7-9-2021 12:45 PM

Music for Self-Attention

Jeffrey A T Lupker, *The University of Western Ontario*

Supervisor: Frehner, Paul, *The University of Western Ontario*
: Turkel, William J., *The University of Western Ontario*

A thesis submitted in partial fulfillment of the requirements for the Doctor of Philosophy degree in Music

Recommended Citation

Lupker, Jeffrey A T, "Music for Self-Attention" (2021). *Electronic Thesis and Dissertation Repository*. 7898.
https://ir.lib.uwo.ca/etd/7898

Abstract

Creating an artificial intelligence aid for music composers requires a practical and modular approach that allows the composer to manipulate the technology as needed in the search for new sounds and ideas. Many existing approaches fail to capture the interest of composers as they are limited beyond their demonstrative purposes or allow minimal interaction with the composer. Score-Transformer (ST) demonstrates a practical integration of artificial intelligence to aid in the creation of new music by working seamlessly alongside any popular notation software. Furthermore, ST can be trained by the user with additional works (including their own compositions), fine-tuning it and minimizing the risk of the software becoming outdated or impractical for continued use. ST was used in the creation of my dissertation piece, *Music for Self-Attention.*

Music for Self-Attention features an innovative algorithmic approach to traditional compositional methods by demonstrating the benefits of using deep learning in music composition to aid with certain pitch and rhythmic decisions. These artificially generated decisions were not intended to fully remove the human element from composing but rather to work in tandem with the composer, in this case, myself. The piece lasts approximately 23 minutes and loosely follows a form of theme and variations in reverse. This steady process gradually deconstructs a series of variations which were each initially generated by artificial means until the final movement—containing no artificial intervention—is revealed. This is not a theme and variations in the usual way as the listener won't hear the theme in each variation. Each movement reflects different processes involved in the training and deployment of artificially intelligent software. *Music for Self-Attention* is meant to demonstrate a symbiotic relationship that can exist between artificial and human creativity in music composition.

Keywords

Music composition, artificial intelligence, deep learning, string quartet, natural language processing

Summary for Lay Audience

Artificial intelligence has more recently been permeating our everyday lives through applications such as personal assistants (Apple's Siri or Amazon Alexa), search, video, and product recommendation (Google, Youtube, etc.) and has even been used to beat human champions at popular games such as Go and Chess. Therefore, it seemed as though music composition could similarly benefit from artificially intelligent software to aid in the creation of new music. Some attempts had been made in the past, but many failed to capture the interest of composers as they could be limited beyond demonstrative purposes or allow minimal interaction with the composer. Score-Transformer (ST) is an artificially intelligent software that can be used to aid in the creation of new music by working seamlessly alongside any popular notation software. Furthermore, ST can be fed additional works (including the user's own compositions) in order to update it, minimizing the risk of the software becoming outdated or impractical for continued use. ST was used in the creation of my dissertation piece, *Music for Self-Attention.*

Music for Self-Attention features an innovative approach to traditional compositional methods by demonstrating the benefits of artificial intelligence used to aid with certain pitch and rhythmic decisions when developing the score. The use of artificial intelligence was not intended to replace the composer in creating new music, but rather to work alongside the composer and make suggestions. The piece, for string quartet, is in six movements and lasts approximately 23 minutes. Each movement of the piece features music that was initially generated by artificial means until the final movement which did not use Score-Transformer. *Music for Self-Attention* is meant to demonstrate a relationship that can exist between artificial and human creativity in music composition.

Acknowledgments

First, I would like to thank my girlfriend Beth Copeland, whose love and support throughout my PhD was integral to its success (especially during times of seemingly unfixable code breaks). I could not have completed this work without you.

Next to my parents Natalie Allen and Steve Lupker, who have edited countless papers, proposals and applications throughout my eleven years in university. Also, for inspiring me to pursue a career in academia.

To Dr. William J. Turkel I would like to thank you for all of the support during my degree and throughout my dissertation. I look forward to our continued research and co-publications together.

I would like to say a special thank you to my supervisor Dr. Paul Frehner for all of your help throughout the last five years of my degree. I am very grateful for all of the time you have spent helping me with my research, writing recommendation letters and all of the guidance you have provided me along the way.

Table of Contents

List of Figures

List of Musical Examples

List of Appendices

Part 1

1 Background

1.1 Non-Musical Systems Applied to Music

1.1.1 Introduction

The use of deep learning technologies in music composition is still relatively new, but the implementation of systems is not. Furthermore, the use of non-musical systems later adapted into a musical context has also been quite common. The inclusion of systems as a pre-compositional technique (sometimes referred to as algorithmic composition) has historical precedents that can be traced back to the birth of early music treatises and perhaps earlier[1]. These early treatises detailed a list of rules and conventions deemed at the time to be common practice and, as such, a composer's pre-compositional work might strictly adhere to them. Perhaps this is what medieval theorist and music treatise writer Guido d'Arezzo was building upon as he later developed one of the first known algorithmic compositional approaches which converted text into music[2,3]. Gerhard Nierhaus describes part of Arezzo's method:

> "Letters, syllables and components of a verse are mapped on tone pitches and melodic phrases (neumes), whereas groups of neumes are separated by caesurae. On the level of groups of neumes, the caesurae correspond to breathing pauses and can also be found in smaller groups in the form of pauses or held notes. The vowels in the text can be mapped on different tone pitches."[4]

[1] Jeffrey A. T. Lupker. "Generative Music and Algorithmic Composition." In P. Frana and M. Klein (Eds.), *Encyclopedia of Artificial Intelligence: The Past, Present, and Future of AI*. (Santa Barbara: ABC-CLIO, 2021), p.168-170.

[2] Gerhard Nierhaus. *Algorithmic Composition*. (Mörlenbach: SpringerWienNewYork, 2009).

[3] Gareth Loy, *Musimathics: The Mathematical Foundations of Music, Volume 1.* (Cambridge: MIT Press, 2011), p.287.

[4] Nierhaus, p.21.

Given their sequential and hierarchical nature, systems designed for natural language are well-suited to musical application, but they are not the only area outside of music from which composers could effectively borrow tools. Notable examples from the 20th century are Schoenberg's use of the twelve-tone matrix or Xenakis' stochastic music. Both composers repurpose non-musical systems (in this case formalized mathematics) "out of necessity" in an effort to expand upon their own compositional techniques and to break away from the conventions of their contemporaries[5,6]. Both of these approaches reflect an ideal of algorithmic composition in which some aspect of human/composer intervention is minimized during the compositional process. However, this does not require the system to fully generate a piece of music autonomously (fully removing human intervention) but allows for minimal to significant aspects of the work to be decided by other means. My own compositional process outlined throughout this document will demonstrate a shared human and machine approach in the application of deep learning technology. Schoenberg and Xenakis' use of methods borrowed from formalized mathematics in their own compositions are detailed below.

1.1.2 Arnold Schoenberg

"The method of composing with twelve tones grew out of a necessity."[7]

As harmony and harmonic progressions dictated by a single root were becoming less apparent in the work of Schoenberg's colleagues, he conceived the necessity of drawing pitch material from a twelve-tone matrix[8]. Convinced that the ear was becoming more accustomed to dissonance, Schoenberg sought a means of freeing dissonance from the conventional methods of treatment, preparing and resolving, which he entitled "the

[5] Arnold Schoenberg. *Style and Idea: Selected Writings of Arnold Schoenberg: 60th Anniversary Ed.* (Berkeley: University of California Press, 2010), p.216.

[6] Edward A. Shanken, editor. *Systems*. (Cambridge, MA: The MIT Press, 2015), p.155.

[7] Schoenberg, p.216

[8] Ibid.

emancipation of dissonances”[9]. Schoenberg himself described his twelve-tone method as a “tool of composition” to be used amongst all the others a composer requires such as knowledge of form, expression, themes, and so on, and not as the sole means for creating music. As Schoenberg’s development and use of twelve-tone rows is well-documented, I will instead provide a brief overview of the area of mathematics from which it derives.

Schoenberg’s use of twelve-tone rows comes from an area of mathematics called combinatorics, which is “the study of how sets can be combined in patterns”[10]. Many combinatoric principles are utilized when deciding upon pitch content taken from a twelve-tone matrix. For example, permutations are an area of combinatorics which deal with the unique orderings of any given set, and its musical counterpart can offer a composer a simple method of deducing all unique orderings available to them when deciding upon a tone row. A twelve-tone matrix offers a visual display of a subset of the possible permutations of any given tone row. While the first row contains an original 12-note row, the next 11 rows (or 11 further permutations of the original) contain transposed versions of the original tone row. To plot each transposed row, one must first find the inversion of the original tone row and plug these pitches into the first column (top to bottom) of the matrix then copy the intervals between each note of the original row to finish each transposed row. A further 36 permutations are visible on the matrix when the retrograde, inversion and retrograde inversion variants are considered.

Another combinatoric principle involves dividing or partitioning a set into smaller subsets. Partitioning a set (or any of its permutations found in the twelve-tone matrix) allows a composer continued freedom in selecting pitch content not fully removed from the ordering determined by their initial row. For example, if one starts with a tone row of 12 notes and partitions it into 2 groups of 6, the similarly ordered first group could now be paired with a newly ordered complimentary set (a set in which no notes are repeated form the first group). Figure 1 shows how reordering the complementary set can lead to

[9] Schoenberg, p.216.

[10] Loy, p.306.

720 permutations of the original tone row using the formula *n*! (for example, *5*! = 5 x 4 x 3 x 2 x 1), providing the composer with many new options for continuing their piece. As the first group remains the same, the ear presumably should not find this reordering of the second set too jarring if it should occur within the same piece.

The use of combinatorics in music composition allowed Schoenberg to avoid a single-note tonic by offering a mathematical approach to grouping the twelve pitches available to him in many different balanced combinations. By applying permutation, partition and other principles, Schoenberg moved beyond conventional pitch orderings such as the various church modes or the major and minor scales. His work would greatly influence many of his contemporaries and future composers such as Webern, Messiaen, Feldman and others who incorporated or even expanded his techniques to non-pitch material such as rhythms, registers or note durations (Boulez, Babbitt, etc.). Even while drawing pitch content from his mathematics-based system, Schoenberg still noted the importance of drawing upon all his formal musical training to complement his algorithmic approach which was not intended as a means of fully automating the process of composing music.

$$P(n,r) = \frac{n!}{(n-r)!}$$

$$n = \left(\frac{n}{n_1, n_2, \ldots, n_r}\right)$$

$$P(n,n_2) = \frac{n!}{(n-n_2)!} = \frac{12!}{(12-6)!} = 720$$

Figure 1: Top row: Permutation formula. Middle row: Partition formula where *n* is the size of the set and *r* is the number of partitions. Bottom row: Formula to calculate the number of possible permutations of partition *n*$_2$, the second of two equal six-note partitions taken from the tone row.

1.1.3 Iannis Xenakis

> “The laws of the calculus of probabilities entered composition through musical necessity.”[11]

After World War II, Iannis Xenakis reacted to the growing total serialist movement as he believed that the complexities involved in serializing every aspect of music diminished the audibility of the processes themselves. In his mind, the now overly complex and intricate linear structures more closely resembled clouds of dense sound masses rather than complex linear polyphony[12,13]. Noticing a “contradiction between polyphonic linear systems and the heard result”[14], Xenakis sought a means of controlling dense sound masses through the introduction of probability, notably in the form of combinatory calculus which offered an “escape route” to overly complex linear structures in music[15]. Not only did he seek to distance himself from serialism but simultaneously was embracing and utilizing then novel technologies such as “statistical clouds of microevents” through thermodynamics processes, the physics of quantum mechanics and perhaps even more generally, the use of computers in his own compositions[16,17]. This marked an important step in music composition as composers began to utilize the power of computers to envision new methods of creating music. For Xenakis, the computer allowed him to calculate statistical methods which were too computationally intensive for a human to do by hand.

[11] Shanken, p.155.

[12] Curtis Roads. *Composing Electronic Music: A New Aesthetic*. (Oxford: Oxford University Press, 2015), p.169.

[13] Balint A. Varga. *Conversations with Iannis Xenakis*. (London, UK: Faber and Faber, 1996), p.54.

[14] Iannis Xenakis. *Formalized Music: Thought and Mathematics in Composition*. (Pentagon Press (Revised Edition), 1992), p.8.

[15] Loy, p.332.

[16] Curtis Roads. *Microsound*. (Cambridge, MA: The MIT Press, 2001), p.15.

[17] Loy, p.333.

Named by Xenakis himself, stochastic music repurposes principles of stochastic science, namely the law of large numbers, statistical thermodynamics, laws of rare events and different aleatory procedures[18]. In his book *Formalized Music*, Xenakis offers a multi-step process for applying stochastic methods in musical composition[19]. I will examine only a few of the principles in this list that best demonstrate the conversion of probabilistic mathematical principles into the musical domain. Akin to Schoenberg's method of applying combinatorics as a compositional technique, the most important aspect is to define musical events or properties which can be substituted for the variables in any given mathematical equation and the space in which operations or transformations can occur (an early step in Xenakis' list). For example, in defining sonic entities occurring from an orchestra (timbre/instrumental family, pitch, intensity and duration) as coordinates of a point (*M*), these points can be plotted along an axis (E_r) with a second axis being drawn at a right-angle constituting time (*t*). Thus, a two-dimensional space has been created through which transformations can be defined (Fig. 2). Xenakis would utilize this method in developing his piece *Pithoprakta*[20].

[18] Xenakis, 1992, p.8.

[19] Ibid, p.22.

[20] Iannis Xenakis. *Pithoprakta*. (London, UK: Boosey & Hawkes, 1967).

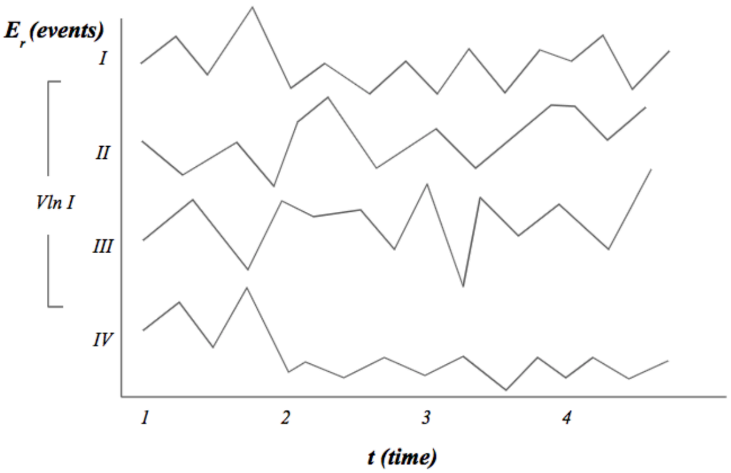


Figure 2: Example of a two-dimensional space in which glissandi events for four violins are plotted.

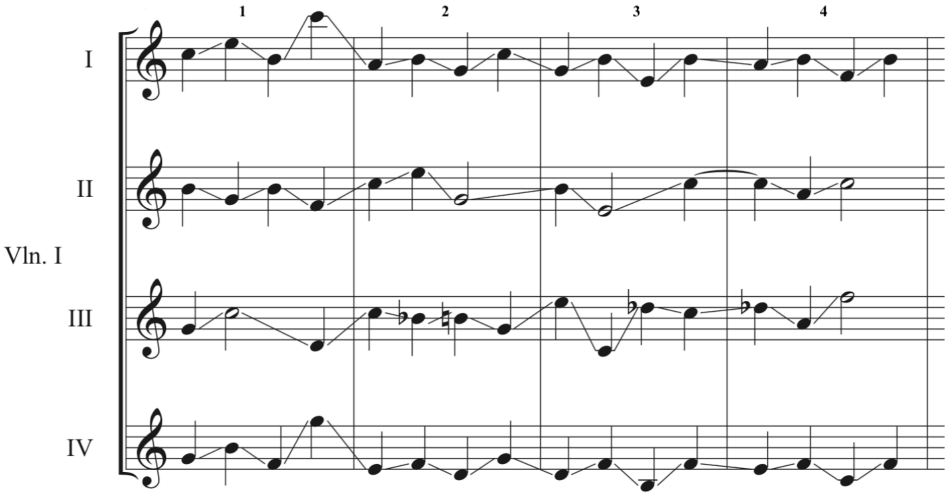


Figure 3: A standard notation representation of Figure 2.

In his piece *Achorripsis*[21], Xenakis uses Poisson's probability formula, the law for the appearances of rare random events (Figure 4), in order to derive a matrix of sonic events and their probability of occurring. Poisson's formula allowed Xenakis to achieve the "greatest amount of asymmetry while using the minimum constraints [(a given space

[21] Iannis Xenakis, *Achorripsis.* (Berlin: Bote & Bock, 1958).

of instruments and players)], casualties and rules” by which he could generate his composition through superficial density[22]. Thus, given some musical event, we can determine the probability of it occurring a certain number of times over the course of some time period using this formula and a predetermined “mean of successes” for how often we want the event to occur. I demonstrate this in Figure 5 by the following tables showing the probability distributions for a given event occurring *K* times (in this case 0-3 times) over some unknown time period with two different means of success, *n* = 0.2 and then *n* = 0.6. The result is an exponential decay in probability as K increases, showing how this law governs rare events and the diminishing likelihood of multiple events. By using this law to govern all decisions regarding musical material, Xenakis was able to control the seemingly aleatoric nature of sound clouds simply through the manipulation of the mean of successes variable. This constitutes a very clever repurposing of mathematical formulas for use within a musical context which extended upon ideas set forth by his contemporaries still working with serialist techniques.

$$P_k = \frac{n^k}{K!}e^{-n}$$

Figure 4: Poisson’s formula where *n* is the mean of the number of successes, *K* is the actual number of successes and *e* is Euler’s number (2.7183…).

***n* = 0.2**	***n* = 0.6**
$P_0 = 0.8187$	$P_0 = 0.5488$
$P_1 = 0.1637$	$P_1 = 0.3293$
$P_2 = 0.0163$	$P_2 = 0.0988$
$P_3 = 0.0011$	$P_3 = 0.0198$

Figure 5: Probability distribution for an example musical element occurring 0-3 times (*K*) over some time period.

[22] Xenakis, 1992, p.23.

1.1.4 Artificial Intelligence Systems in Music Composition[23]

Artificial intelligence (AI) systems in music include those based on generative grammar, knowledge-based systems, genetic algorithms and more recently, artificial neural networks. Generative grammar is a system of rules designed to describe natural languages, developed by Noam Chomsky and his colleagues. The rules rewrite hierarchically organized elements to describe a space of possible serial orderings of elements. Adapted to algorithmic composition, generative grammars can be used to output musical passages. An example of this, and similarly one of the earliest examples of computer-generated music can be found in Hiller and Isaacson's *Illiac Suite for String Quartet* (1957). This piece features four movements where a computer is programmed on a set of rules to generate different musical elements such a pitch, rhythm and dynamics. While generative grammar is used to control some rule-based musical element selections, so too are Markov chains, probability distributions and stochastic rules (akin to those discussed earlier in Xenakis' works)[24].

David Cope's software *Experiments in Musical Intelligence* (EMI) provides perhaps the most well-known use of a generative grammar in music composition. Furthermore, Cope himself is one of the most prolific users of artificial intelligence in music creation as his software EMI produced around 11,000 pieces of music. Initially developed to combat writer's block, Cope trained his software to convincingly write music in the style of many different composers such as Bach, Mozart and Chopin[25]. His method included three basic principles: analyze and deconstruct music into parts, find

[23] This section contains an excerpt adapted from my contribution "Generative Music and Algorithmic Composition" in Philip Frana and Michael Klein (eds.), *Encyclopedia of Artificial Intelligence: The Past, Present, and Future of AI*. (Santa Barbara: ABC-CLIO, 2021), p.169-170.

[24] Örjan Sandred, Mikael Laurson & Mika Kuuskankare. "Revisiting the Illiac Suite – a rule-based approach to stochastic processes." *Sonic Ideas/Ideas Sonicas*. (2009), p.42-46.

[25] David Cope, "Experiments in Musical Intelligence". Retrieved from *http://artsites.ucsc.edu/faculty/cope/experiments.htm.*

commonalities to retain style and finally recombine into new music (recombinancy)[26.] Simply put, his method was to create new music through logically recombining aspects of existing music, usually in the style of one composer. Some notable composers that he was able to emulate through this system are Beethoven, Bach, Chopin, Stravinsky and Vivaldi.

Other early types of artificial intelligence systems used in the 20th century to create music include knowledge-based systems and genetic algorithms. In knowledge-based systems, information regarding the type of music the composer wishes to imitate is encoded as a database of facts that can be drawn upon to create an artificial 'expert' that can be consulted by the composer. Genetic algorithms mimic the process of biological evolution and provide another method for algorithmic composition. A population of randomly created compositions is tested for closeness to the desired musical output, then artificial mechanisms modeled on those in nature are used to increase the probability that musically desirable traits will increase in subsequent generations. The composer interacts with the system allowing both computer and observer to generate new ideas.

More recent methods of AI-generated composition feature deep learning systems such as generative adversarial networks (GANs). GANs in music pit a generator, which creates new music based on knowledge of a compositional style, against a discriminator, which attempts to distinguish between the output of the generator and a human composer. Each time the generator fails, it receives new information until the discriminator cannot tell the difference between real and generated musical material. The repurposing of non-musical algorithms for musical purposes increasingly drives music in new and exciting directions.

1.2 Final Remarks

These brief examples of methodology pertaining to Schoenberg and Xenakis demonstrate non-musical systems of a mathematical nature used in musical composition.

[26] Ibid.

Furthermore, work by David Cope and researchers who adapted other non-musical artificial intelligence systems into music demonstrate earlier introductions of AI in music. Some of these earlier usages of AI in music involve systems such as generative grammars which originated in a natural language domain and are a precursor to my own work involving natural language systems in music. Many of these past AI systems that were used to create music required explicit instructions on how to write music. My own interests in applying AI technologies into music composition involve newer models which do not require this type of hand-coded information and instead "learn" by themselves from huge datasets. AI systems have been used to write music for purposes of experiment and demonstration but seem to have failed to capture the attention of serious composers besides a few individuals such as David Cope. Capturing the attention of composers is an important aspect of my work, which attempts to create software that could be used by composers beyond experimental works or single use compositions. Akin to Xenakis, I believe that current technologies can provide an excellent resource in reshaping the way we think about music composition. For this reason, I believe incorporating deep learning into music as a compositional tool might provide new insight into music creation.

Part 2

2 Score-Transformer

Section 2.1 will discuss methodology and application of my software Score-Transformer (ST). ST was used throughout the entire compositional process in order to demonstrate an innovative and practical use of the transformer model in the creation of new music. All code can be found online at https://github.com/lupks/score_transformer and is freely available with the MIT open-source license which permits both private and commercial modification and reuse.[27]

The terms 'artificial intelligence', 'machine learning' and 'deep learning' are all used throughout and a basic chart of the hierarchy of these terms can be found in Figure 6. ST features deep learning technologies where 'learning' refers to the ability to derive meaningful information from various representations of any given dataset and 'depth' refers to the number of layers within a particular learning algorithm[28]. The many uses of deep learning are ever-changing and developing at an extremely fast pace yet are an exciting prospect for the creation of new music.

[27] Any Score-Transformer source files modified from original free open-sourced resources have been cited within the code. They are also referenced and cited throughout this document.

[28] Jeffrey A. T. Lupker. "Deep Learning." In P. Frana and M. Klein (Eds.), *Encyclopedia of Artificial Intelligence: The Past, Present, and Future of AI*. (Santa Barbara: ABC-CLIO, 2021), p.112-114.

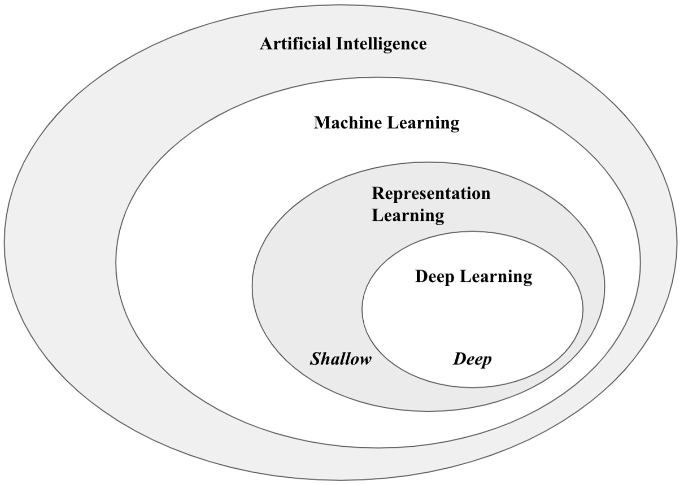


Figure 6: Hierarchy of artificial intelligence terms.

2.1 Introduction

The process of training machine learning models to learn to write MIDI music autonomously has slowly been generating more coherent and human-like examples, as these models become more adept at determining hierarchical and sequential patterns found in music. Models developed for natural language tasks have shown excellent promise when utilized in a musical domain as both language and music contain linear sequences of classes of elements that draw some of their meaning from the relationships those elements have with one another. Recent research applying the transformer model to musical tasks (a type of deep learning model which excels at parsing syntax, translating languages and generating documents[29]) has shown increased ability over previous model-types in autonomous MIDI music generation[30,31]. This is due in large part to the attention mechanism[32], which allows a model to "pay attention" to what comes before it in a

[29] Aurélien Géron. "*Hands-On Machine Learning with Scikit-Learn, Keras & Tensorflow: Concepts, Tools, and Techniques to Build Intelligent Systems* (2nd Ed.). (O'Reilly Media Inc., 2019), p. 549-562.

[30] Cheng-Zhi Anna Huang et al. "Music Transformer." *arXiv.org* (2018).

[31] Christine Payne. "MuseNet." *OpenAI.* (2019).

[32] Géron. (2019), p. 549.

sequence during output generation, providing context for its prediction of the next output in that sequence[33]. While artificial creative ability is subjective, here it will be defined as a model's understanding and reproduction of the elements in polyphonic music generation including more complex elements such as recurring motifs, phrases and underlying musical structures. While recent work demonstrates the application of state-of-the-art deep learning technologies to music[34,35], they also have limited practical abilities beyond their demonstrative purposes and often fail to capture the interest of music performers or composers. Furthermore, these models can be so large and unwieldy that a composer would require access to powerful computers in order to generate music fast enough to keep their attention. A model that takes more than 30 seconds to autonomously generate music is probably going to try the patience of a composer in the midst of a creative moment.

This section thus introduces Score-Transformer (ST), a modular and practical approach in applying a new variant of the transformer-decoder model to software in which a composer can work in tandem with an artificially intelligent assistant using their favourite notation software to create new ideas, reference past stylistic conventions or at times break writer's block[36]. ST is a model which was trained from scratch on an initial dataset of approximately 180,000 MIDI recordings of music but it can be further fine-tuned with new MIDI datasets of a user's choosing. A composer may wish to fine-tune the model on their own works, specialize in a specific style of music or just increase the dataset if they feel the output doesn't match their needs. During output generation, ST allows the composer full control over the parameters by which the predictive ability of the model can be manipulated to maximize creativity. An important distinction between

[33] Dzmitry Bahdanau et al., "Neural Machine Translation by Jointly Learning to Align and Translate". In *Proceedings of the Second International Conference on Learning Representations* (ICLR 2014), 2014.

[34] Huang et al., 2019.

[35] Payne, 2019.

