sound synthesis programs pod4, pod5 and pod6. *Sonological Reports*, *2*.

Truax, B. D. (1976). A comunicational approach to computer sound programs. *Journal of Music Theory*, *20*(2), 227–300.

Truax, B. D. (1980). The inverse relation between generality and strength in computer music programs. *Interface*, *9*, 49–57.

Tzanetakis, G. [G.], & Cook, P. [P.]. (2002). Musical genre classification of audio signals. *IEEE Transactions on Speech and Audio Processing*, *10*(5), 293–302. doi:10.1109/TSA.2002.800560

Tzanetakis, G. [George], & Cook, P. [Perry]. (2000). Marsyas: A framework for audio analysis. *Organised Sound*, *4*(3), 169–175.

Vaggione, H. (1993). Determinism and the false collective about models of time in early computer-aided composition. *Contemporary Music Review*, *7*(2).

Vaggione, H. (2001). Some ontological remarks about music composition processes. *Computer Music Journal*, *25*(1), 54–61.

Valle, A., & Sanfilippo, D. (2012). Towards a typology of feedback systems. In *Proceedings of the international computer music conference, ICMC 2012*. Retrieved from https://quod.lib.umich.edu/i/icmc/bbp2372.2012.006

van Eck, C. (2013). *Between air and electricity: Microphones and loudspeakers as musical instruments* (Doctoral dissertation, Leiden University).

Varese, E. (2004). The liberation of sound. *Audio Culture: Readings in Modern Music*.

Vercoe, B. (1984). The synthetic performer in the context of live performance.

Vesna, V. (2007). *Database aesthetics: Art in the age of information overflow*. University of Minnesota Press.

Vicinanza, D. (2006). A Java Framework for Data Sonification and 3D Graphic Rendering. doi:10.5281/zenodo.849321

Vigliensoni, G., & Fujinaga, I. (2017). The music listening histories dataset. In S. J. Cunningham, Z. Duan, X. Hu, & D. Turnbull (Eds.), *Proceedings of the 18th international society for music information retrieval conference, ISMIR 2017, suzhou, china, october 23-27, 2017* (pp. 96–102). Retrieved from https://ismir2017.smcnus.org/wp-content/uploads/2017/10/180_Paper.pdf

Vinet, H. (2005). The semantic hifi project. In *Proceedings of the 2005 international computer music conference, ICMC 2005, barcelona, spain, september 4-10, 2005*, Michigan Publishing. Retrieved from http://hdl.handle.net/2027/spo.bbp2372.2005.171

Vinet, H., Herrera, P., & Pachet, F. (2002a). The CUIDADO project. In *ISMIR 2002, 3rd international conference on music information retrieval, paris, france, october 13-17, 2002, proceedings*. Retrieved from http://ismir2002.ismir.net/proceedings/02-FP06-3.pdf

Vinet, H., Herrera, P., & Pachet, F. (2002b). The CUIDADO project: New applications based on audio and music content description. In *Proceedings of the 2002 international computer music conference, ICMC 2002, gothenburg, sweden, september 16-21, 2002*, Michigan Publishing. Retrieved from http://hdl.handle.net/2027/spo.bbp2372.2002.093

Visi, F., Caramiaux, B., Mcloughlin, M., & Miranda, E. (2017). A knowledge-based, data-driven method for action-sound mapping. In *Proceedings of the international conference on new interfaces for musical expression* (pp. 231–236). Copenhagen, Denmark: Aalborg University Copenhagen. Retrieved from http://www.nime.org/proceedings/2017/nime2017_paper0043.pdf

Vogl, R., & Knees, P. (2017). An intelligent drum machine for electronic dance music production and performance. In *Proceedings of the international conference on new interfaces for musical expression* (pp. 251–256). Copenhagen, Denmark: Aalborg University Copenhagen. Retrieved from http://www.nime.org/proceedings/2017/nime2017_paper0047.pdf

Vogt, K., Pirro, D., Rumori, M., & Hoeldrich, R. (2012). Sounds of simulations: Data listening space. In *Proceedings of the international computer music conference, ICMC 2012*, Michigan Publishing. Retrieved from https://quod.lib.umich.edu/i/icmc/bbp2372.2012.096/1

von Neumann, J., & Burks, A. (1946). Preliminary discussion of the logical design of an electronic computing instrument. *Engineering, College of - Technical Reports*. Retrieved from https://library.ias.edu/files/Prelim_Disc_Logical_Design.pdf

Walker, B. N., & Cothran, J. T. (2003). ICAD 2004: The 13th meeting of the international conference on auditory display, boston, ma, usa, 6-9 july 2003, proceedings. In S. Barrass & P. Vickers (Eds.), International Community for Auditory Display.

Walker, B. N., & Nees, M. A. (2011). Theory of sonification. In T. Hermann, A. Hunt, & J. G. Neuhoff (Eds.), *The sonification handbook* (Chap. 2, pp. 9–39). Berlin, Germany: Logos Publishing House. Retrieved from http://sonification.de/handbook/chapters/chapter2/

Wang, G., & Cook, P. R. (2003). Chuck: A concurrent, on-the-fly, audio programming language. In *Proceedings of the 2003 international computer music conference, ICMC 2003, singapore, september 29 - october 4, 2003*, Michigan Publishing. Retrieved from http://hdl.handle.net/2027/spo.bbp2372.2003.055

Wang, X., & Haque, S. A. (2017). Classical music clustering based on acoustic features. *CoRR*, *abs/1706.08928*. arXiv: 1706.08928. Retrieved from http://arxiv.org/abs/1706.08928

Weinbren, G. (2007). Ocean, database, recut. *Database Aesthetics: Art in the Age of Information Overflow*.

Whalley, I. (2014). Broadening telematic electroacoustic music by affective rendering and embodied real-time data sonification. In *Proceedings of the international computer music conference, ICMC 2014*, Michigan Publishing. Retrieved from https://quod.lib.umich.edu/i/icmc/bbp2372.2014.046/1

Wilkins, J., Seetharaman, P., Wahl, A., & Pardo, B. (2018). Vocalset: A singing voice dataset. In E. Gómez, X. Hu, E. Humphrey, & E. Benetos (Eds.), *Proceedings of the 19th international society for music information retrieval conference, ISMIR 2018, paris, france, september 23-27, 2018* (pp. 468–474). Retrieved from http://ismir2018.ircam.fr/doc/pdfs/114_Paper.pdf

Wilson, C. M., & Lodha, S. K. (1996). Listen: A data sonification toolkit. In *Proceedings of the 3rd international conference on auditory display (icad1996), palo alto, california, november 4-6, 1996. eds.: S. frysinger; g. kramer. international community for auditory display, 1996.* Georgia Institute of Technology. Retrieved from http://hdl.handle.net/1853/50809

Worrall, D., Bylstra, M., Barrass, S., & Dean, R. (2007). ICAD 2004: The 13th meeting of the international conference on auditory display, montreal, canada, june 26-29 2007, proceedings. In S. Barrass & P. Vickers (Eds.), International Community for Auditory Display.

Wüst, O., & Celma, Ò. (2004). An MPEG-7 database system and application for content-based management and retrieval of music. In *ISMIR 2004, 5th international conference on music information retrieval, barcelona, spain, october 10-14, 2004, proceedings*. Retrieved from http://ismir2004.ismir.net/proceedings/p010-page-48-paper227.pdf

Xambo, A., Roma, G., Herrera, P., & Laney, R. (2012). Factors in Human Recognition of Timbre Lexicons Generated by Data Clustering. doi:10.5281/zenodo.850102

Xambo, A., Roma, G., Lerch, A., Barthet, M., & Fazekas, G. (2018). Live repurposing of sounds: Mir explorations with personal and crowdsourced databases. In T. M. Luke Dahl Douglas Bowman (Ed.), *Proceedings of the international conference on new interfaces for musical*

expression (pp. 364–369). Blacksburg, Virginia, USA: Virginia Tech. Retrieved from http://www.nime.org/proceedings/2018/nime2018_paper0081.pdf

Xenakis, I. (1992). *Formalized music: Thought and mathematics in music*. Pendragon Revised Edition.

Xi, Q., Bittner, R. M., Pauwels, J., Ye, X., & Bello, J. P. (2018). Guitarset: A dataset for guitar transcription. In E. Gómez, X. Hu, E. Humphrey, & E. Benetos (Eds.), *Proceedings of the 19th international society for music information retrieval conference, ISMIR 2018, paris, france, september 23-27, 2018* (pp. 453–460). Retrieved from http://ismir2018.ircam.fr/doc/pdfs/188_Paper.pdf

Xu, Y., Zang, C., & Yang, J. (2005). Semi-supervised classification of musical genre using multi-view features. In *Proceedings of the 2005 international computer music conference, ICMC 2005, barcelona, spain, september 4-10, 2005*, Michigan Publishing. Retrieved from http://hdl.handle.net/2027/spo.bbp2372.2005.020

Yang, C. (2001). *Music database retrieval based on spectral similarity* (Technical Report No. 2001-14). Stanford InfoLab. Stanford. Retrieved from http://ilpubs.stanford.edu:8090/489/

Yeh, C., Bogaards, N., & Röbel, A. (2007). Synthesized polyphonic music database with verifiable ground truth for multiple F0 estimation. In S. Dixon, D. Bainbridge, & R. Typke (Eds.), *Proceedings of the 8th international conference on music information retrieval, ISMIR 2007, vienna, austria, september 23-27, 2007* (pp. 393–398). Austrian Computer Society. Retrieved from http://ismir2007.ismir.net/proceedings/ISMIR2007_p393_yeh.pdf

Yeo, W. S., & Berger, J. (2005). Application of image sonification methods to music. In *Proceedings of the 2005 international computer music conference, ICMC 2005, barcelona, spain, september 4-10, 2005*, Michigan Publishing. Retrieved from http://hdl.handle.net/2027/spo.bbp2372.2005.036

Yeo, W., Berger, S., Lee, J., & Zune. (2004). Sonart: A framework for data sonification, visualization and networked multimedia applications. In *Proceedings of the international computer music conference, ICMC 2004*, Michigan Publishing. Retrieved from https://quod.lib.umich.edu/i/icmc/bbp2372.2004.128/1

Yolk, A., Wiering, F., & van Kranenburg, P. (2011). Unfolding the potential of computational musicology. In *Problems and possibilities of computational humanities - 13th IFIP iwra 2011 ifip wgs.l — international conference on informatics and semiotics in organisations, ICISO 2011, netherlands, july 4-6, 2011. proceedings* (pp. 137–144).

Young, D., & Deshmane, A. (2007). Bowstroke database : A web-accessible archive of violin bowing data. In *Proceedings of the international conference on new interfaces for musical expression* (pp. 352–357). New York City, NY, United States. Retrieved from http://www.nime.org/proceedings/2007/nime2007_352.pdf

Zicarelli, D. (1998). An extensible real-time signal processing environment for max. In *Proceedings of the 1998 international computer music conference, ICMC 1998, ann arbor, michigan, usa, october 1-6, 1998*, Michigan Publishing. Retrieved from http://hdl.handle.net/2027/spo.bbp2372.1998.274