[36] David Cope states his initial attempts at using AI for composition were for this exact reason. http://artsites.ucsc.edu/faculty/cope/experiments.htm

ST and past research is the capability to provide the model with previously unseen musical prompts (some number of measures) of any MIDI track size[37]. While examples of past work seem to bias the output by using prompts related to the model's own database to generate examples, ST allows any prompt to be used, regardless of whether or not it contains notes or chords that were previously seen by the model during training or fine-tuning.

2.2 Model

Training a model to write and score compelling MIDI music in any of the popular notation software (*Finale*, *Sibelius*, etc.) involves the following processes: data preprocessing, training and fine-tuning, and sampling. All training was accomplished using access to GPUs from cloud compute resources from Google's Colaboratory and by support provided by Compute Canada (www.computecanada.ca)[38]. After training, Score-Transformer can be run on standard laptop and desktop computers.

2.2.1 Data Preprocessing

As transformer models are generally used for natural language tasks, my initial research focused on the requirement for the encoding process to convert from a MIDI file to something understood by a deep learning model. Encoding a dataset for natural language, commonly called the corpus, involves the creation of a vocabulary set (also called the dictionary set) where unique words, characters or bytes, referred to as tokens[39], are listed and each given a number. For example, in a natural language model, anytime the word "apple" appears in the corpus, the model will instead convert that word into some positive integer (say 42) which will from then on only be associated with that same word. After a numbered vocabulary set of unique tokens has been created, each word in the

[37] MIDI tracks contain musical information for each instrument or stave. ST is not limited by amount of instruments found on a recording.

[38] Both provided access to Nvidia Tesla P100 and V100s which were capable of processing ST's training.

[39] Ian Goodfellow et al. *Deep Learning*. (Cambridge, MA: The MIT Press, 2016), p.448.

corpus can be converted to a number (which corresponds to a unique token). Thereafter, the corpus as a whole can be thought of as a series of vectors where each component of the vector is a token which in turn represents words, numbers or punctuation. This is important as a transformer-decoder model takes as its input a single continuous strand of tokens (or batches of them). A similar encoding system can likewise be used as a method of tokenizing MIDI event messages.

While monophonic music converts straightforwardly into such a vectorized form, polyphony adds some difficulties. Each stave or track from the MIDI file must become one sequential vector for the model to understand. Past research offers potential ways to accomplish this[40,41,42] but as part of the goal was to keep the model size as small as possible for use on standard commercial computers, a conversion method was required which kept the vocabulary size as low as possible. Beyond the computing resources needed for excessively large vocabulary sizes, a model learns by pattern recognition found within the data[43], so if the model tries to interpret patterns amongst data with very high dimensionality, the patterns can become harder to find. Training time thus becomes significantly longer and loss, a metric for determining how well the model is learning (see section 2.2.5), can become stuck at some undesirable point. Work by Oore et al.[44] provided a method whereby the vocabulary size after encoding could be limited to 388 entries (tokens), where each MIDI note-on (128 values), note-off (128-values), time-shift (100) and velocity (32 groups of 4 velocity values) events were listed. As I was not concerned with velocity since the goal was to notate the music, I could further reduce this to 357 elements by using the same velocity value for every note-on element that appears

[40] Huang et al., 2018.

[41] Payne, 2019.

[42] Sageev Oore et al. “This Time with Feeling: Learning Expressive Musical Performance.” *arXiv.org*. (2018).

[43] Lupker, 2021.

[44] Oore et al., 2018, p.13-14.

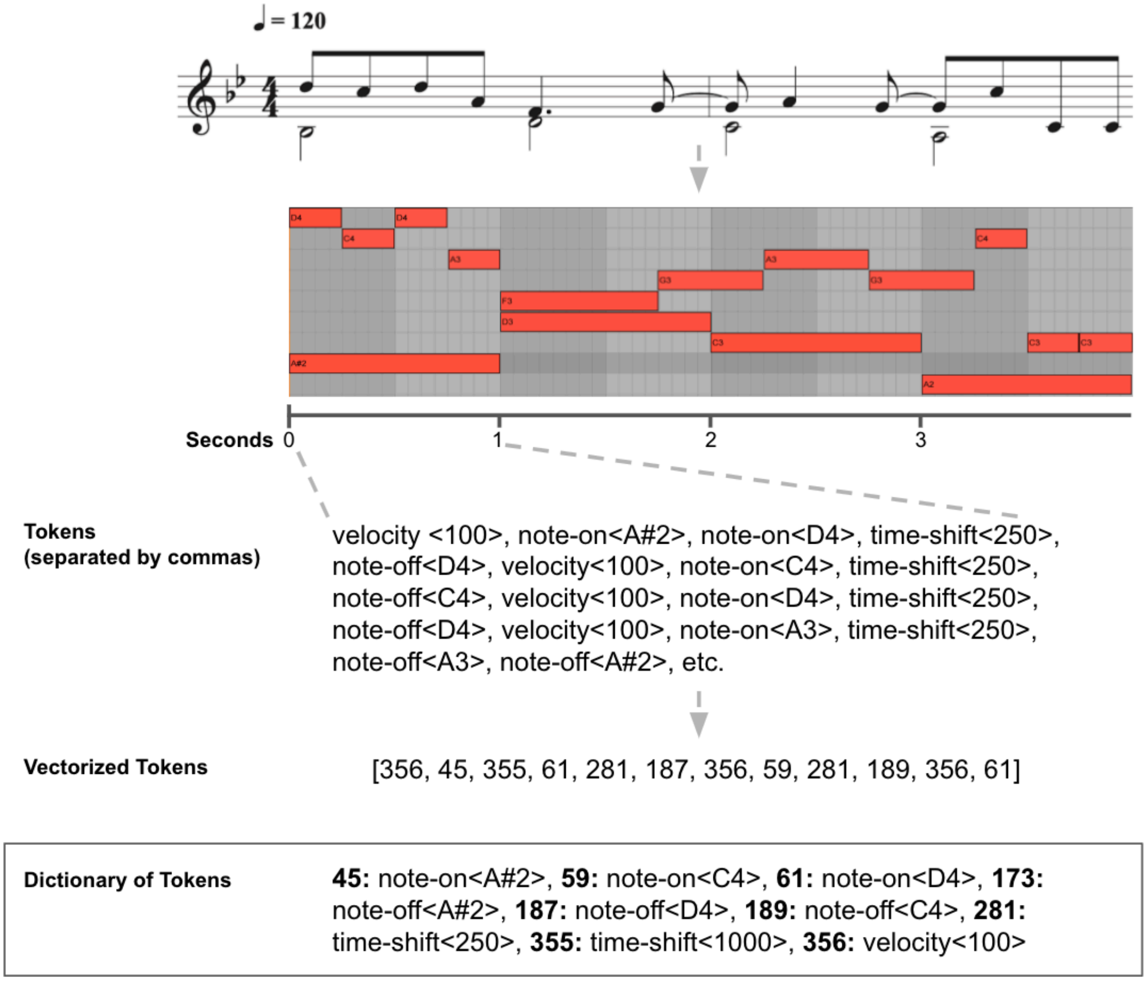
(I used a value of 100 so ST would playback any generated excerpts at about a mezzo forte). This elegant solution consists of note events (note-on and note-off) and the distance in time between these events referred to as 'time-shifts' by Oore et al. (Fig. 7). Time-shifts occur in increments of 10 milliseconds up to 1 second[45] and tell when the next note event will occur. In stacking notes into a chord, the encoding process simply adds no time-shifts between these note-on events. To create a melodic line, each note-on event would be followed by a time-shift (see Fig. 7). Before converting each event to its respective token (tokenizing), each track (MIDI instruments or staves) must be concatenated into a single vector of MIDI events by ordering each event found in each separate list by the exact time it occurs before quantizing these values into time shifts. Once this single vector which describes the music of any number of staves or tracks has been created, a vectorized form of a MIDI recording exists which can now be understood by the model.

This conversion process reduces a MIDI recording to two basic musical elements, pitch and duration (excluding velocity since it is kept the same). As mentioned, the goal is to notate the generated output excerpt from the model and therefore this simplistic rendering of music is all that is required. While more musical elements could theoretically be retained, ST is not meant to fully replace the human composer in all aspects of music creation but rather to aid them, not unlike precedents set forth by Arnold Schoenberg's twelve-tone matrix[46] or Iannis Xenakis' stochastic music[47]. Furthermore, each musical element that is added will increase computational expense which can affect the practicality of ST.

[45] Ibid., p.14.

[46] Arnold Schoenberg. *Style and Idea: Selected Writings of Arnold Schoenberg: 60th Anniversary Ed.* (Berkeley: University of California Press, 2010), p.207-250.

[47] Xenakis, 1992, p.1-42.



Tokens (separated by commas)

velocity <100>, note-on<A#2>, note-on<D4>, time-shift<250>, note-off<D4>, velocity<100>, note-on<C4>, time-shift<250>, note-off<C4>, velocity<100>, note-on<D4>, time-shift<250>, note-off<D4>, velocity<100>, note-on<A3>, time-shift<250>, note-off<A3>, note-off<A#2>, etc.

Vectorized Tokens

[356, 45, 355, 61, 281, 187, 356, 59, 281, 189, 356, 61]

Dictionary of Tokens

45: note-on<A#2>, **59:** note-on<C4>, **61:** note-on<D4>, **173:** note-off<A#2>, **187:** note-off<D4>, **189:** note-off<C4>, **281:** time-shift<250>, **355:** time-shift<1000>, **356:** velocity<100>

Figure 7: Encoding process. Vectorized tokens relate to musical events occurring in the MIDI recording in order. Time-shifts are in milliseconds.

After this encoding system was developed, ~180,000 MIDI recordings of music[48] were converted in order to become the input for the model. Using these MIDI databases which contain a wide variety of music is beneficial for the output of the model as too much of one composer, style or genre will likely bias the output. However, as will be noted in the next section, one can fine-tune the model on unseen MIDI recordings if the user notices an unwanted bias or simply wishes to train the model further for a

[48] MIDI recordings were taken from the MAESTRO dataset (Hawthorne et al., 2019), the Lakh dataset (Raffel, 2016) and www.imslp.com.

specialized task. In creating a genre-agnostic model, a perfect dataset would contain an evenly split cache of pieces pertaining to all the various types of music. However, due to the subjectivity of genre and style, it is more likely that the model will contain some bias to overrepresented groups and that is why fine-tuning is a necessary and desirable aspect of the model's design. In an effort to further reduce bias pertaining to key or pitch collections, data augmentation, a method of artificially increasing the dataset[49], was applied. The dataset was augmented by transposing each MIDI recording up 6 and down 5 semitones in an effort to transpose each piece to encompass all possible pitch collections. By transposing each piece 11 times, the dataset grew in size to ~1.43 million pieces of music. As a high correlation exists between the accuracy of deep learning systems and their dataset size, this addition was included to increase the model's output ability. Thus, this increase in the dataset will boost the model's ability to supply a composer with appropriate excerpts regardless of pitch collection.

2.2.2 Training & Fine-tuning

Training a deep learning model first involves selecting the best possible model architecture for the task at hand. Model architecture refers to the number and types of layers used in order to best interpret meaningful information in a given dataset[50]. A successful model architecture is one where the actual structure of the data matches well with the assumptions made by the model[51]. For example, when a deep learning model (such as the transformer-decoder model) is trained on a dataset of English language samples, it will determine parts of speech by itself in order to write grammatically correct sentences during output generation. By contrast a shallow model (with few layers) may need human guidance as to where certain parts of speech are in a sentence in order to return similar levels of accuracy during generation. Therefore, I selected a deep learning model over other potential machine learning models to best absorb necessary musical

[49] François Chollet. *Deep Learning with Python.* (Shelter Island: Manning Publications, 2017), p.138-142.

[50] Lupker, 2021.

[51] Chollet, 2018, 319.

information or rules without supervision, such as cadences, functional harmony or counterpoint, from which the model could later autonomously create convincing musical excerpts. The specific deep learning model I chose was a multi-layer transformer-decoder model[52] which I modified to include relative positional representations[53] to increase the ability of the model during generation (Fig. 8). The transformer-decoder model is a variant of the original transformer model[54] which applies a multi-headed self-attention operation (see section 2.2.3) over the input in order to establish an output probability distribution which covers all potential 357 tokens[55]. Simplified, the multi-headed self-attention mechanism makes its predictions based upon observations of past frames in a sequence[56]. The base model architecture was open-source Python-language code created by artificial intelligence research company OpenAI for their transformer-decoder model "GPT-2"[57], which I later modified to use relative positional representations (RPRs) [58]. These RPRs embed the positions of nearby frames in a sequence to capture recurring stylistic musical gestures, maximizing the representation learning of preceding frames. The implementation of RPRs better interpret what musical gestures tend to follow each other given the stylistic approach of certain composers. This may include gestures which other composers may not use, such as Chopin's predilection for complex flourishes. The addition of RPRs was influenced by previous work by Huang et al.[59] who similarly added them to their own separate variant of the transformer model (a transformer-encoder

[52] Alec Radford et al., "Improving Language Understanding by Generative Pre-Training". *openai.com*, (2018).

[53] Peter Shaw et al., "Self-Attention with Relative Position Representations". *arXiv.org*, (2018).

[54] Ashish Vaswani et al., "Attention is All You Need". *arXiv.org*, (2017).

[55] Radford et al., 2018.

[56] Dzmitry Bahdanau et al., 2014.

[57] Radford et al., 2018.

[58] Shaw et al., 2018.

[59] Huang et al., 2018.

model unrelated to my model or GPT-2) and demonstrated increased ability of their model to generate lower-level stylistic musical gestures. Section 2.2 will further discuss specific individual sections of my transformer-decoder model in depth.

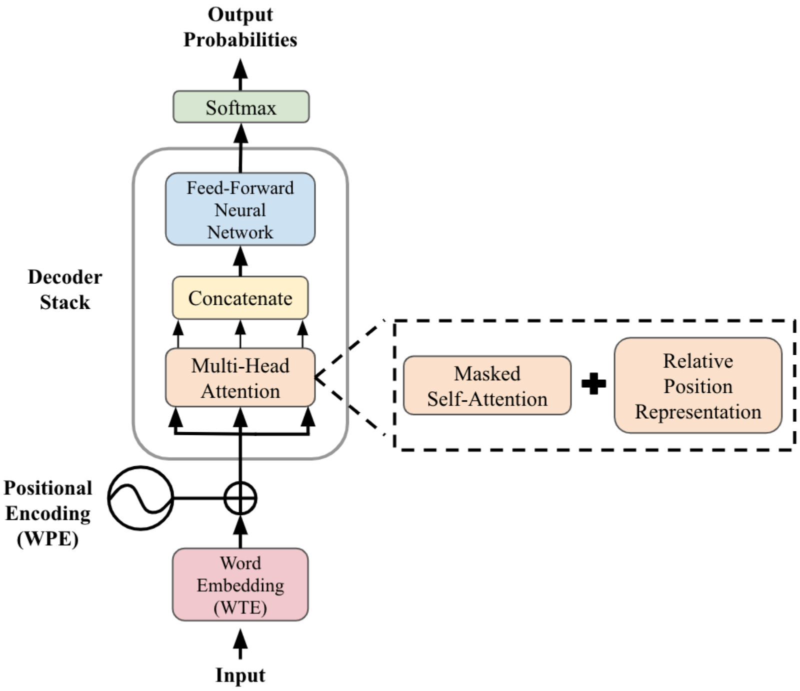


Figure 8: Transformer-Decoder model architecture modified to include relative position representations.

The artificial intelligence behind ST was built to learn in two stages: first, the initial high-capacity training where the model ‘learns’ to write music by absorbing all available information from using a large corpus (dataset) and second, fine-tuning, training using a much smaller corpus previously unseen by the trained model in order to support new goals. Training involves a series of steps whereby each involves a selection of the dataset being passed through the model for processing and testing. How well the model performs on each test is then fed back to the model and used to adjust parameters in order to (hopefully) do better in the next step. The goal is to minimize errors or loss (see section 2.2.5) as much as possible (perfect equaling a loss of 0 where 0 errors were made), thus establishing when the model can be said to be trained. ST was trained over a period of ~50,000 steps while reaching a loss of approximately 1.7[60] (see section 2.2.5). The size of the model (Fig. 8), which is measured by the number of decoder stacks, number of 'attention heads' in the multi-head attention layer and the size of input (how many tokens to send the model) can be increased to boost the efficacy of the model's ability to extract useful information but comes at a computational cost. The bigger the model, the slower the performance during generation. If generation is too slow, it could cause composers to lose interest in the software. A model size of 24 decoder stack layers, 16 attention heads and a maximum input size of 1024 tokens was found to produce an accurate model that could generate excerpts[61] of 200 tokens in under 30 seconds on a standard Apple MacBook laptop. In comparison, a model size of 10 decoder stack layers, 10 attention heads could generate a 200-token excerpt in ~5 seconds. The quality of the latter excerpt, however, was noticeably weaker than that generated by the larger model.

Fine-tuning has been previously demonstrated in natural language tasks where the pre-trained model struggled with specialized tasks but showed improvements when fine-tuned on new datasets[62]. While ST has been trained to accomplish many musical tasks, there will be times when a composer may require the software to accomplish something specific, therefore making it an imperative feature to incorporate within the software. The composer can also fine-tune ST on their own work, giving the model a better idea of compositional conventions or preferences that the composer may have. Thus, ST becomes modular, allowing the user to shape it in any direction they wish, keeping it constantly updated.

[60] This seems to be in keeping with *Music Transformer* (Huang et al., 2018) and other models addressed within their paper with losses varying from 1.8-2.0 on the MAESTRO dataset. As my model was trained upon a larger and more diverse dataset, it's hard to address exact comparisons between the ability addressed in the paper and my own.

[61] Excerpts generated during output cannot be longer than the input length, in this case 1024 tokens. For context, a single note is generally made up of about 4 tokens (velocity, note-on, note-off, time-shift).

[62] Iz Beltagy et al., "SciBERT: Pretrained Contextualized Embeddings for Scientific Text." *arXiv.org.* (2019).

2.2.3 Sampling

Once the model has been trained, a weighted probability distribution is established across all possible tokens to be used during sampling. Sampling involves the random selection of the next token in a sequence based on this weighted probability distribution[63] repeating until the length parameter has been satisfied (Fig. 9). This probability distribution can be further manipulated with three parameters: *temperature*, *top k* and *top p*[64]. In this context, the term *temperature* is borrowed from statistical thermodynamics[65] where a high *temperature* equals a more random output for token selection, and a low *temperature* equals a more deterministic output[66]. Low *temperatures* will increase a model's confidence in token selection, selecting the more probable options while high *temperatures* will decrease confidence (Fig. 9). A *temperature* of 0 would tell the model to make completely deterministic decisions (Fig. 10). The *temperature* value can be any positive real number.

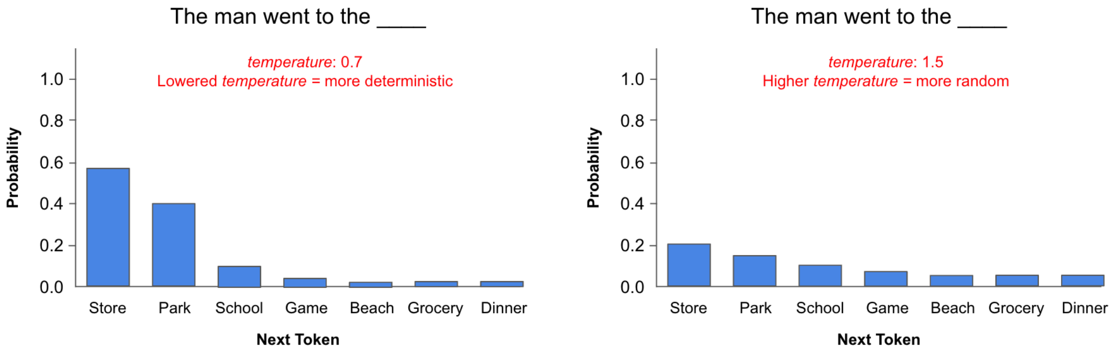


Figure 9: Probability distribution affected by altering hypothetical *temperature* values.

[63] Chollet, 2017, p.276.

[64] Ibid.

[65] Ben Mann. "How to Sample from Language Models." *towardsdatascience.com* (2019), p.2

[66] Chollet, p.276.

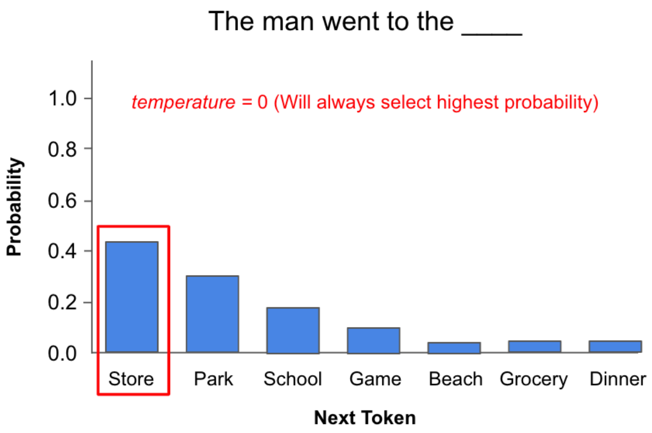


Figure 10: *Temperature* of 0 makes completely deterministic decisions.

Top k is a parameter which tries to "sample from the tail," meaning that when all tokens are ranked by their respective probabilities from lowest to highest, a cut-off can be determined in order to only select from *k* amount[67] of high probability tokens (Fig. 11). For example, if we have 100 tokens where the top 4 have decidedly higher probabilities than the last 96, we can remove those unwanted 96 tokens from the selection pool (sample space) entirely by implementing a *top k* of 4. Furthermore, instead of determining a hard cutoff like *top k*, we can also sample from the tail by means of calculating and then enforcing a cutoff based on cumulative probability distribution using the *top p* (also called nucleus sampling).

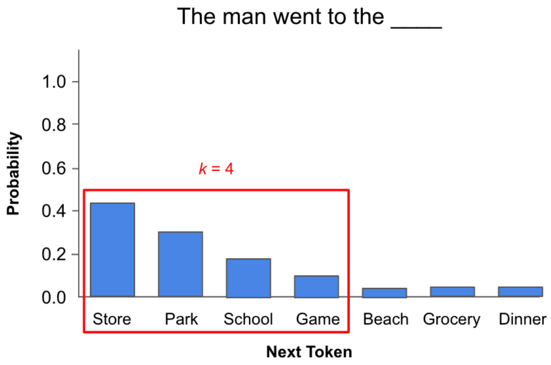


Figure 11: Top *k* used to select the top 4 highest probability tokens.

[67] The value of *top k* can't exceed the number of entries in the dictionary (357).

Top p is theorized to improve the quality of token selection by modulating the k-token cutoff amount on a case-by-case basis[68] (Fig. 12). As the probability distribution changes with every generated token as the model makes predictions based on everything that precedes it, having a hard cutoff of *k* tokens can have unwanted effects. For example, a broad distribution would feature a situation where the top 3 tokens might have a cumulative probability of 0.8[69] while the next 97 tokens will have a cumulative probability of 0.2, meaning that each found in this section has an extremely low probability in comparison to the top 3. Therefore, we might want to cut-off the bottom 97 tokens as they might not fit the context of our generated music. Conversely, with a narrow distribution, it might take 40 tokens to reach a cumulative probability of 0.8, a situation where we might actually want to sample from all of these given their similar probabilities. Thus, *top p* can potentially increase the accuracy of generated music by constantly increasing and decreasing the sample space for each next possible token.

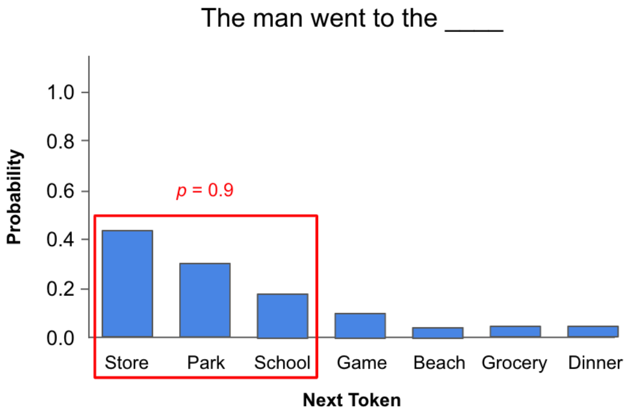


Figure 12: *Top p* used to select the top 3 highest probability tokens.

[68] Ari Holtzman et al., “The Curious Case of Neural Text Degeneration,” *arXiv.org* (2020), p.2.

[69] The sum of the probability distribution always equals 1 (see footnote 87). Thus, the *top p* value cannot exceed 1.

2.3 Operating Score-Transformer

ST is designed to work with any common music notation software in order to efficiently aid a composer's workflow. However, as I use MakeMusic's *Finale* 26 as my notation software, henceforth I will only refer to *Finale* throughout the rest of this document. ST is a standalone Mac application (Fig. 13) built using Python 3.6 language with a Tensorflow 1.4 backend, that can be opened to work in tandem with Finale. The following steps outline the operating process of using ST to either fine-tune the model on a new dataset or to generate an excerpt of music.

Fine-tune the model

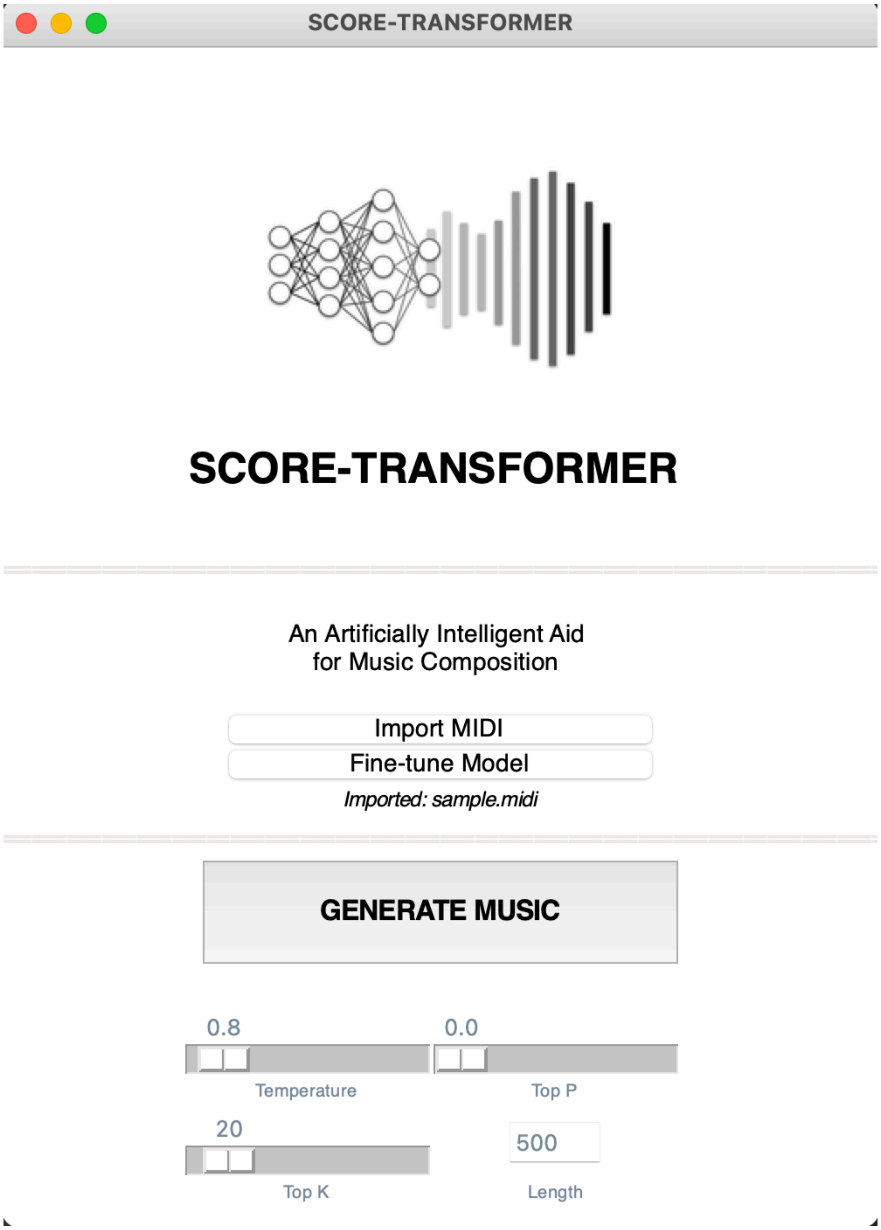
1. In Score-Transformer, select **Fine-tune model**.
2. This brings up a window for the user to select a folder with MIDI files.
3. Score-Transformer will begin fine-tuning on the new dataset and will show complete when finished.

Generate a musical excerpt

1. In Finale, export any musical selection (typically 1-16 measures of music) to a MIDI file using the pathway **Export → MIDI File**… → **Save**.
2. In Score-Transformer, import the MIDI file using the **Import MIDI** button. This will bring up the sampling parameters which can be manipulated.
3. Select sampling parameters and hit **Generate Music**.
4. Once music has been generated, the app will playback the initial imported MIDI file along with all newly generated material. A prompt will ask whether to **Open Score** in Finale or to go back to the sampling parameter window (Step 3) in order to generate a **New Sample**.
5. If **Open Score** was selected, the newly generated music will be opened up as a separate Finale file. Any required edits can now be made directly to the score or it can simply be copy and pasted into the score the composer is working on.

Notes:

- Finale’s quantization parameters can be changed to better suit the incoming file.
- ST generates music onto two staves regardless of how many there were in the prompt.



An Artificially Intelligent Aid
for Music Composition

Imported: sample.midi

Temperature 0.8

Top P 0.0

Top K 20

Length 500

Figure 13: Main window for Score-Transformer.

Part 3

3 *Music for Self-Attention*

Music for Self-Attention is a piece for string quartet in six movements which utilizes Score-Transformer (ST), software built using current state-of-the-art deep learning technologies more commonly associated with natural language. The use of deep learning technology within ST software as a compositional aid provided an innovative algorithmic approach to traditional compositional methods while demonstrating that this non-musical system has potential benefits for music composition that go beyond previous demonstrations. *Music for Self-Attention* is meant to exhibit one kind of symbiotic relationship that can exist between artificial and human creativity in music composition.

The string quartet was chosen for its diverse range of possible pitch registers, textures and playing techniques. All of these seemed beneficial in an effort to highlight various ways in which ST can be applied to composition using acoustic instruments. Since ST was trained on the pitch and durational values in MIDI recordings of music and has no ability to read any performance or technique notes within scores, no extended techniques are utilized within the piece. While these may be additional features incorporated into future work using deep learning software aids, this piece focusses on the early process of adopting the methods of this technology into composition. As I had no initial knowledge as to what the rhythmic or pitch content would be for the first five movements of the piece, I did not want to gravitate to any particular composer's style in composing for string quartet. The following composer's works for string quartet were casually consulted in preparation for this piece: Arnold Schoenberg (see section 3.2.7), Bela Bartok, late Ludwig van Beethoven string quartets, Iannis Xenakis, Anton Webern and Alban Berg.

As noted in part 1, inspiration for the use of non-musical systems applied within a musical domain came from Schoenberg and Xenakis' application of formalized mathematics principles into their own compositional methods. Combinatorics and the laws of the calculus of probabilities provided Schoenberg and Xenakis respectively the means to develop the twelve-tone matrix and stochastic music[70]. Two movements of the piece feature small homages to these composers which will be discussed in later sections. The following sections will provide the score for the piece and an extensive analysis. Analytical subsections per movement will include the usage of ST and pre-compositional work incorporating transformer-decoder processes into the movement themselves.

3.1 Score for *Music for Self-Attention*

3.1.1 Instrumentation

- Violin I
- Violin II
- Viola
- Cello

3.1.2 Score

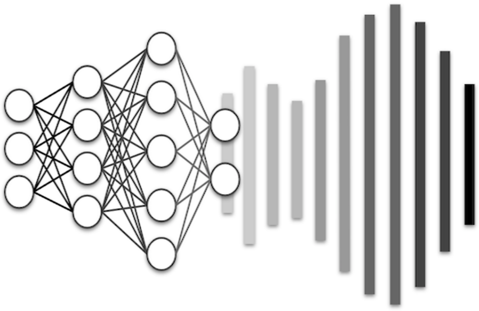
Begins on the following page.

[70] Loy, 2011, p.306, 332.

J. LUPKER

MUSIC FOR SELF-ATTENTION

for String Quartet



2021

I
WTE + WPE

II
Multi-Head Attention

III
Neural Networks

IV
Loss & Gradient Descent

V
Hyperparameter Search

VI
Generation

Violin I

Violin II

Viola

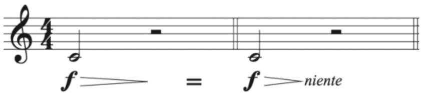
Cello

***Music for Self-Attention** features an innovative algorithmic approach to traditional compositional methods by demonstrating the benefits of deep learning systems (artificial intelligence) in music composition. This piece of music, for string quartet, utilized Score-Transformer throughout the entire compositional process to aid certain pitch and rhythmic decisions. These artificially generated decisions are not intended to fully remove the human element from composing but rather to work in tandem with the composer, in this case myself. The piece lasts approximately 23 minutes and loosely follows a form of theme and variations in reverse. This steady process gradually deconstructs a series of variations each initially generated by artificial means until the final movement, which contains music created with zero artificial intervention, is revealed. Each movement imitates different processes involved in the training and deployment of artificially intelligent software.*

***Music for Self-Attention** is meant to demonstrate a symbiotic relationship that can exist between artificial and human creativity in music composition.*

Notes

Any decrescendos that precede rests without specific dynamic demarcation are to be realized as a decrescendo to niente.



for Beth

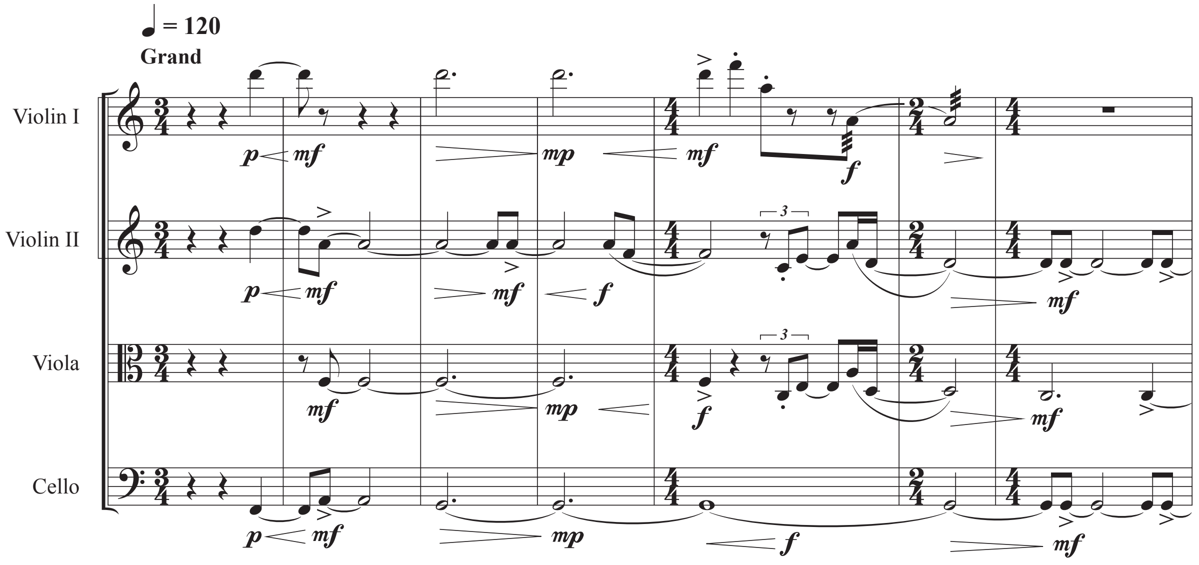
J. LUPKER

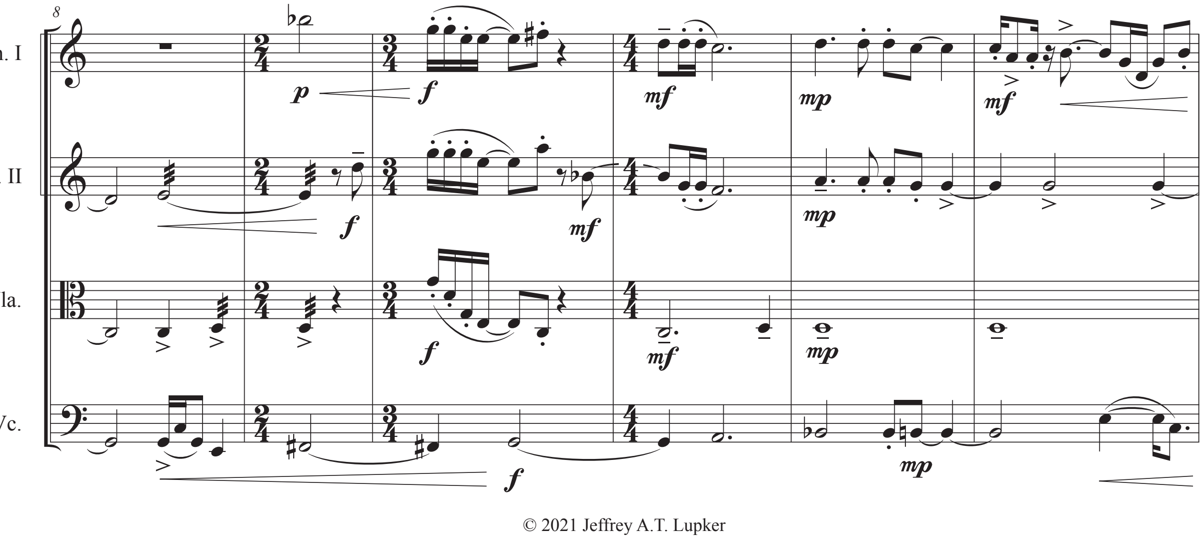
MUSIC FOR SELF-ATTENTION

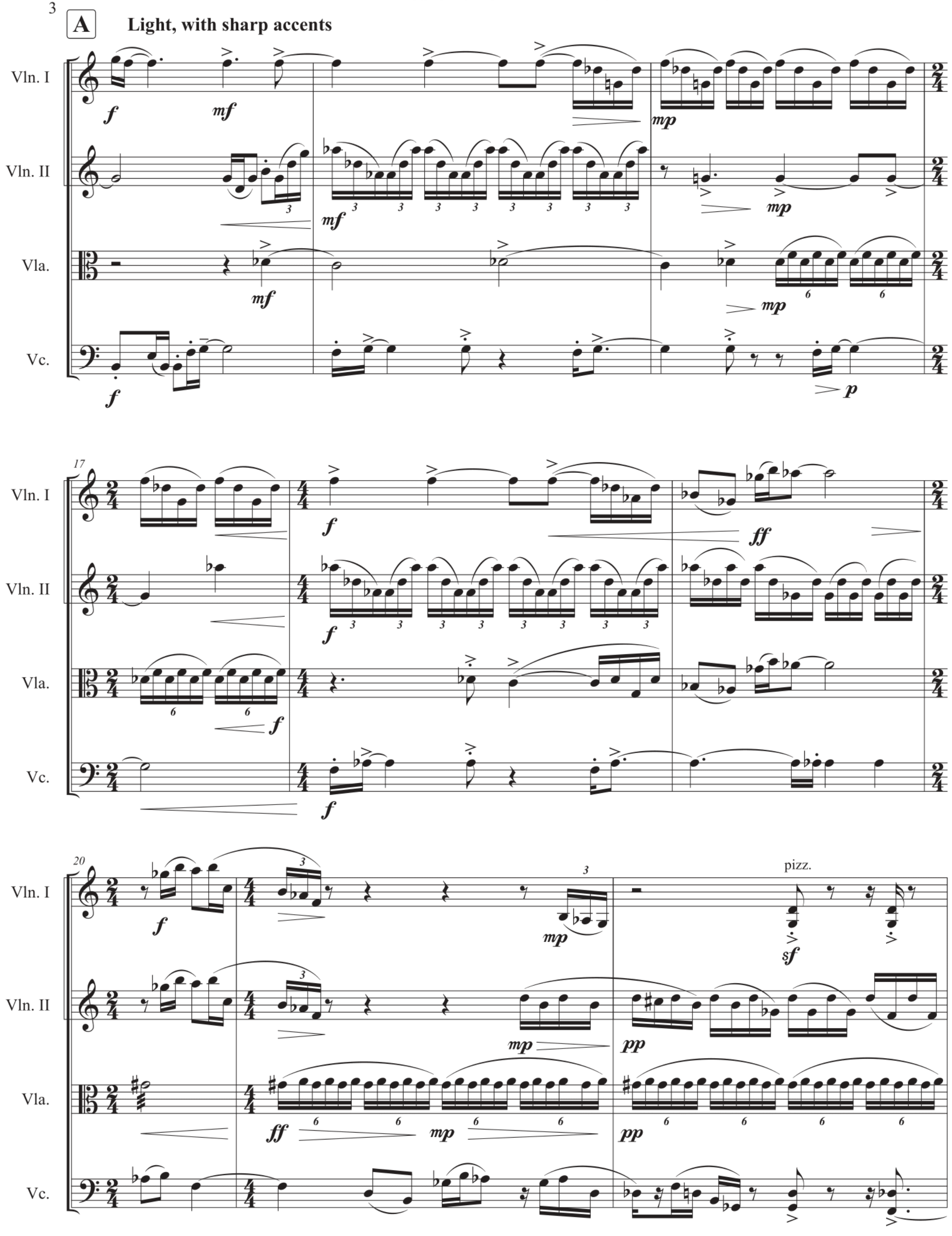
2021

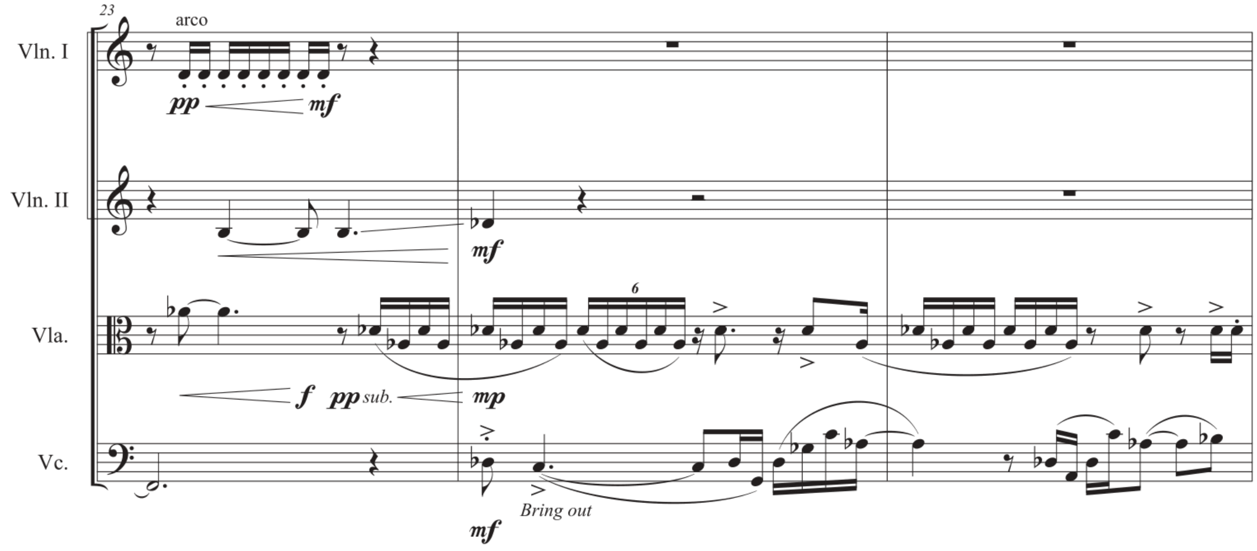
I

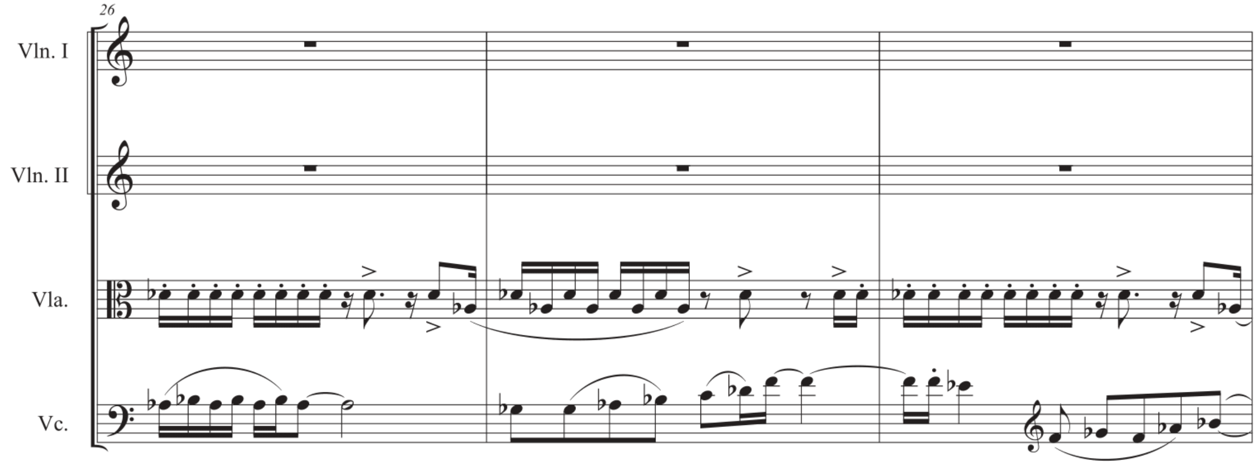
WTE + WPE

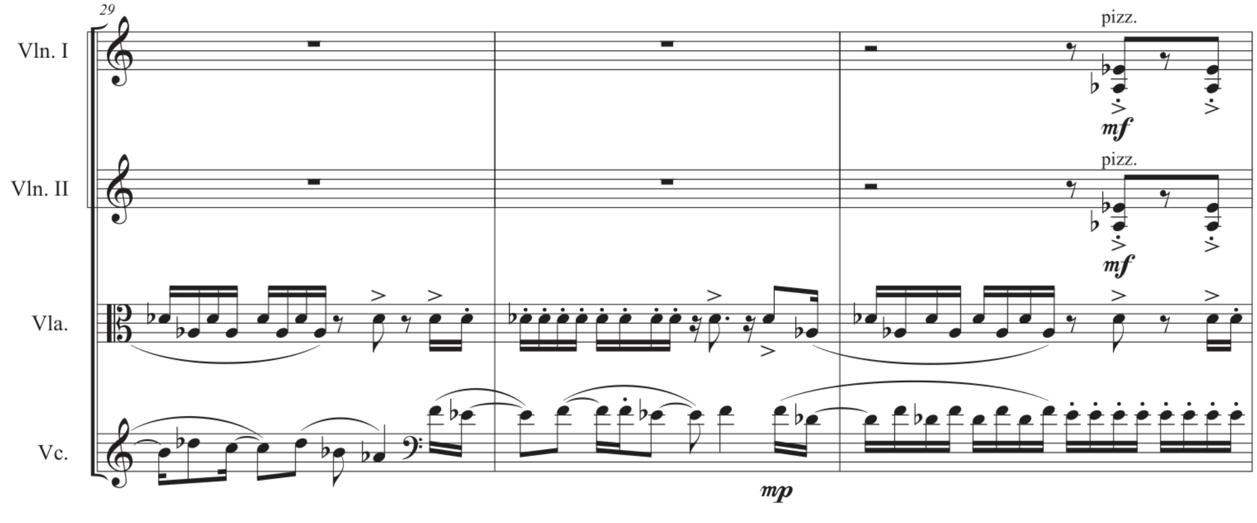


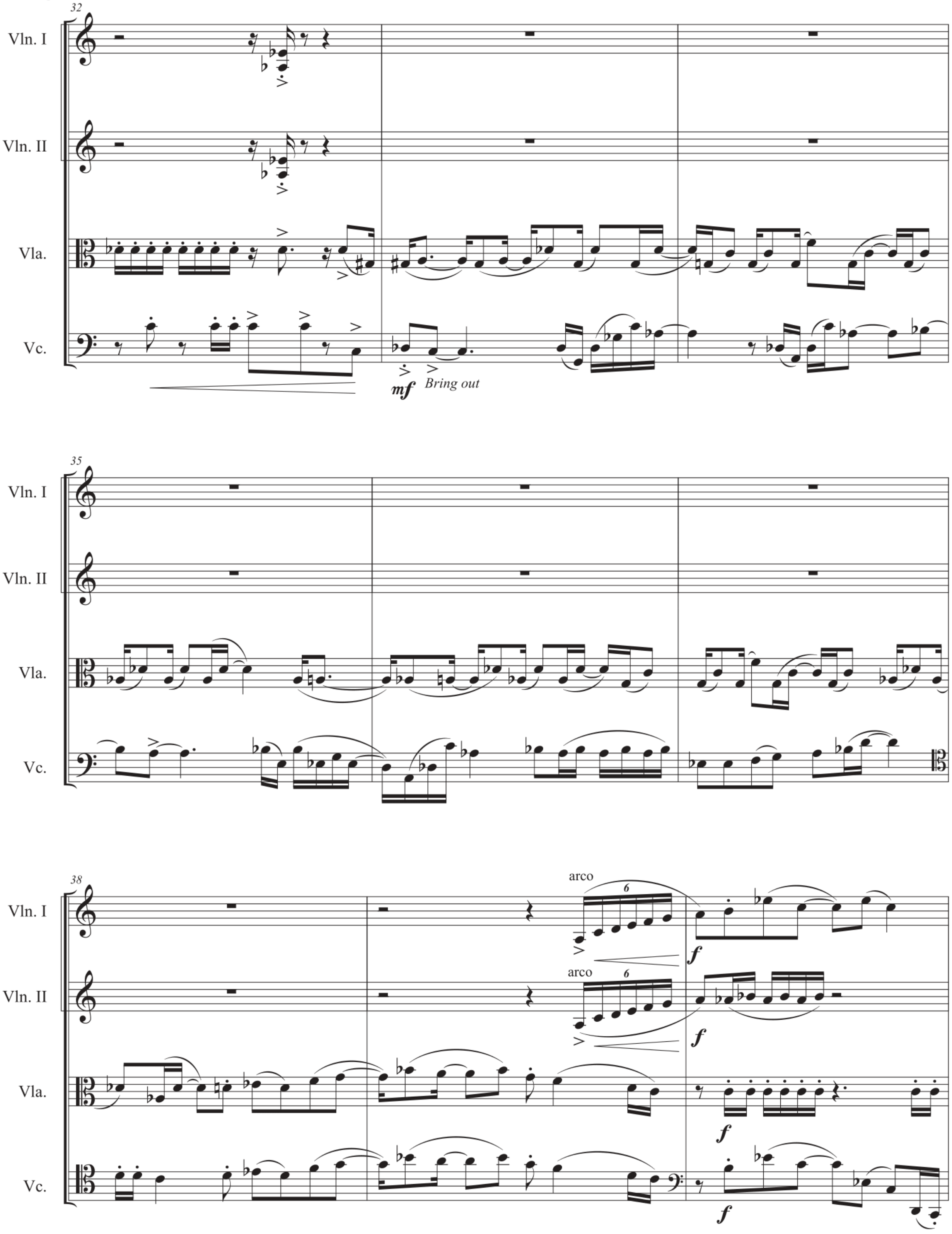


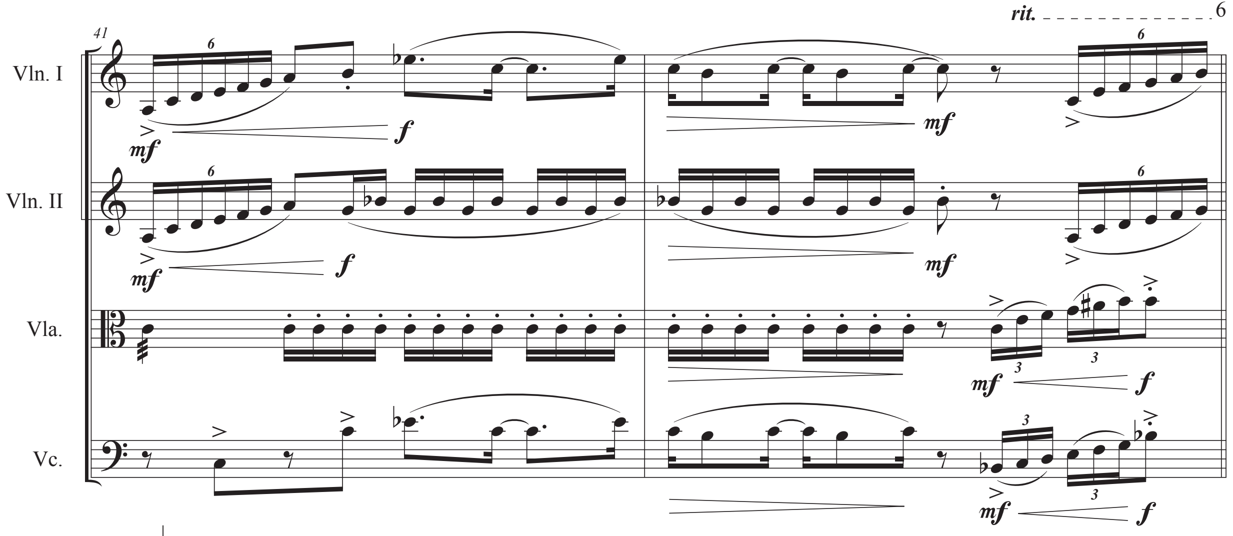

3
A
Light, with sharp accents
Vln. I
Vln. II
Vla.
Vc.
f
mf
mp
mf
mf
mp
f
p
17
f
ff
f
f
f
20
f
mp
pizz.
sf
mp
pp
ff
mp
pp

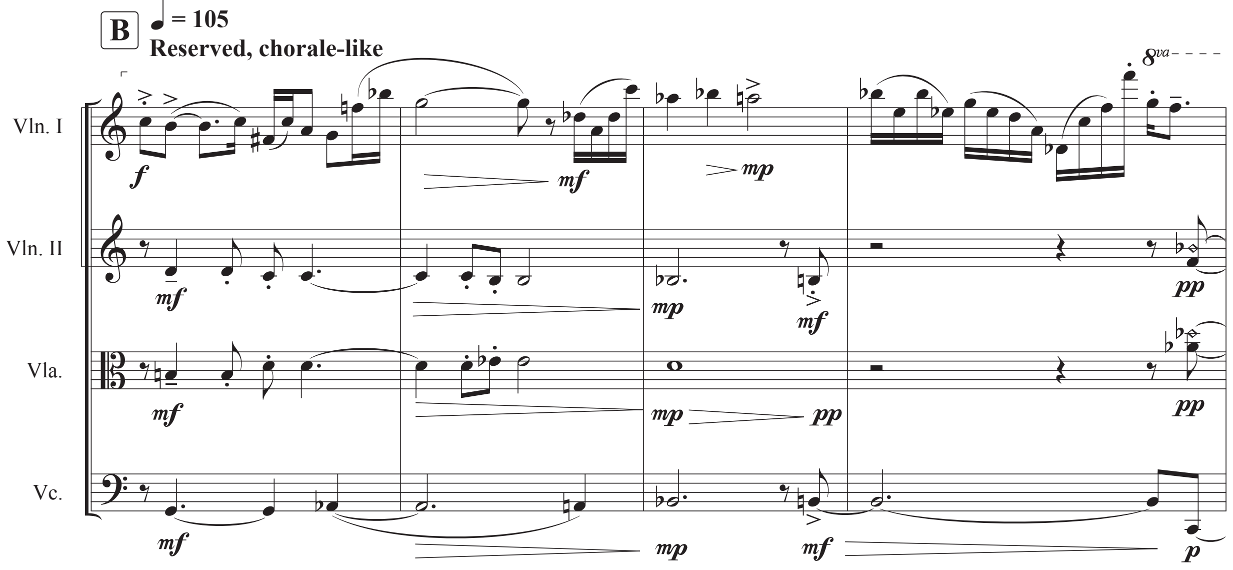

23
Vln. I
Vln. II
Vla.
Vc.
arco
pp
mf
mf
6
f
pp sub.
mp
mf
Bring out

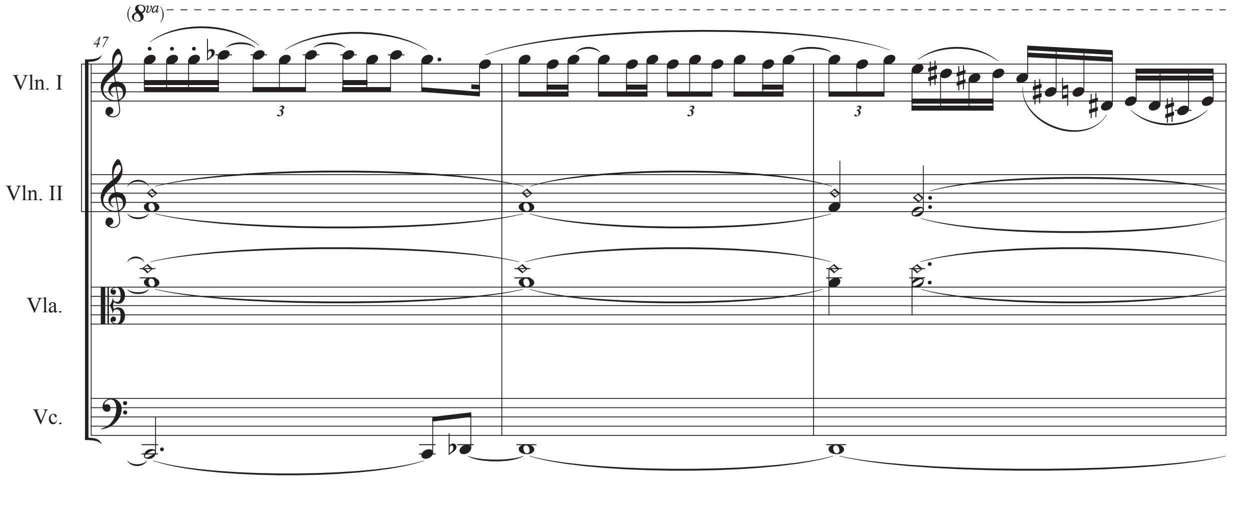

26
Vln. I
Vln. II
Vla.
Vc.

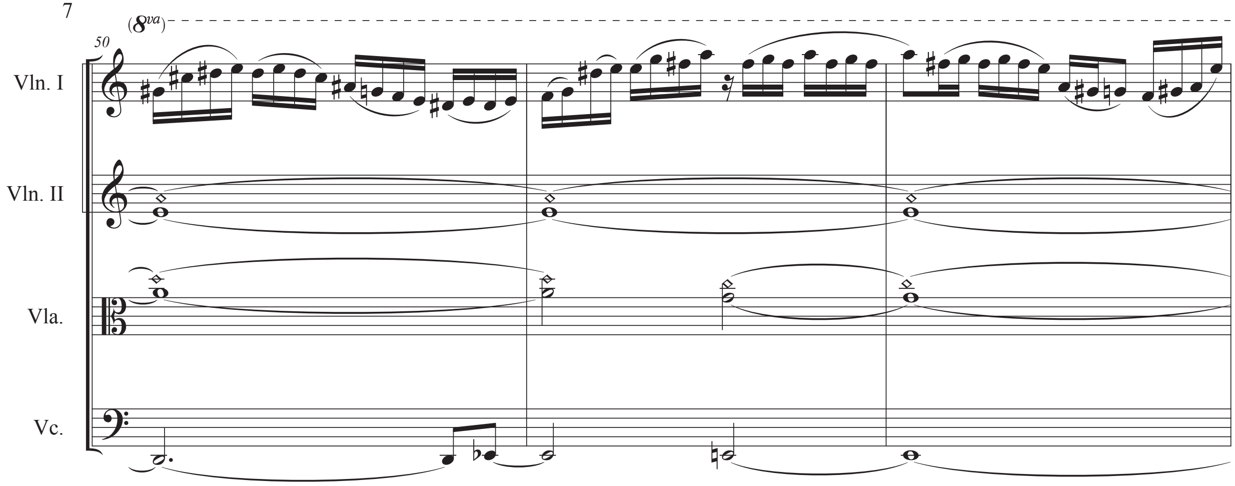

29
Vln. I
Vln. II
Vla.
Vc.
pizz.
mf
pizz.
mf
mp

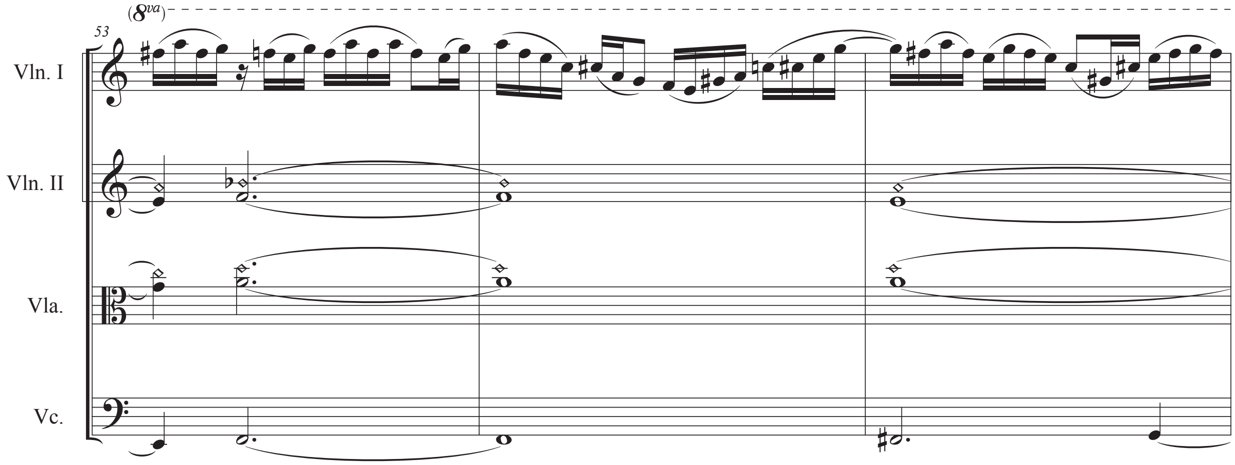

32
Vln. I
Vln. II
Vla.
Vc.
mf
Bring out
35
Vln. I
Vln. II
Vla.
Vc.
38
Vln. I
Vln. II
Vla.
Vc.
arco
6
f
arco
6
f
f
f

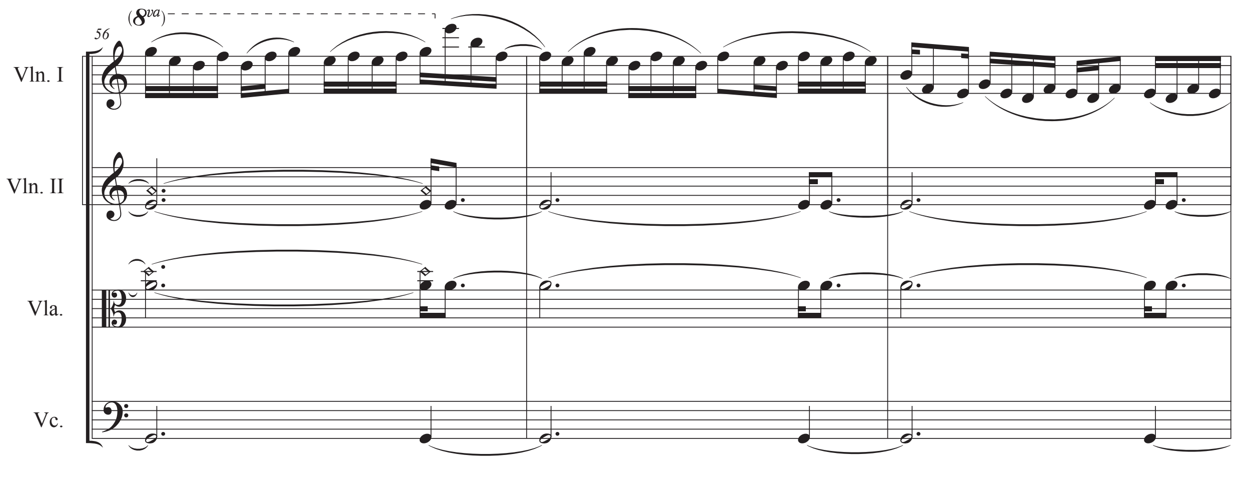

rit.
6
41
Vln. I
Vln. II
Vla.
Vc.
6
mf
f
mf
6
3
mf
f

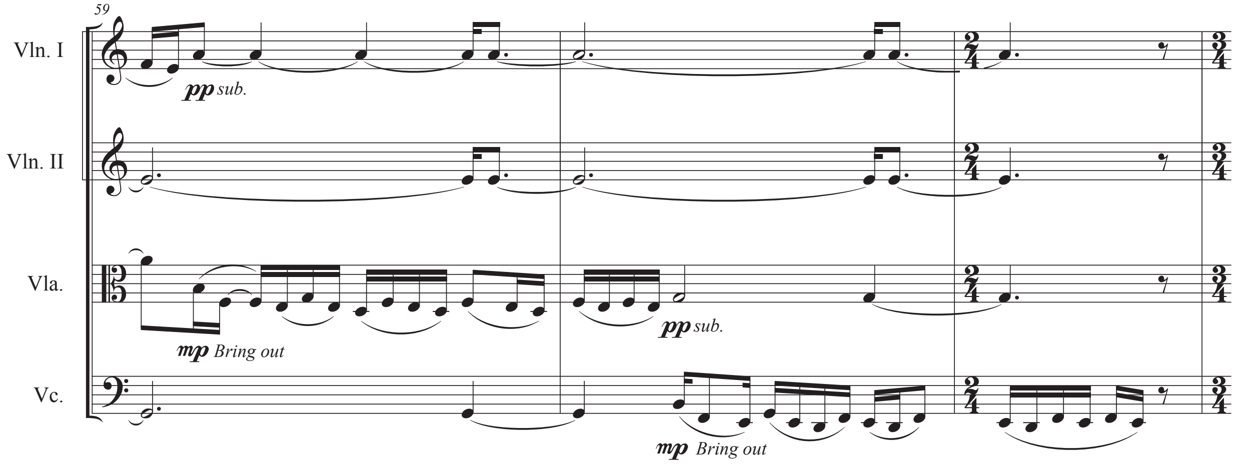

B
♩ = 105
Reserved, chorale-like
8va
Vln. I
Vln. II
Vla.
Vc.
f
mf
mp
mf
mp
mf
pp
mf
mp
pp
pp
mf
mp
mf
p

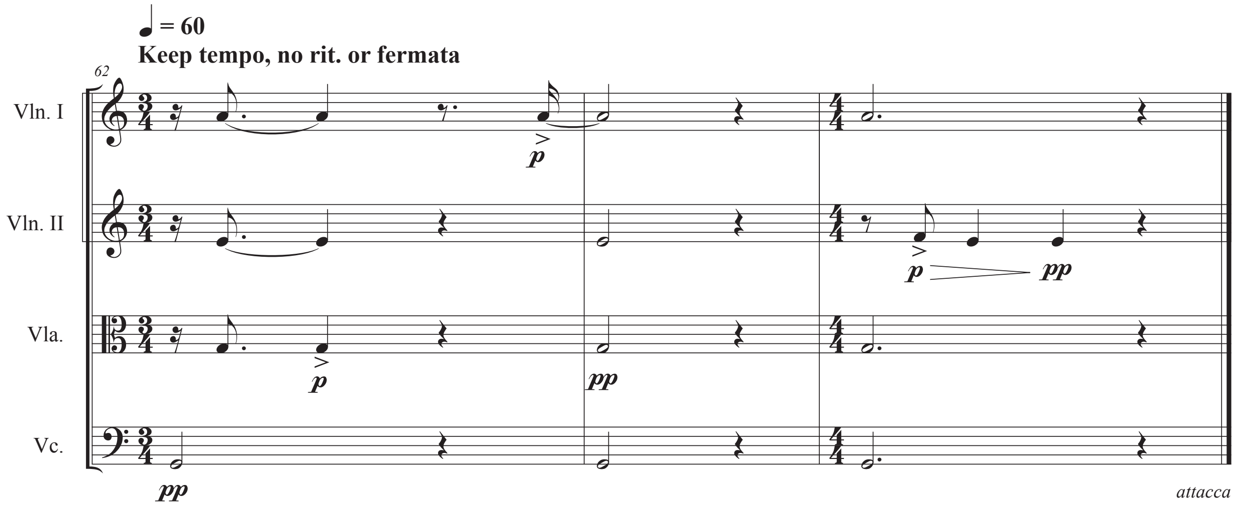

(8va)
47
Vln. I
Vln. II
Vla.
Vc.
3
3
3


(8va)
50
Vln. I
Vln. II
Vla.
Vc.

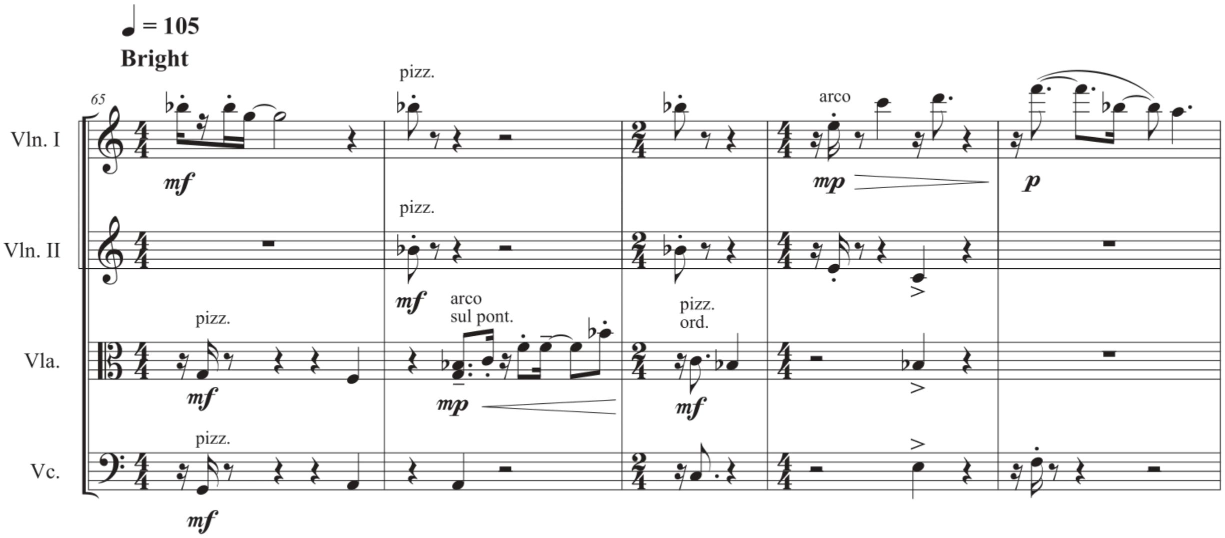

(8va)
53
Vln. I
Vln. II
Vla.
Vc.

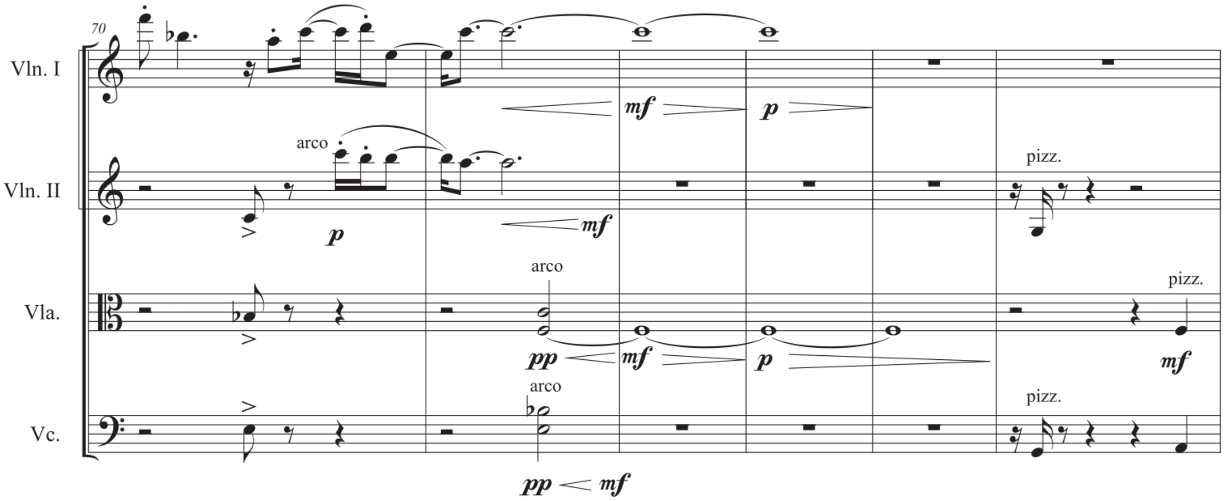

(8va)
56
Vln. I
Vln. II
Vla.
Vc.

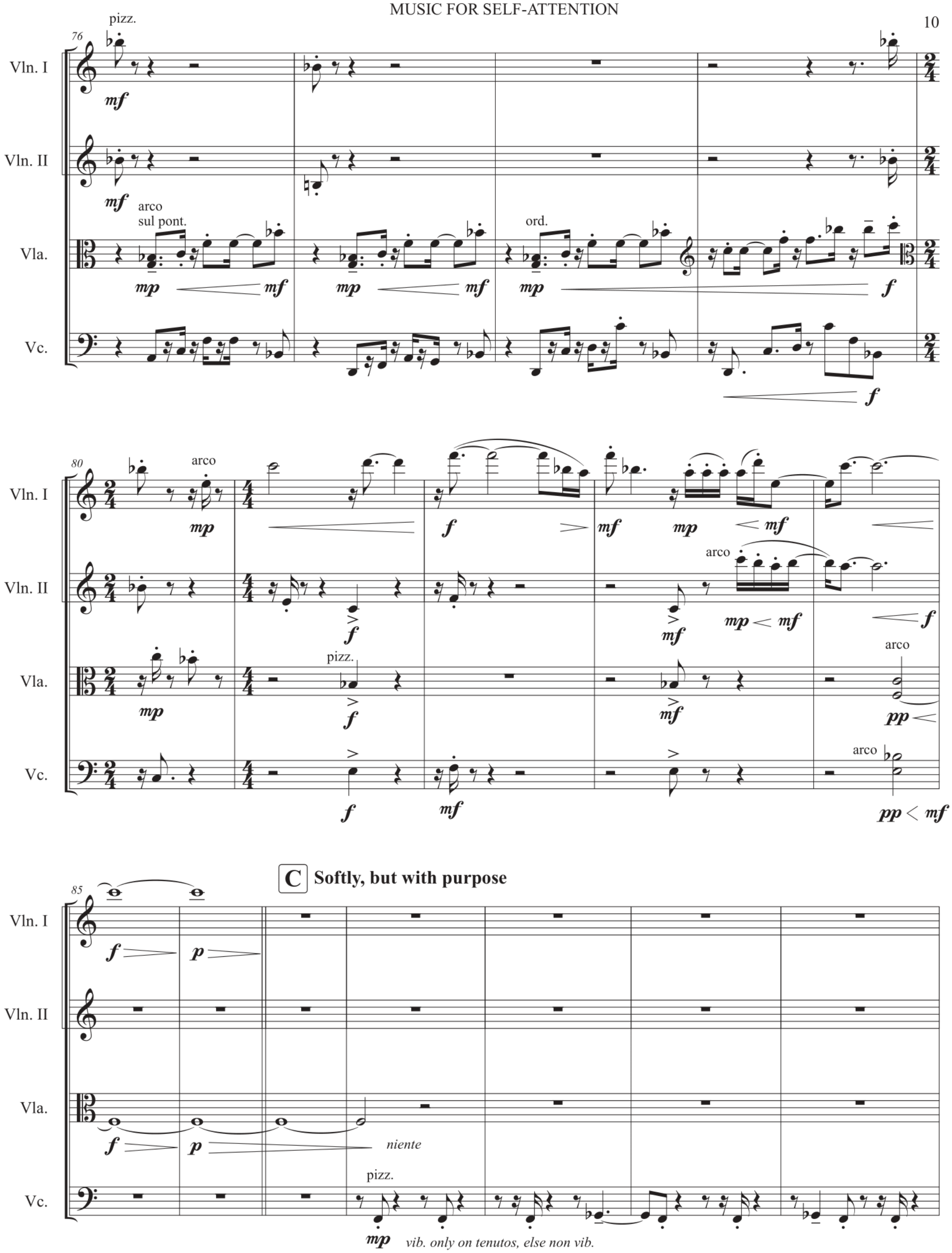

59
Vln. I
Vln. II
Vla.
Vc.
pp sub.
mp Bring out
pp sub.
mp Bring out


♩ = 60
Keep tempo, no rit. or fermata
62
Vln. I
Vln. II
Vla.
Vc.
p
p
pp
p
pp
pp
attacca

II
Multi-Head Attention

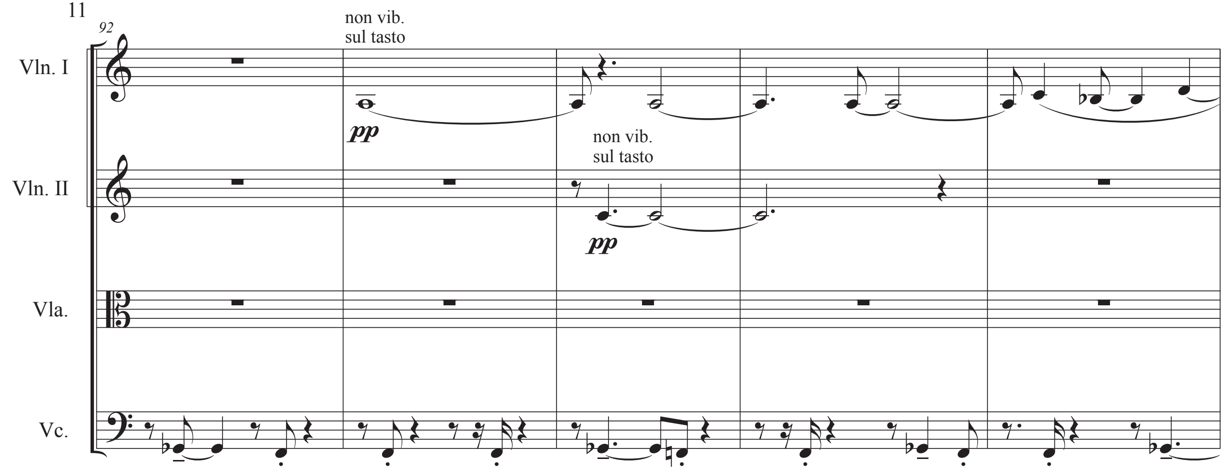


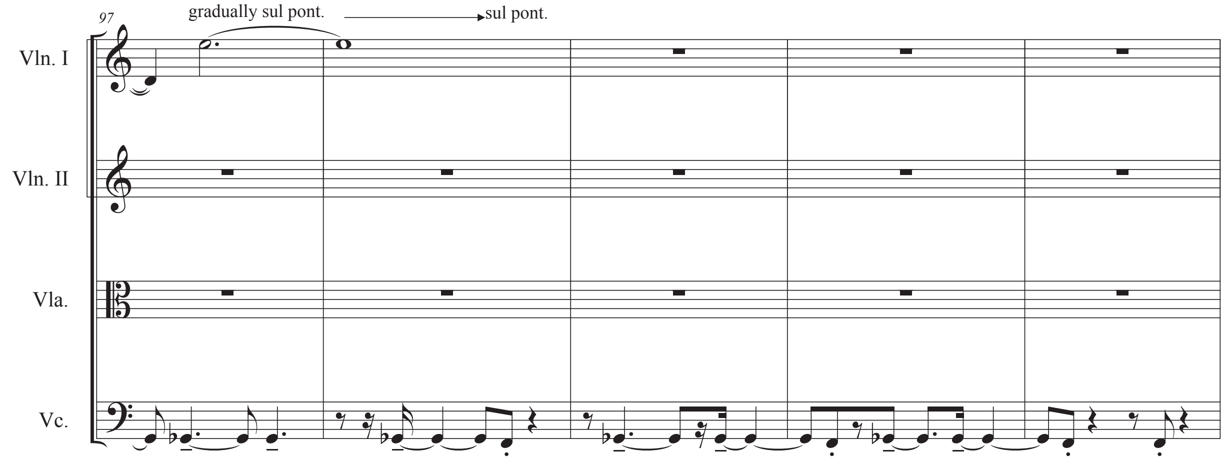


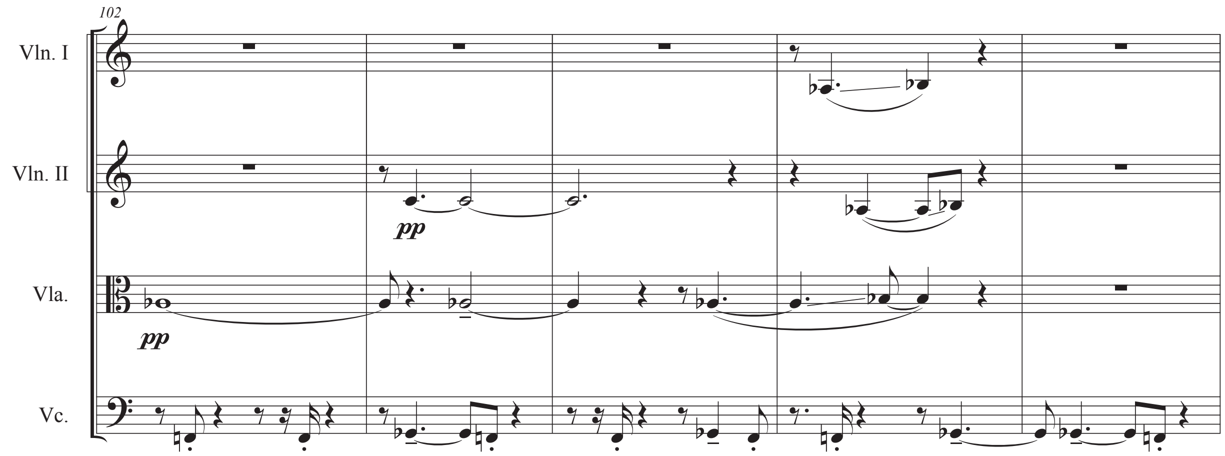

76
pizz.
Vln. I
mf
Vln. II
mf
arco
sul pont.
Vla.
mp
mf
mp
mf
ord.
mp
f
Vc.
f
80
arco
Vln. I
mp
f
mf
mp
mf
Vln. II
arco
f
mf
mp
mf
f
Vla.
mp
pizz.
f
mf
arco
pp
Vc.
f
mf
arco
pp
mf
85
C
Softly, but with purpose
Vln. I
f
p
Vln. II
Vla.
f
p
niente
Vc.
pizz.
mp
vib. only on tenutos, else non vib.

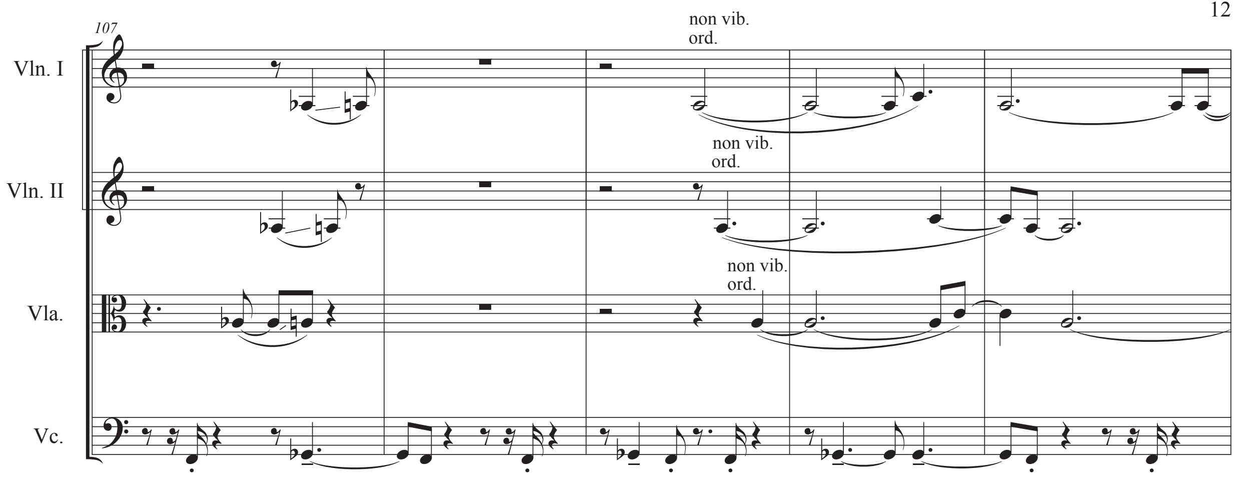
MUSIC FOR SELF-ATTENTION

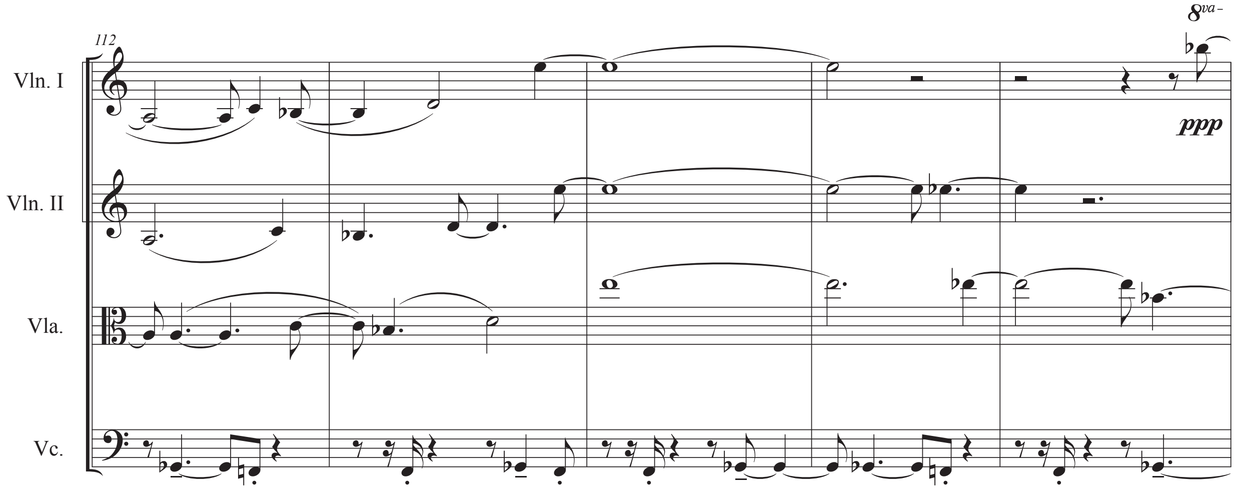
11

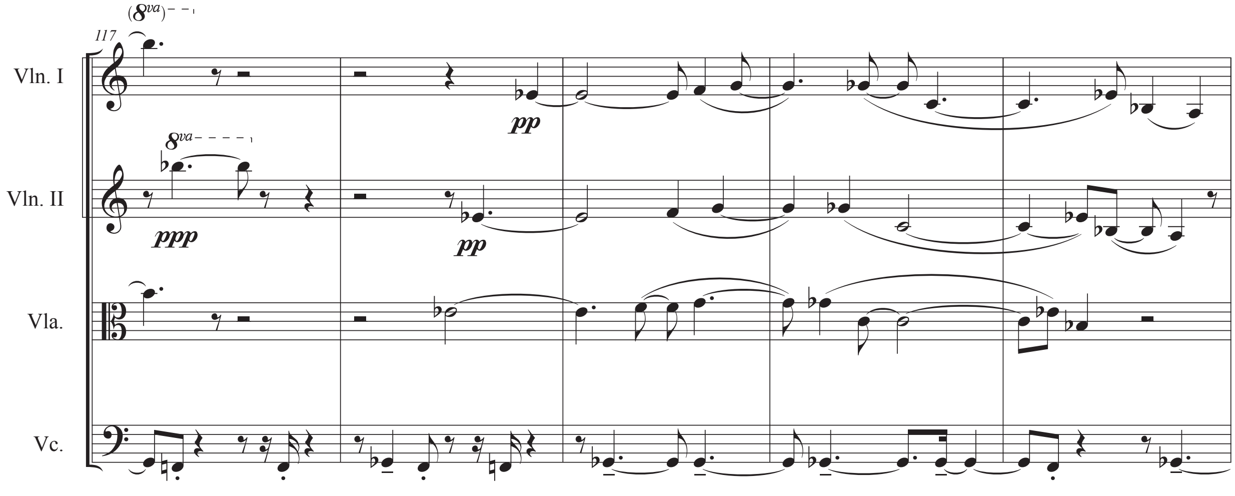


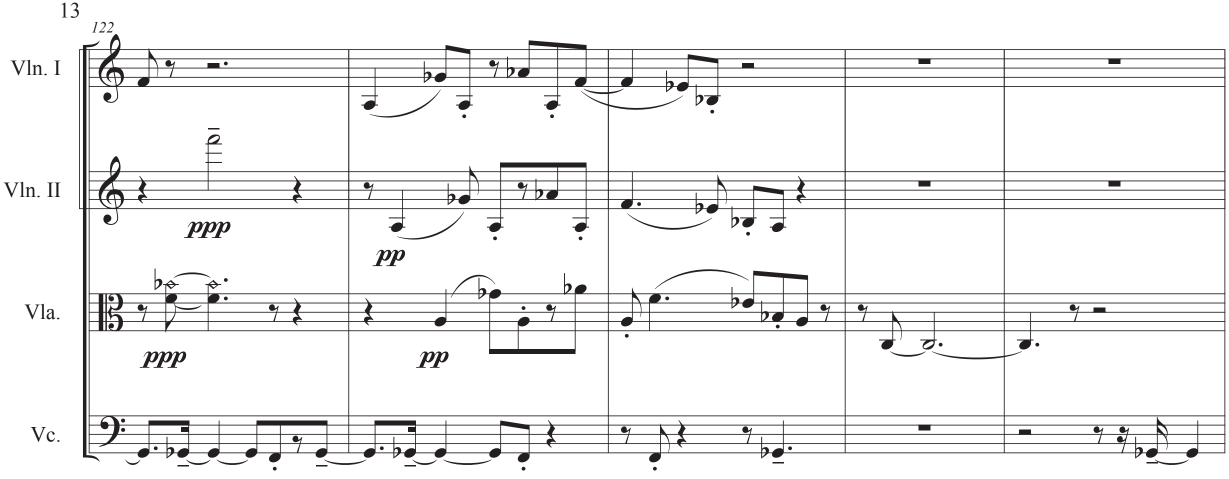


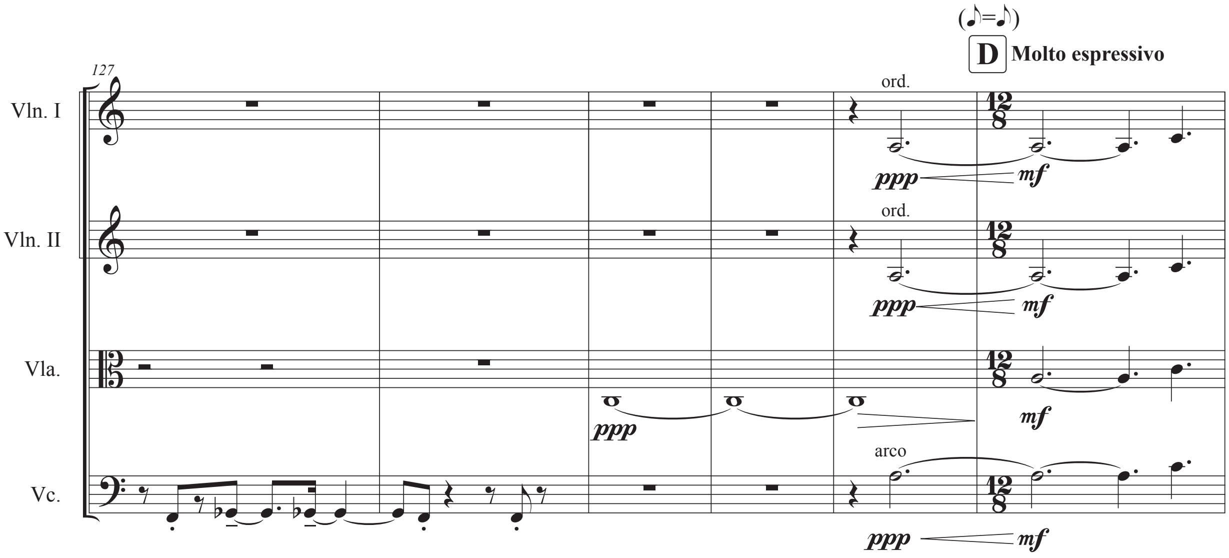


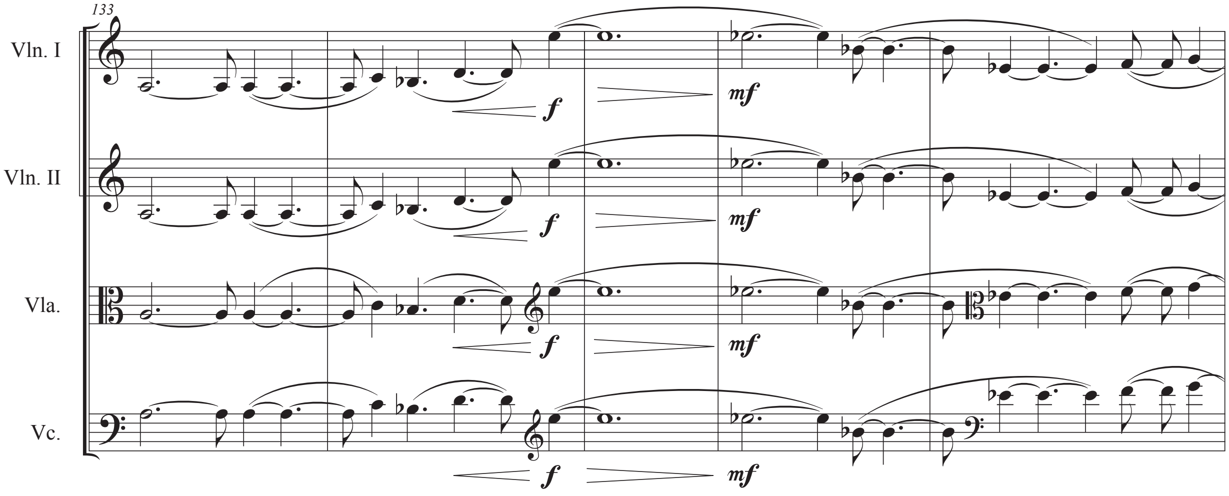

107
Vln. I
Vln. II
Vla.
Vc.
non vib.
ord.
non vib.
ord.
non vib.
ord.

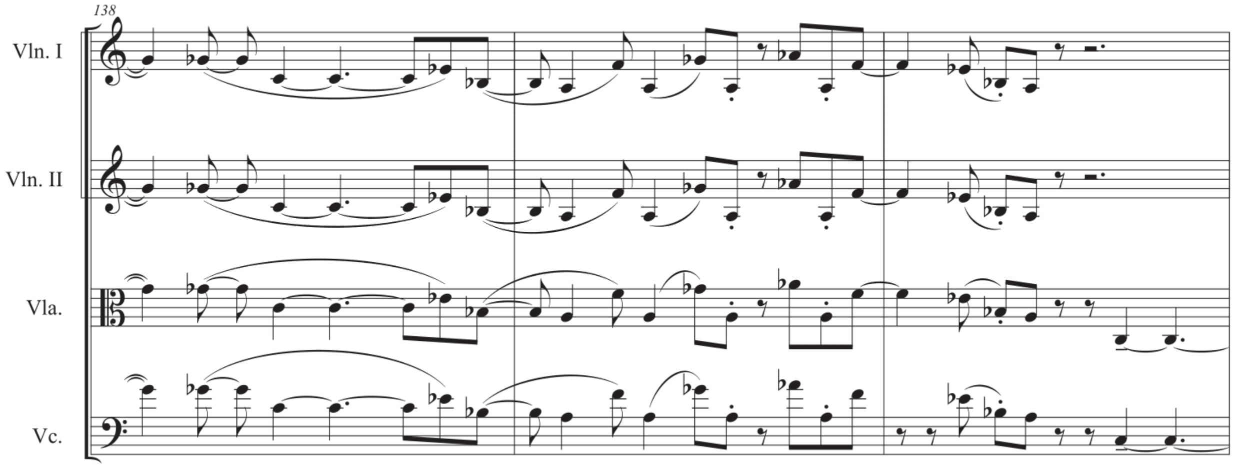

112
Vln. I
Vln. II
Vla.
Vc.
8va
ppp

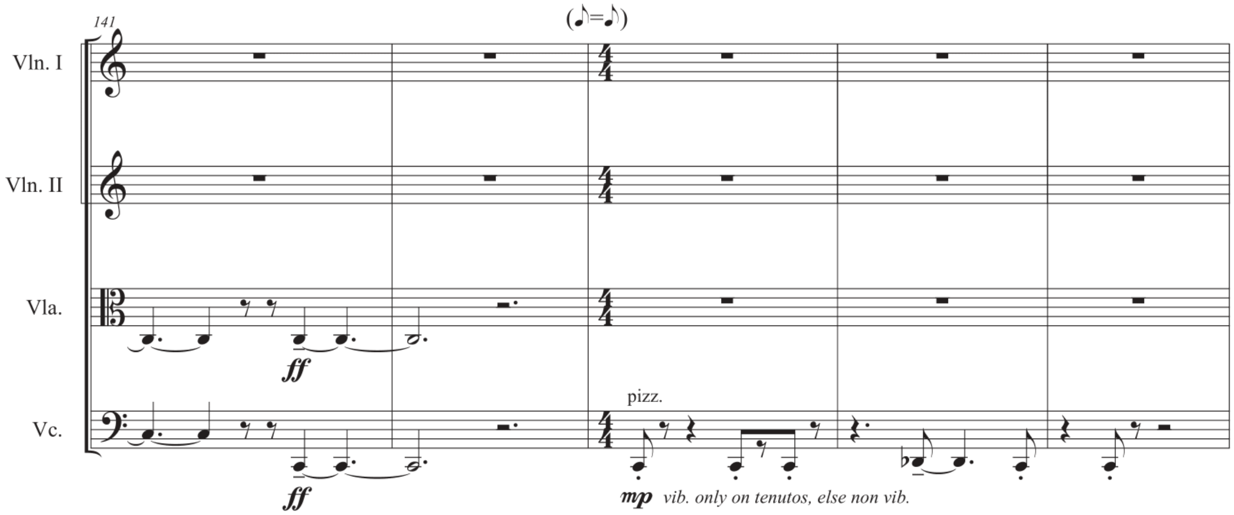

117
(8va)
8va
Vln. I
Vln. II
Vla.
Vc.
pp
ppp
pp

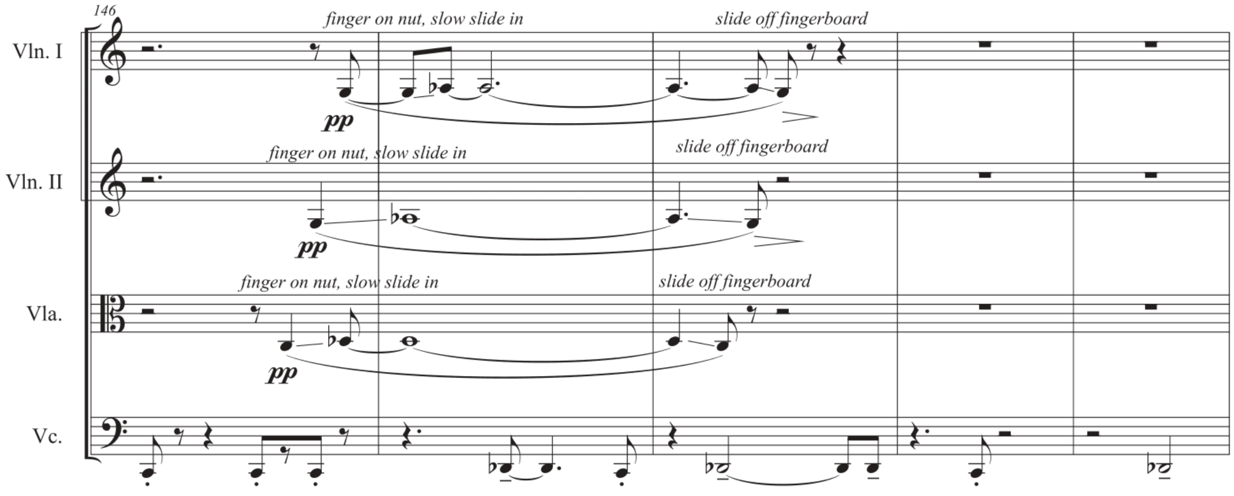

122
Vln. I
Vln. II
ppp
pp
Vla.
ppp
pp
Vc.

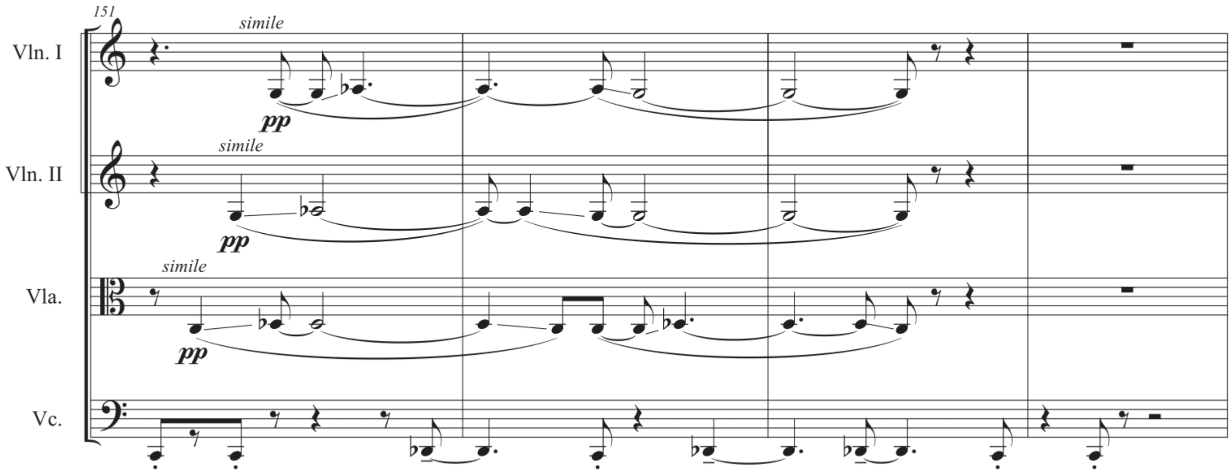

(♪=♪)
D
Molto espressivo
127
Vln. I
ord.
ppp
mf
Vln. II
ord.
ppp
mf
Vla.
ppp
mf
Vc.
arco
ppp
mf

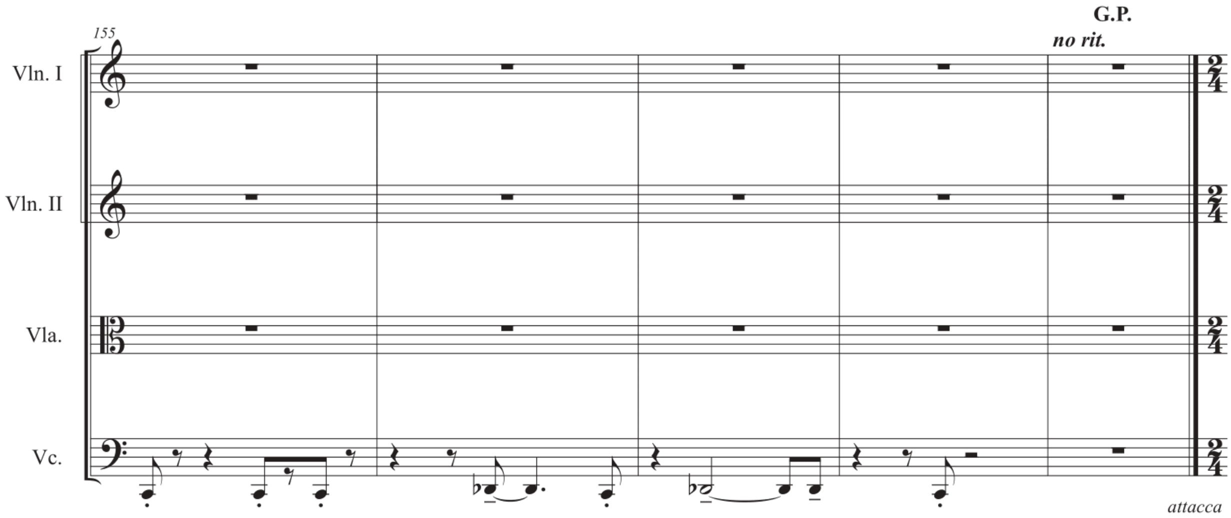

133
Vln. I
f
mf
Vln. II
f
mf
Vla.
f
mf
Vc.
f
mf


138
Vln. I
Vln. II
Vla.
Vc.

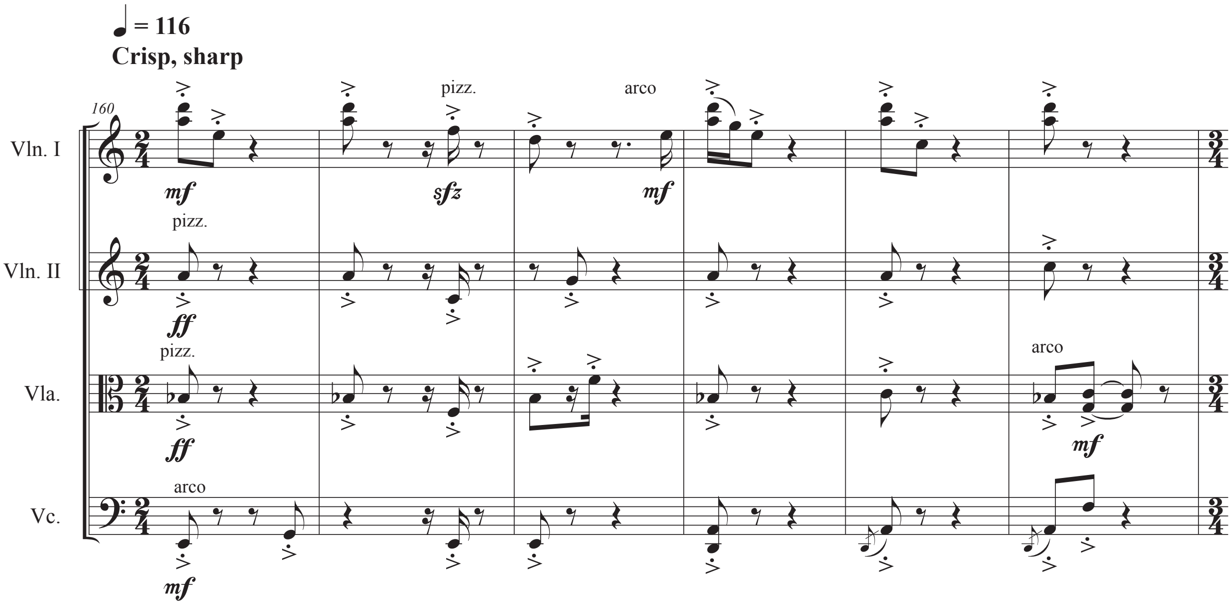

141
(♪=♪)
Vln. I
Vln. II
Vla.
Vc.
ff
ff
pizz.
mp vib. only on tenutos, else non vib.

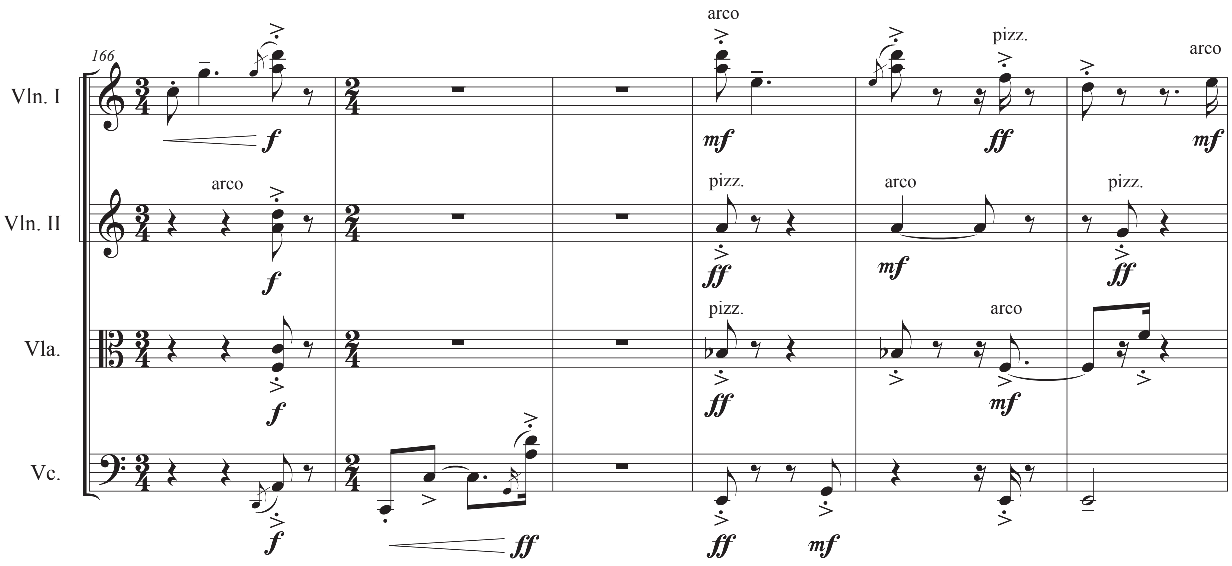

146
Vln. I
finger on nut, slow slide in
slide off fingerboard
pp
Vln. II
finger on nut, slow slide in
slide off fingerboard
pp
Vla.
finger on nut, slow slide in
slide off fingerboard
pp
Vc.


151
Vln. I
simile
pp
Vln. II
simile
pp
Vla.
simile
pp
Vc.

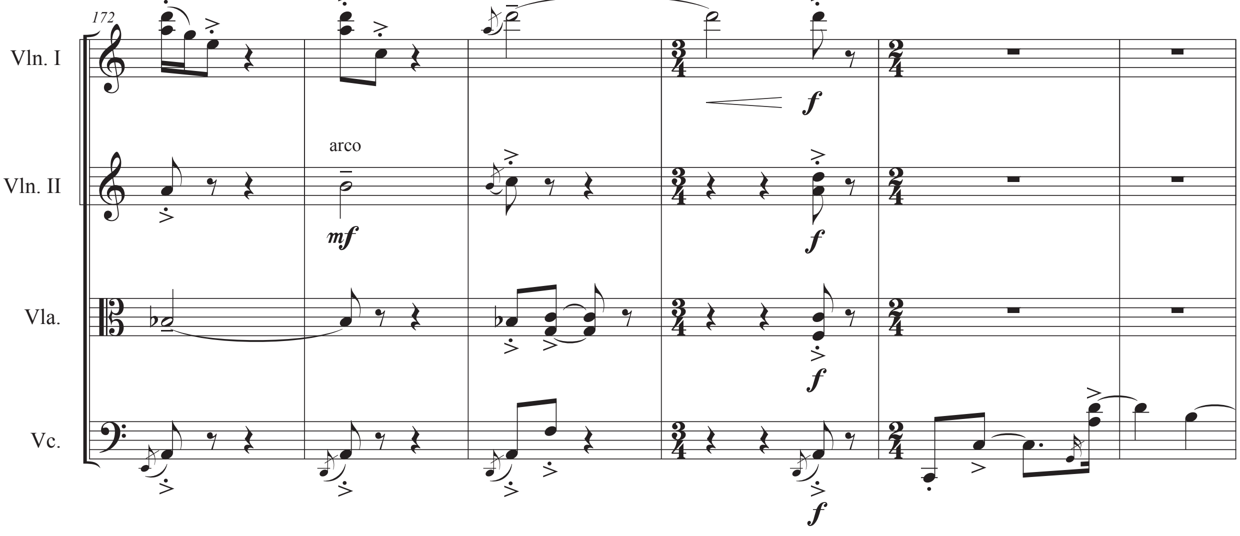

155
G.P.
no rit.
Vln. I
Vln. II
Vla.
Vc.
attacca

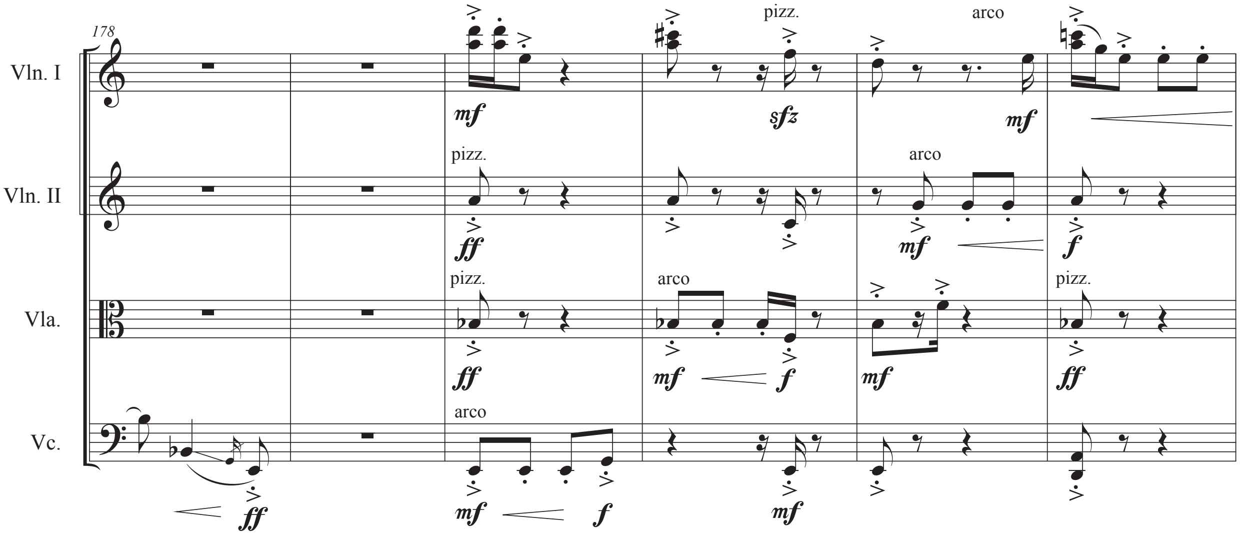
III
Neural Networks

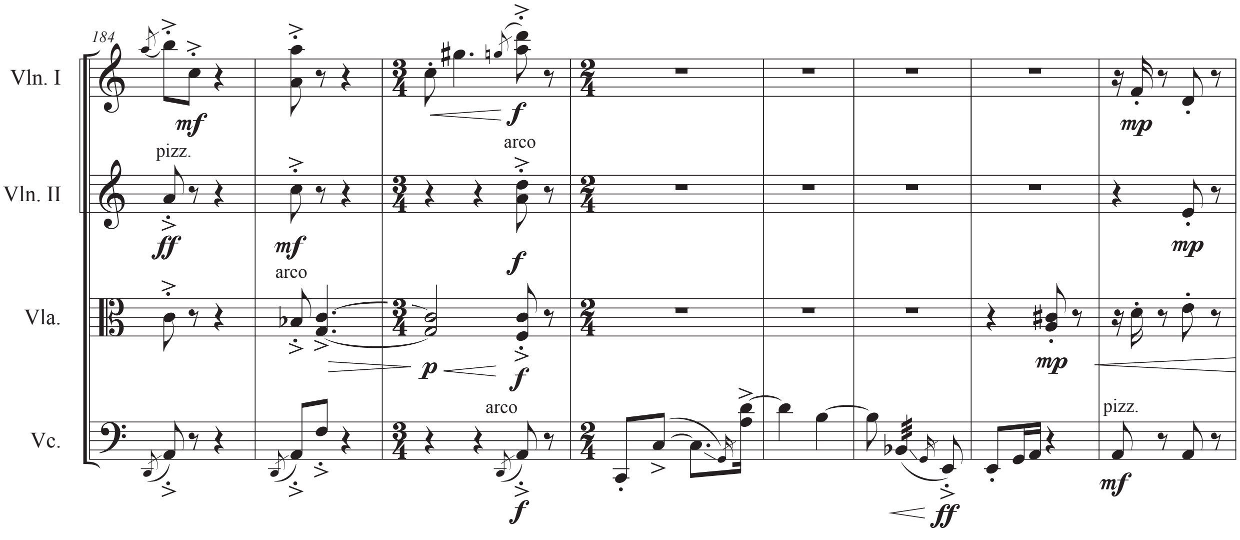


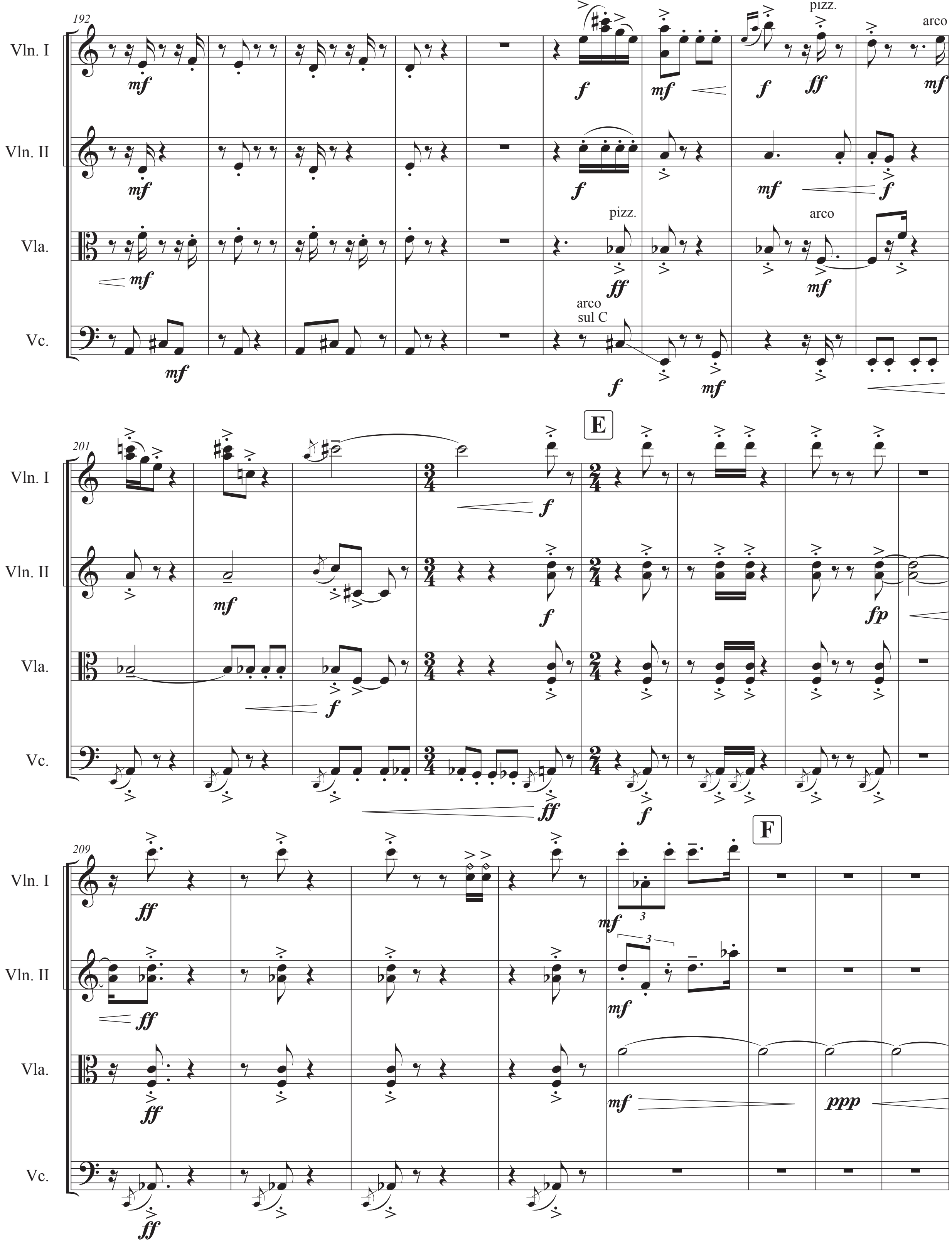


17

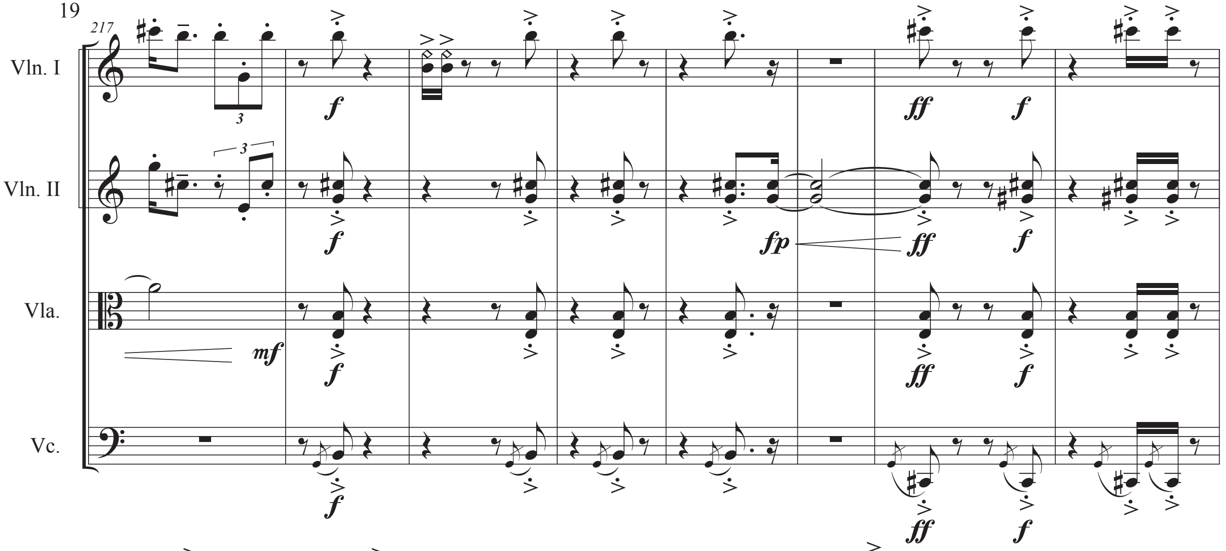

172
Vln. I
Vln. II
Vla.
Vc.
arco
mf
f
f
f
f

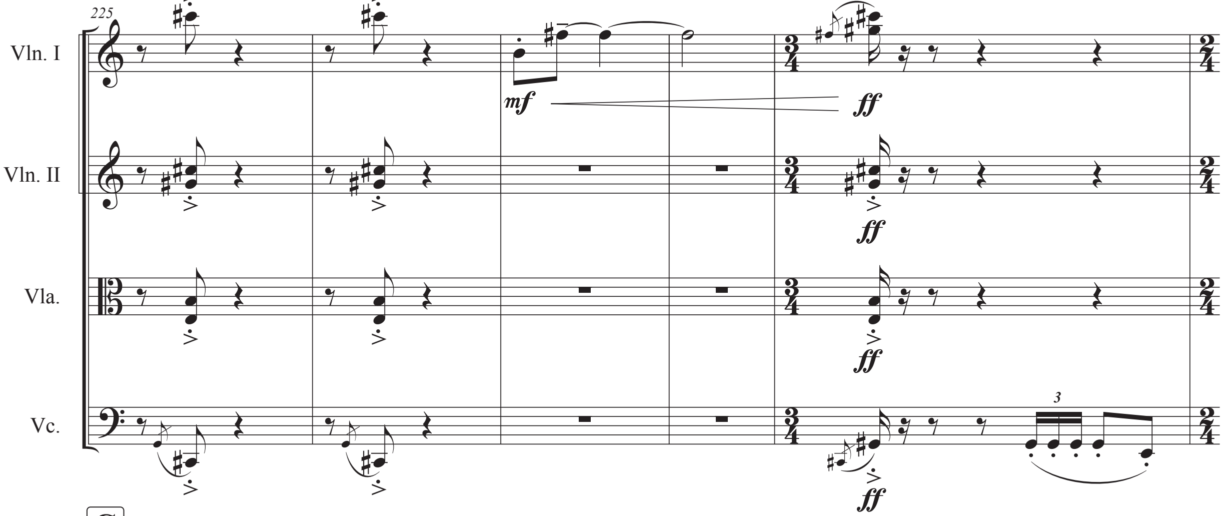

178
Vln. I
Vln. II
Vla.
Vc.
pizz.
arco
pizz.
arco
pizz.
arco
pizz.
arco
mf
sfz
mf
ff
mf
f
ff
mf
f
mf
ff
ff
mf
f
mf


184
Vln. I
Vln. II
Vla.
Vc.
mf
f
mp
pizz.
arco
ff
mf
f
mp
arco
p
f
mp
arco
pizz.
f
ff
mf

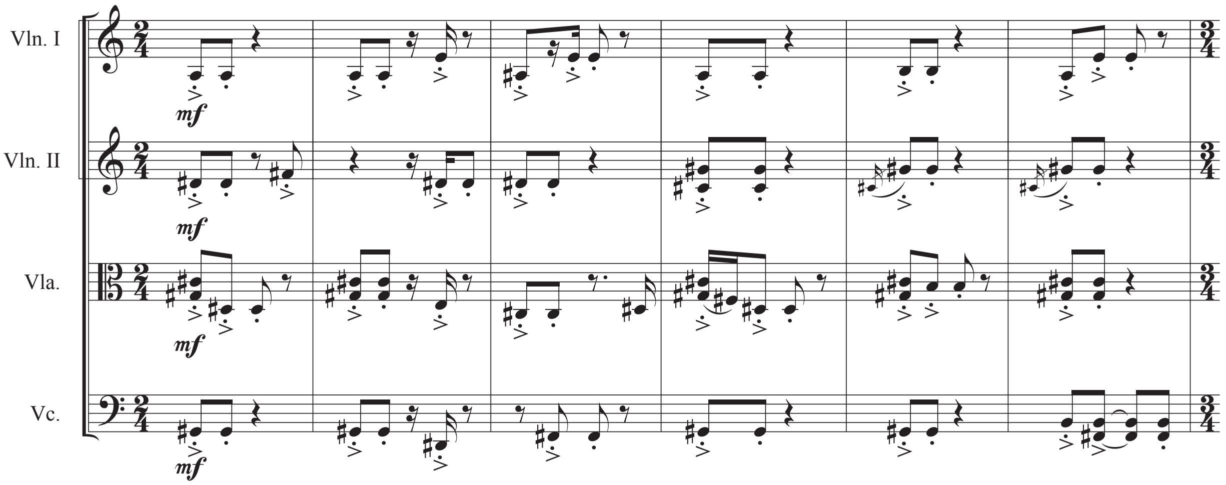

192
Vln. I
Vln. II
Vla.
Vc.
mf
mf
mf
mf
pizz.
arco
sul C
f
f
ff
f
mf
mf
f
ff
mf
pizz.
arco
mf
arco
mf
f
201
E
Vln. I
Vln. II
Vla.
Vc.
mf
f
f
ff
f
f
fp
209
F
Vln. I
Vln. II
Vla.
Vc.
ff
ff
ff
ff
mf
3
mf
3
mf
ppp

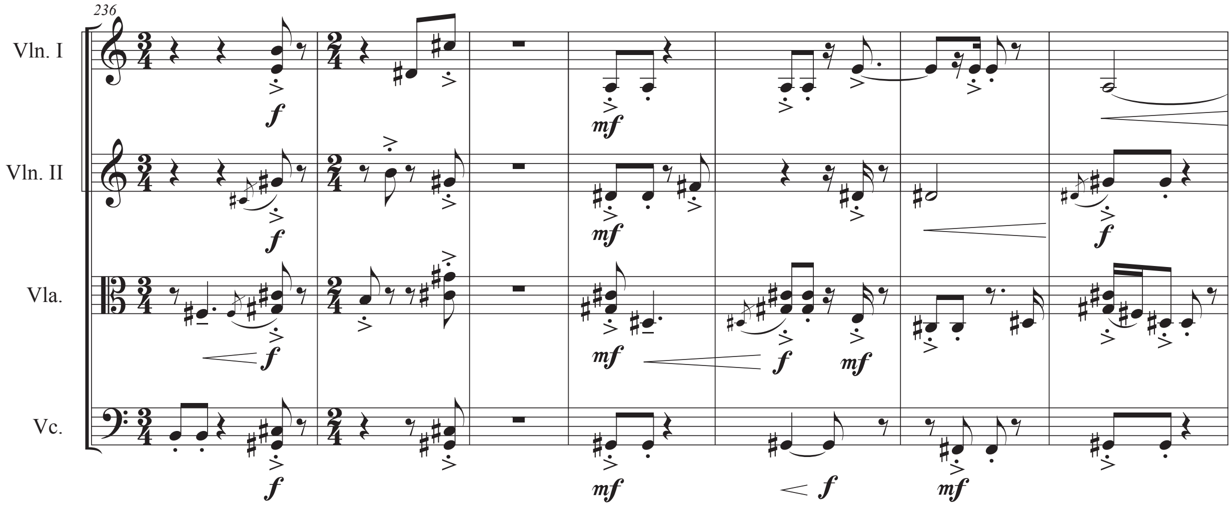
19

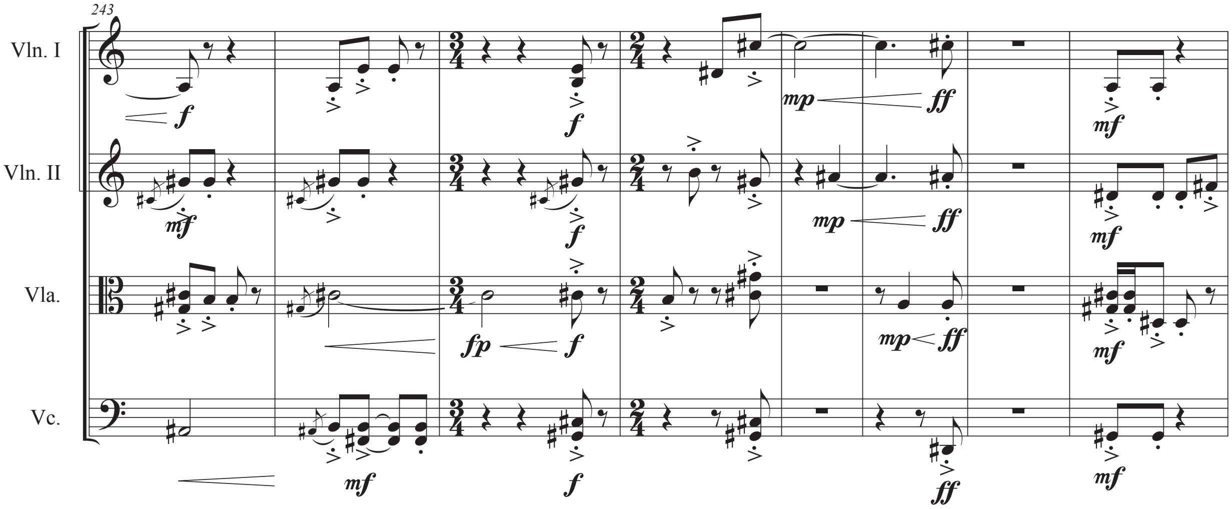

217
Vln. I
Vln. II
Vla.
Vc.
f
ff
f
fp
ff
mf
3

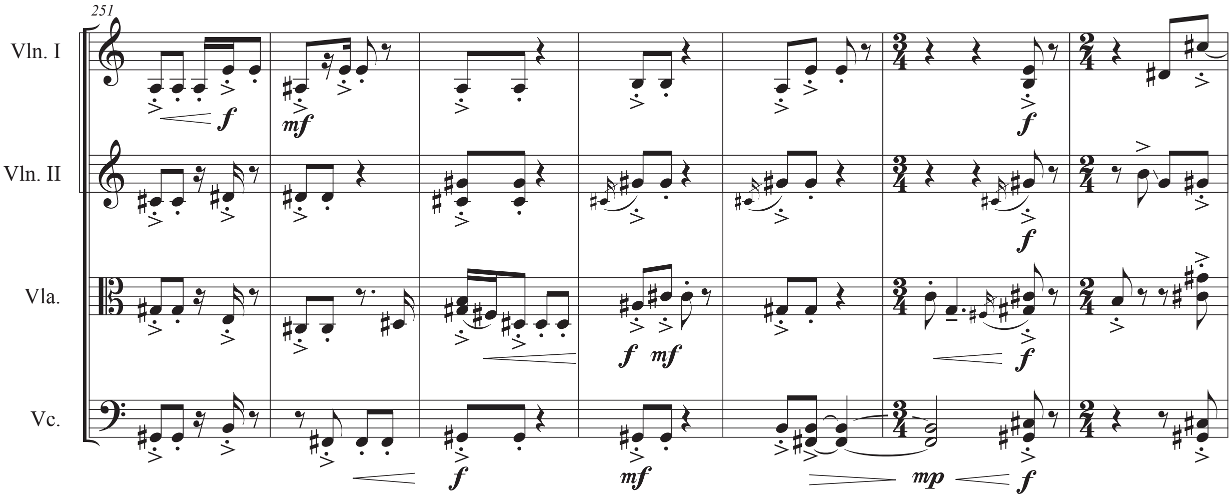

225
Vln. I
Vln. II
Vla.
Vc.
mf
ff
3
3/4
2/4

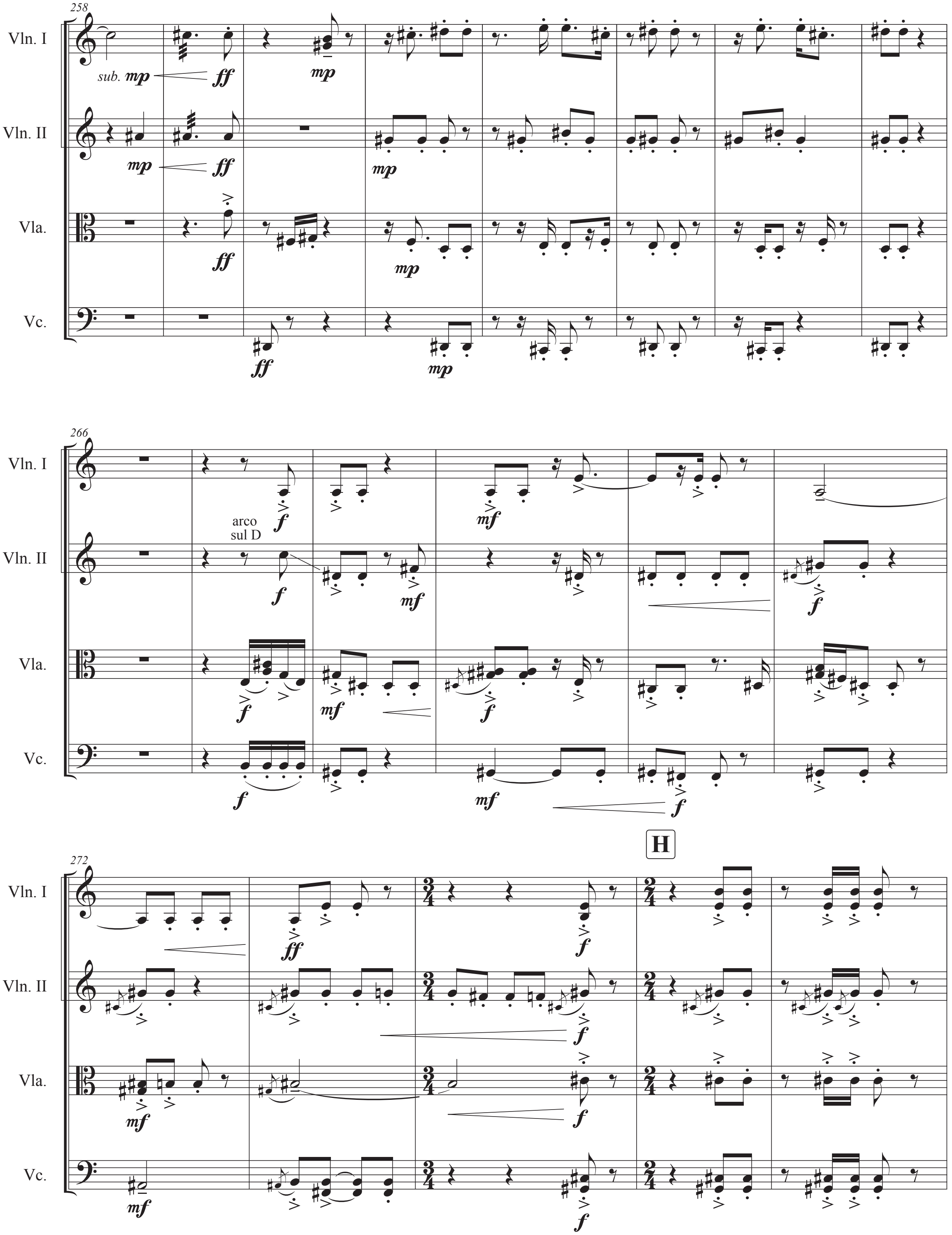
G

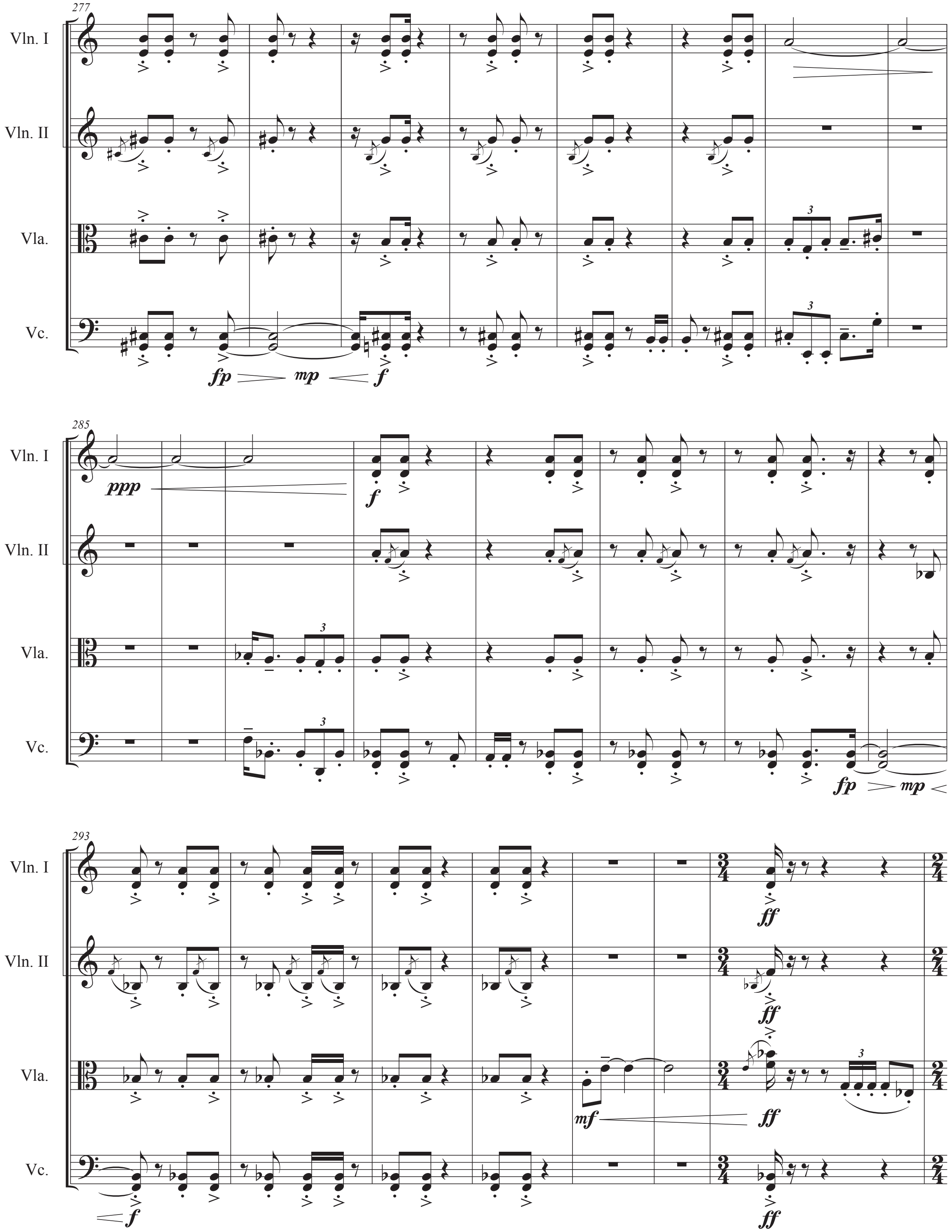

Vln. I
Vln. II
Vla.
Vc.
2/4
3/4
mf

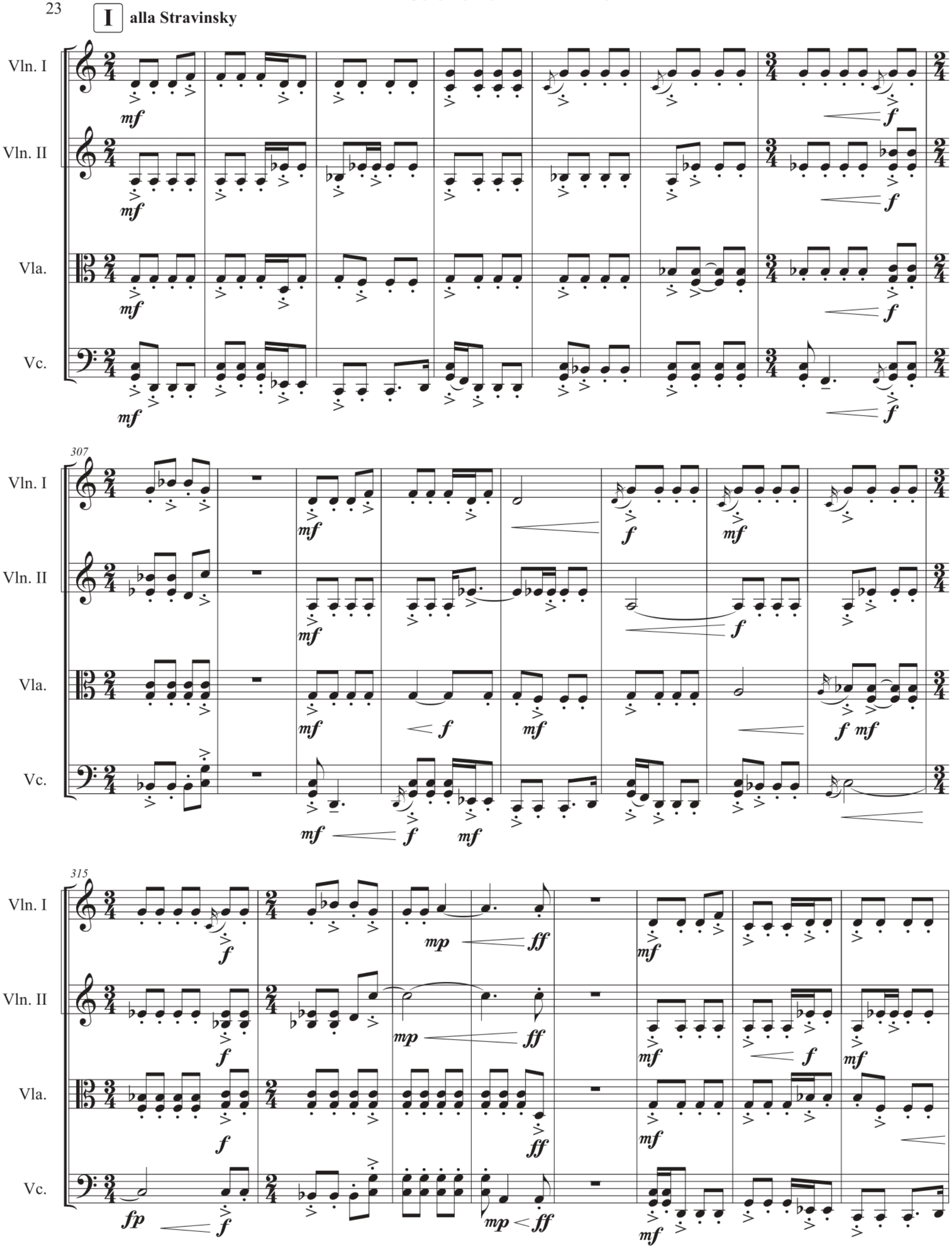

236
Vln. I
Vln. II
Vla.
Vc.
f
mf
mp
f
mf
f
mf

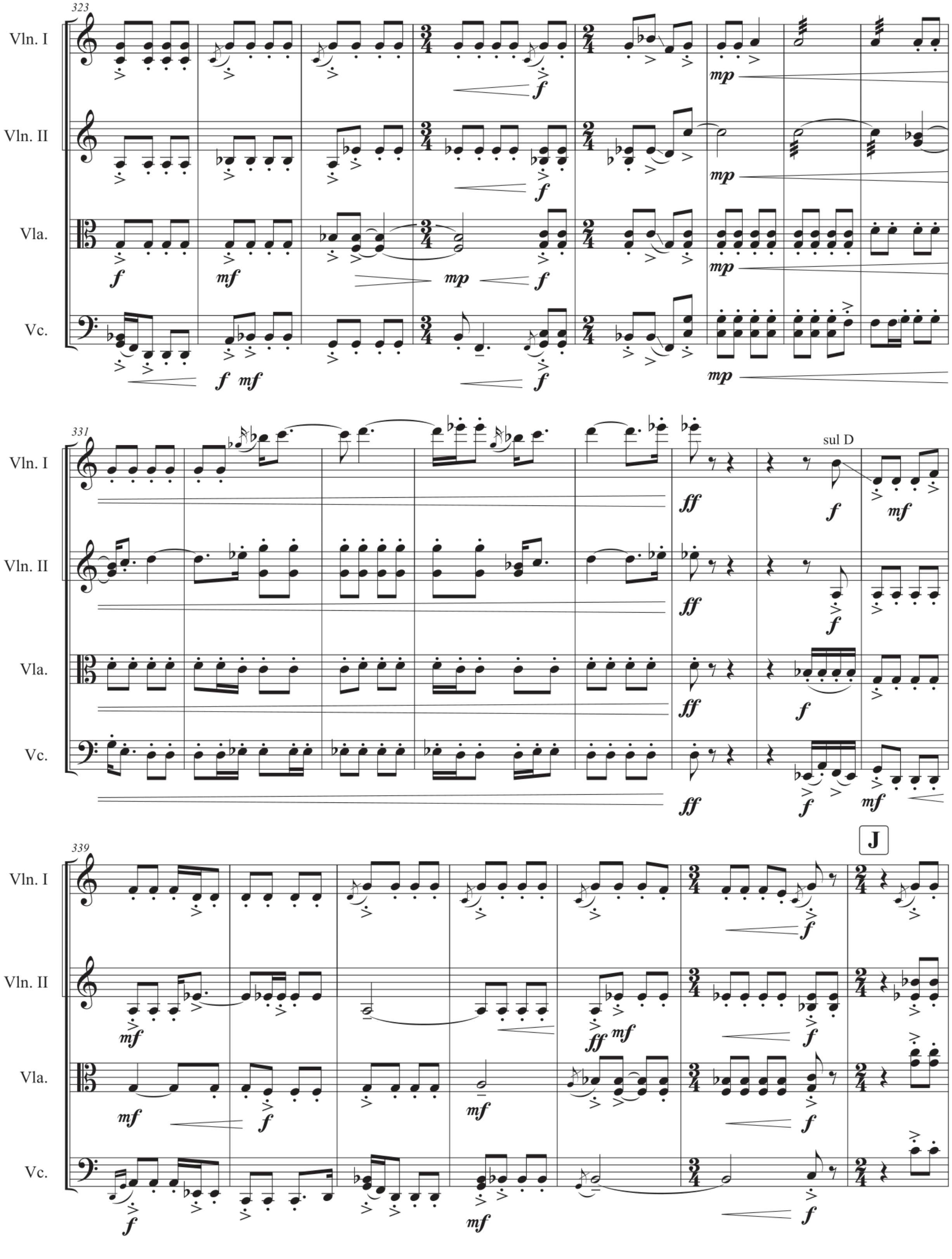

243
Vln. I
Vln. II
Vla.
Vc.
f
mp
ff
mf
fp
f
mp
ff

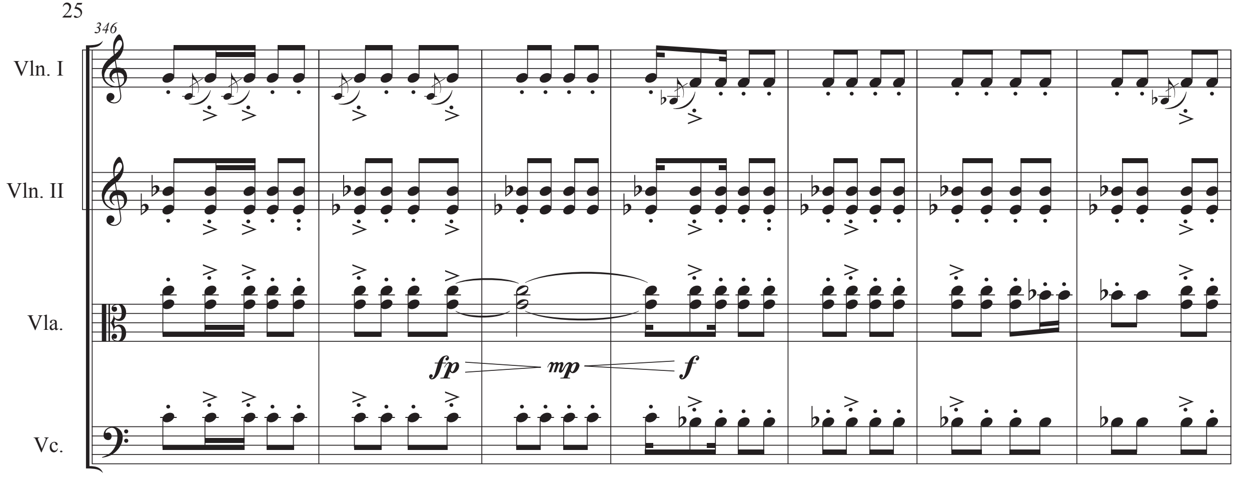

251
Vln. I
Vln. II
Vla.
Vc.
f
mf
f
mf
mp
f

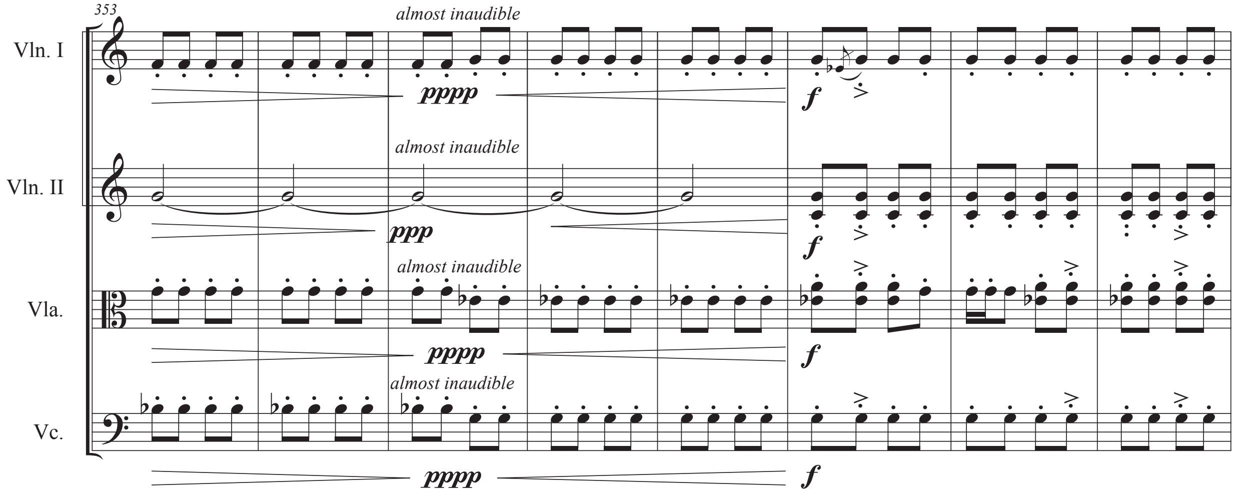

258
Vln. I
Vln. II
Vla.
Vc.
sub. mp
ff
mp
mp
ff
mp
ff
ff
mp
mp
266
arco
sul D
f
f
mf
mf
f
mf
f
f
mf
f
H
272
ff
f
f
mf
f
mf
f

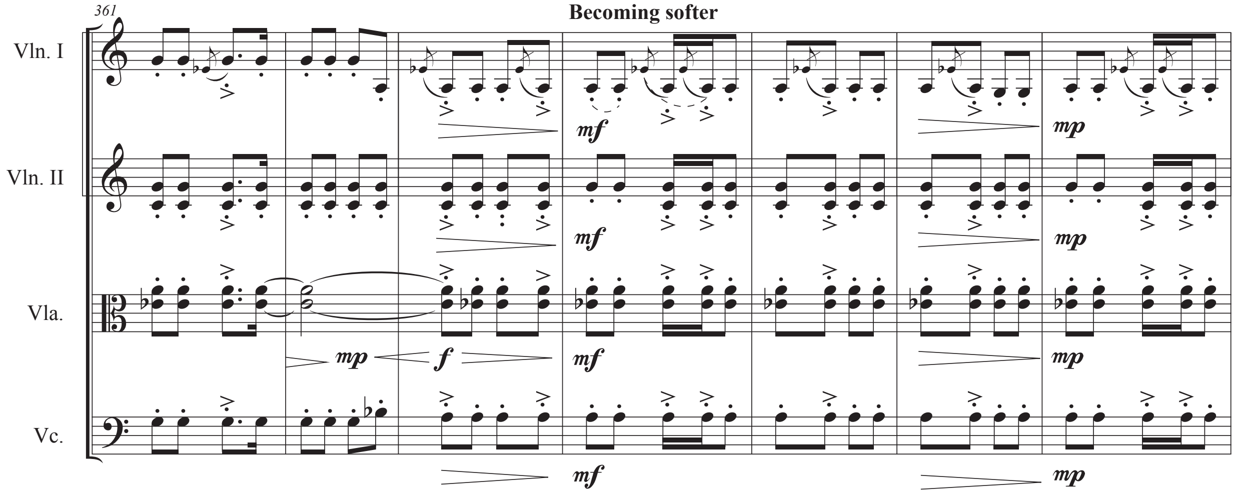

277
Vln. I
Vln. II
Vla.
Vc.
3
fp
mp
f
285
ppp
f
3
fp
mp
293
ff
mf
ff
3
ff

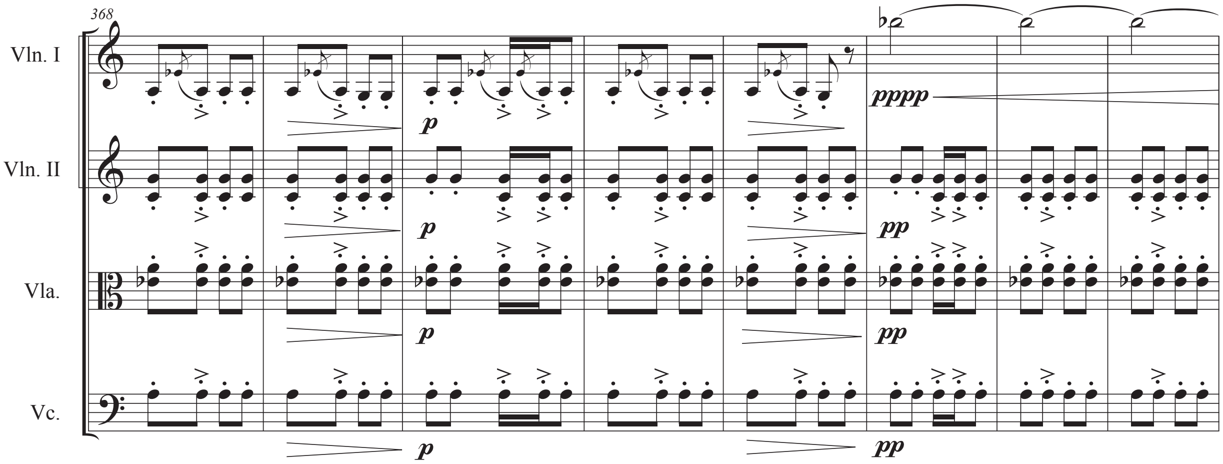

I
alla Stravinsky
Vln. I
Vln. II
Vla.
Vc.
mf
f
307
315
mp
ff
fp

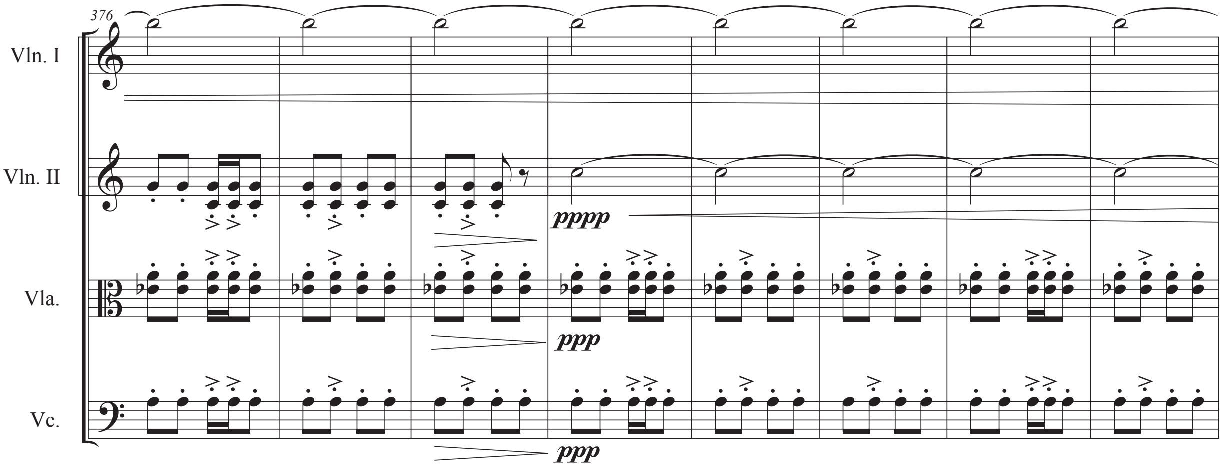

323
Vln. I
Vln. II
Vla.
Vc.
mp
f
mf
331
sul D
ff
339
J

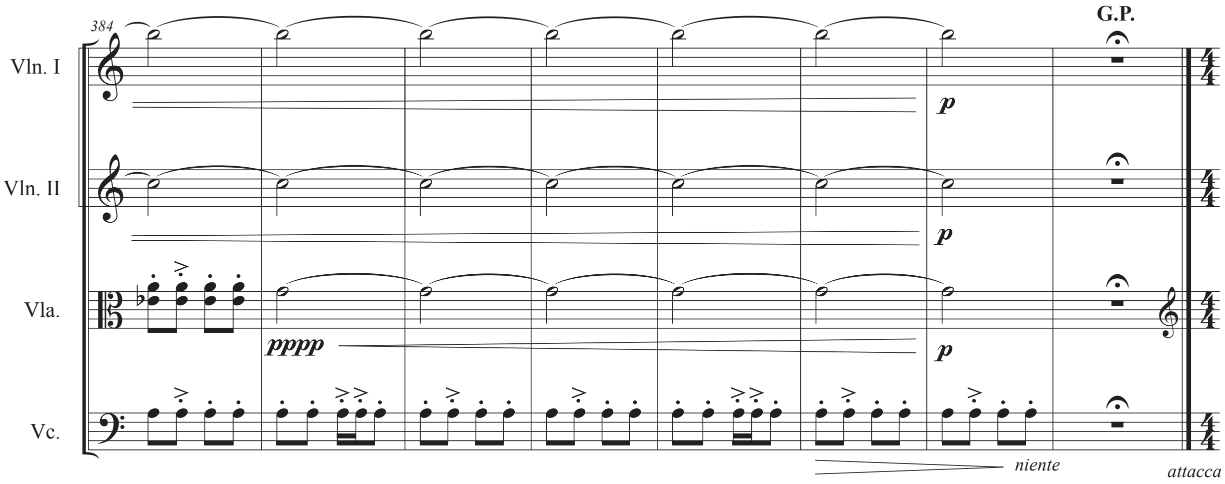

346
Vln. I
Vln. II
Vla.
Vc.
fp
mp
f


353
Vln. I
almost inaudible
pppp
f
Vln. II
almost inaudible
ppp
f
Vla.
almost inaudible
pppp
f
Vc.
almost inaudible
pppp
f


361
Becoming softer
Vln. I
mf
mp
Vln. II
mf
mp
Vla.
mp
f
mf
mp
Vc.
mf
mp

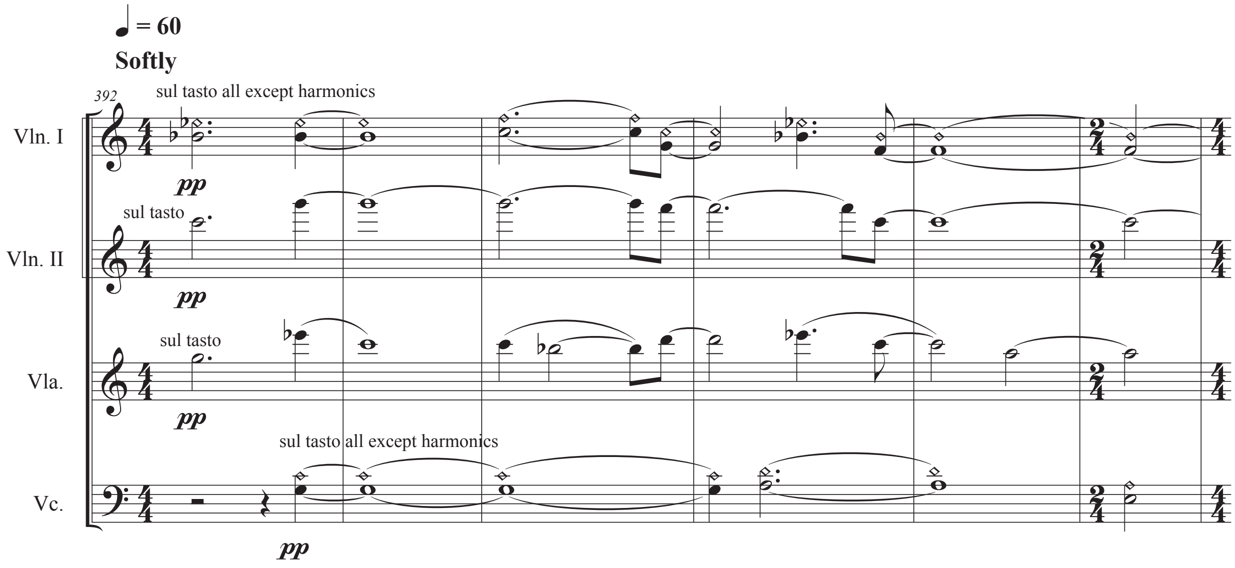

368
Vln. I
Vln. II
Vla.
Vc.
p
pppp
pp

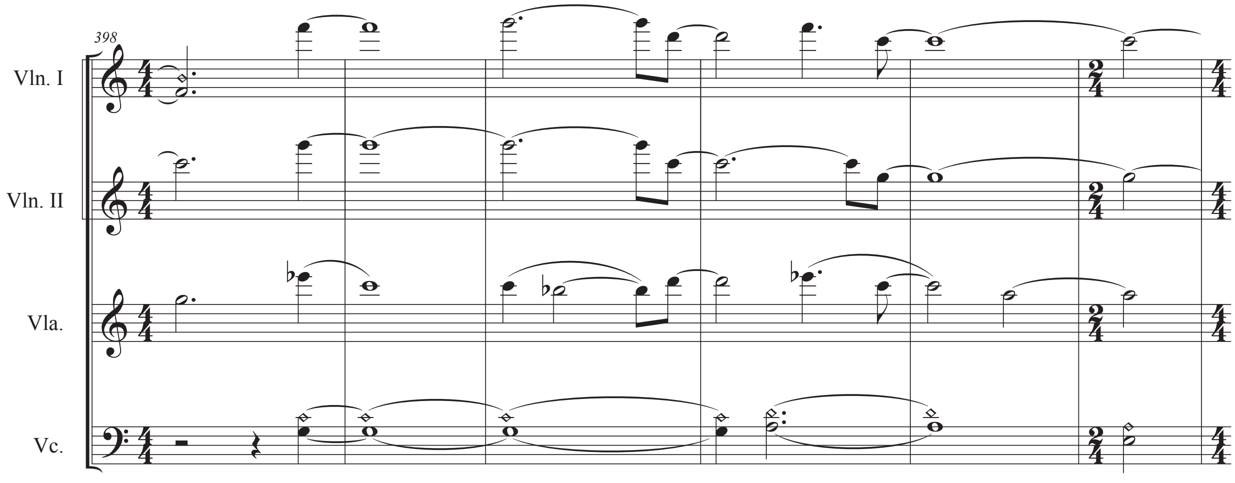

376
Vln. I
Vln. II
Vla.
Vc.
pppp
ppp

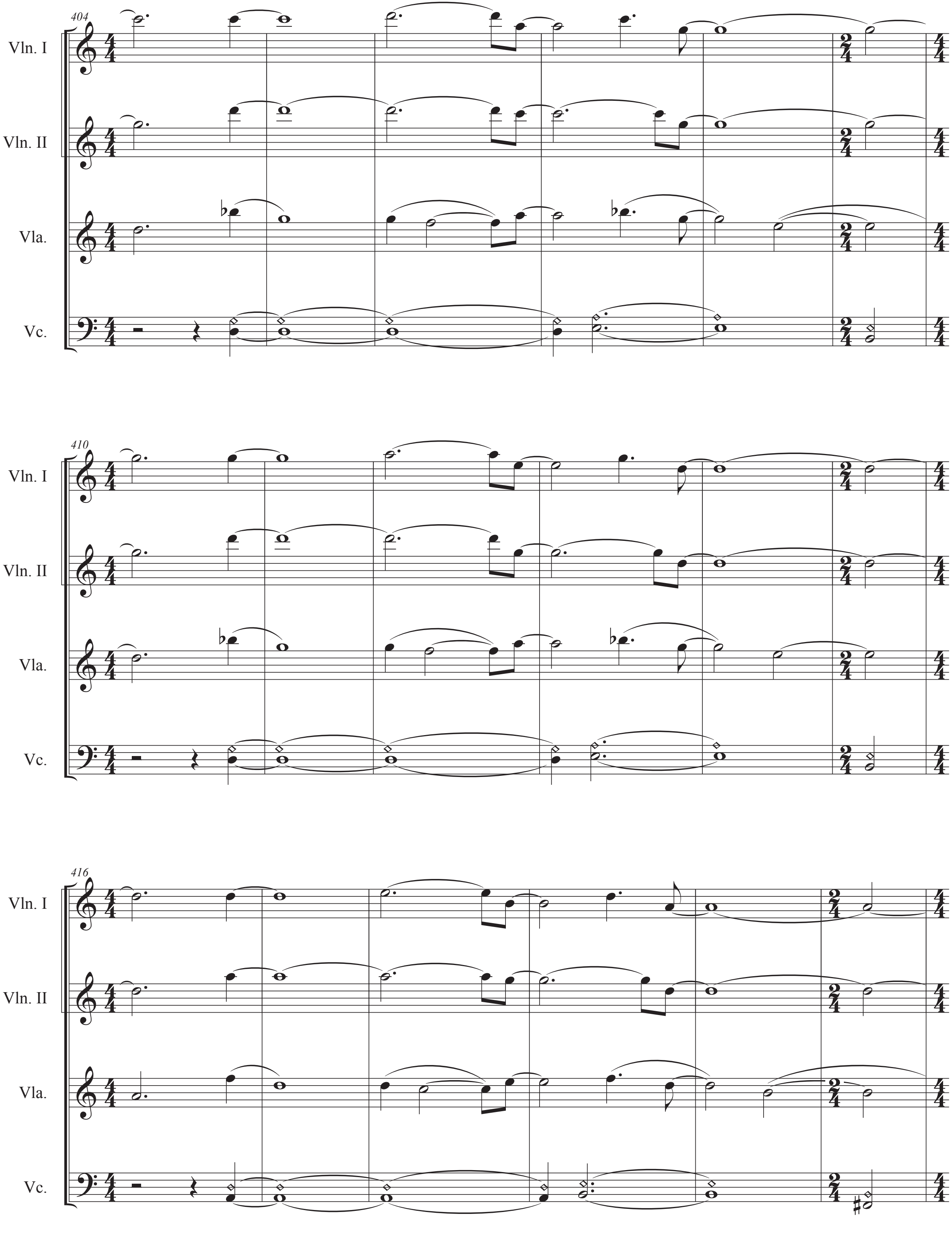

384
G.P.
Vln. I
Vln. II
Vla.
Vc.
p
pppp
niente
attacca

IV

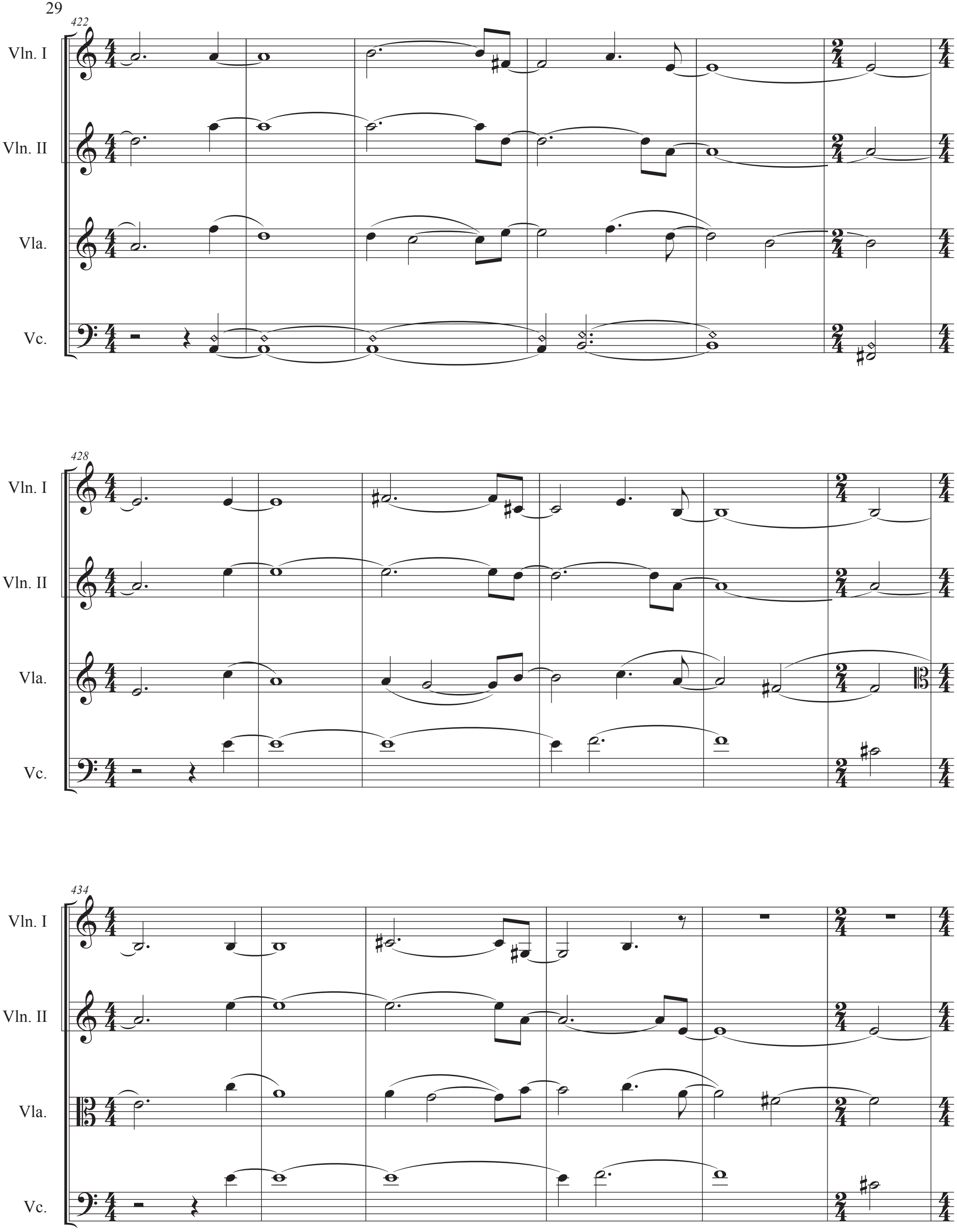
Loss & Gradient Descent



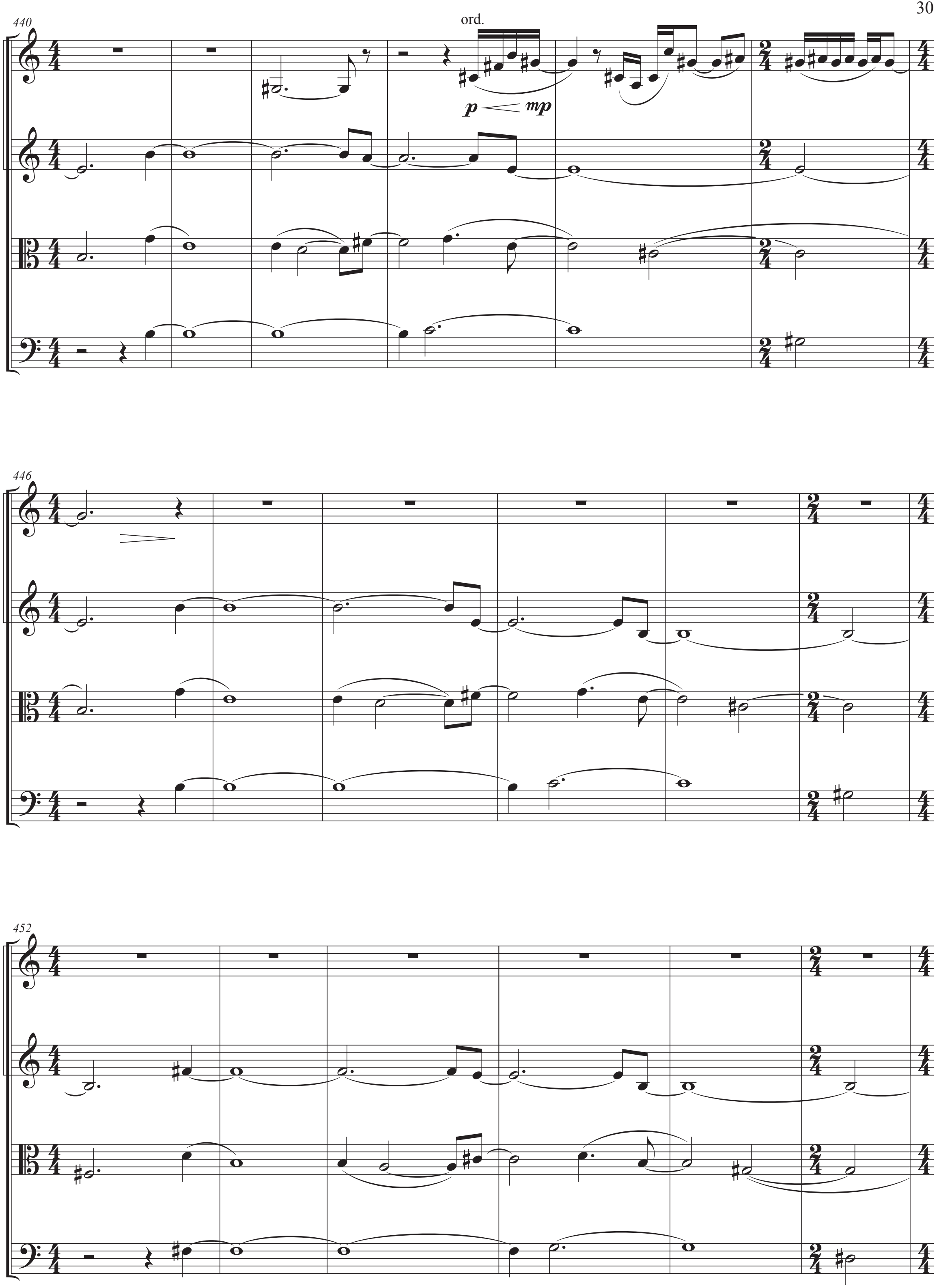


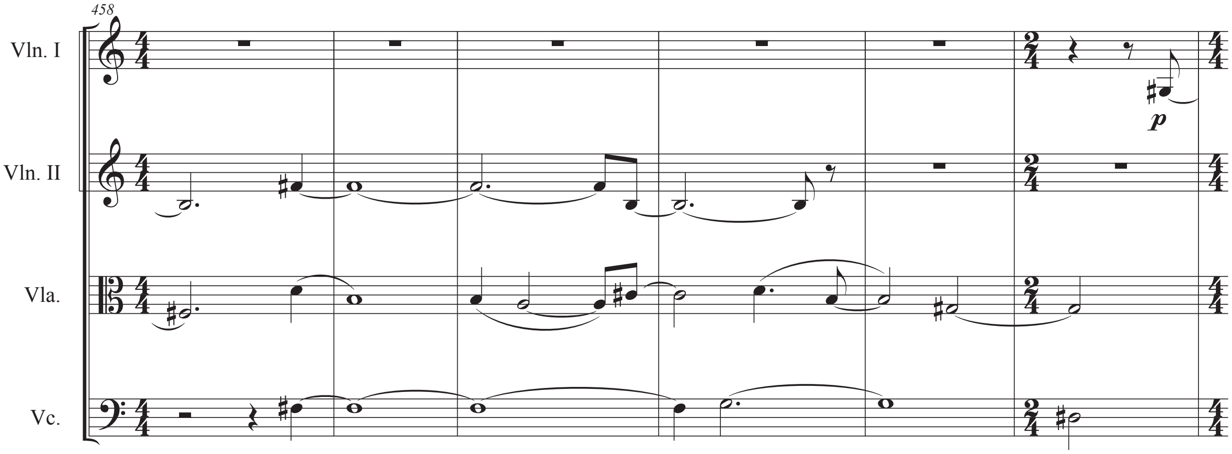

404
Vln. I
Vln. II
Vla.
Vc.
410
Vln. I
Vln. II
Vla.
Vc.
416
Vln. I
Vln. II
Vla.
Vc.

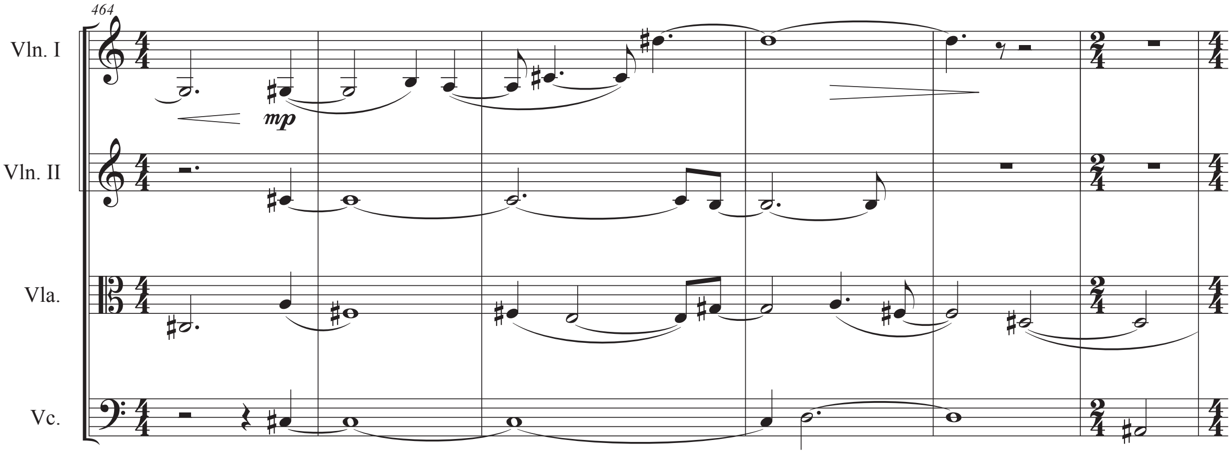
MUSIC FOR SELF-ATTENTION

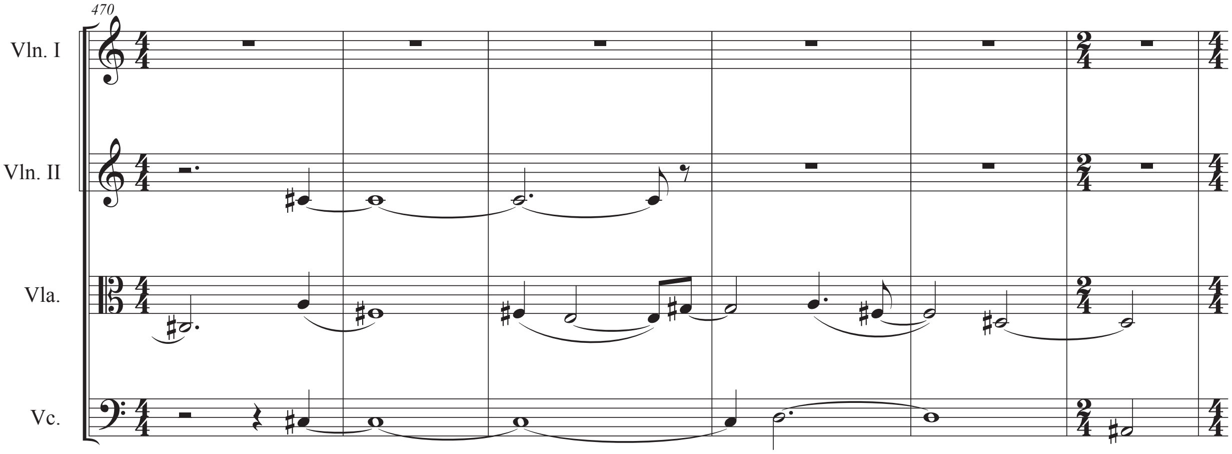


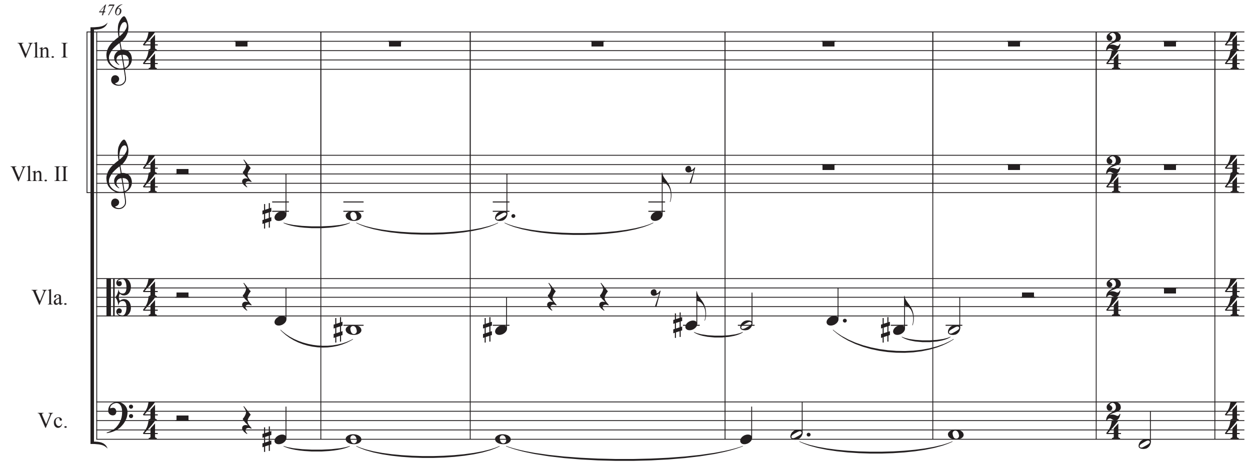
MUSIC FOR SELF-ATTENTION

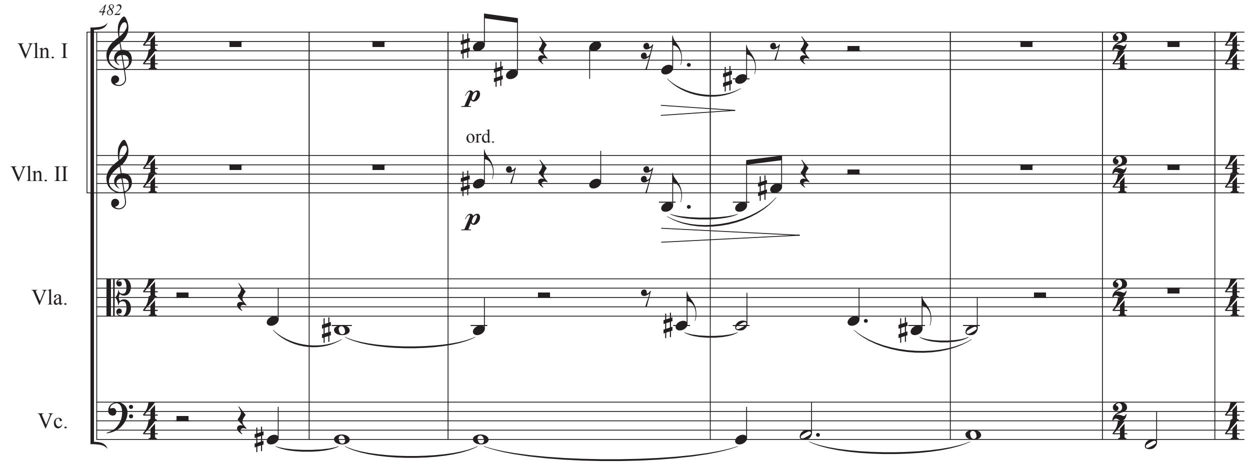


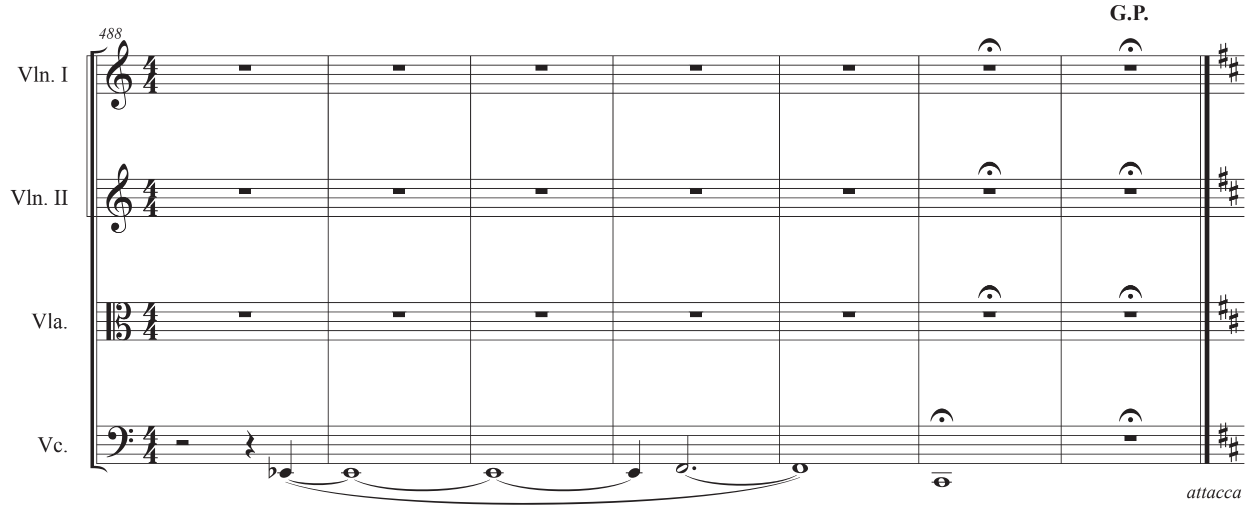

458
Vln. I
Vln. II
Vla.
Vc.
p


464
Vln. I
Vln. II
Vla.
Vc.
mp

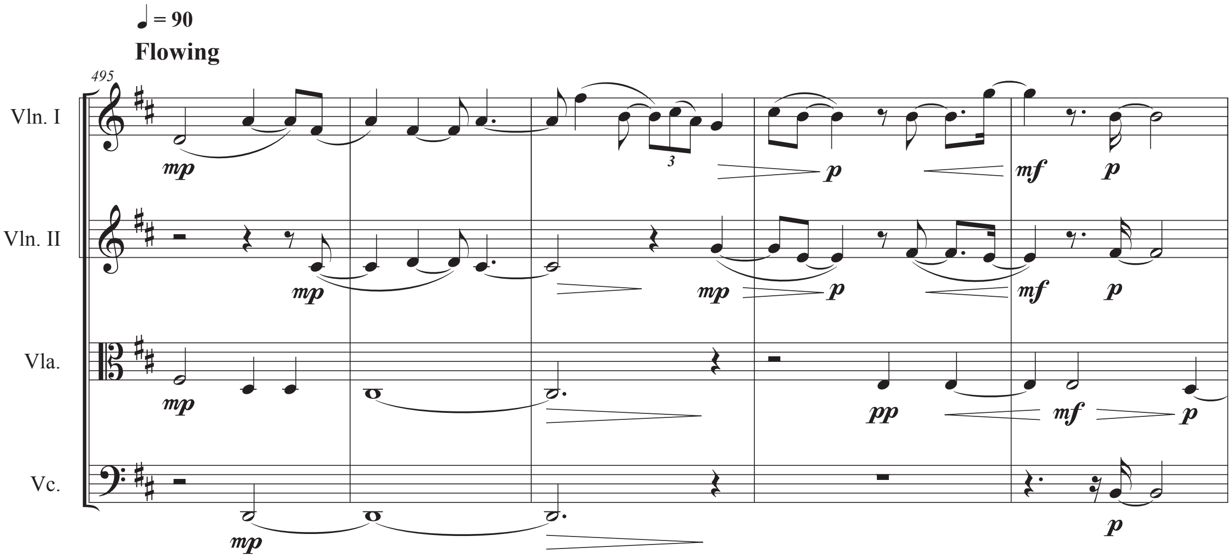

470
Vln. I
Vln. II
Vla.
Vc.

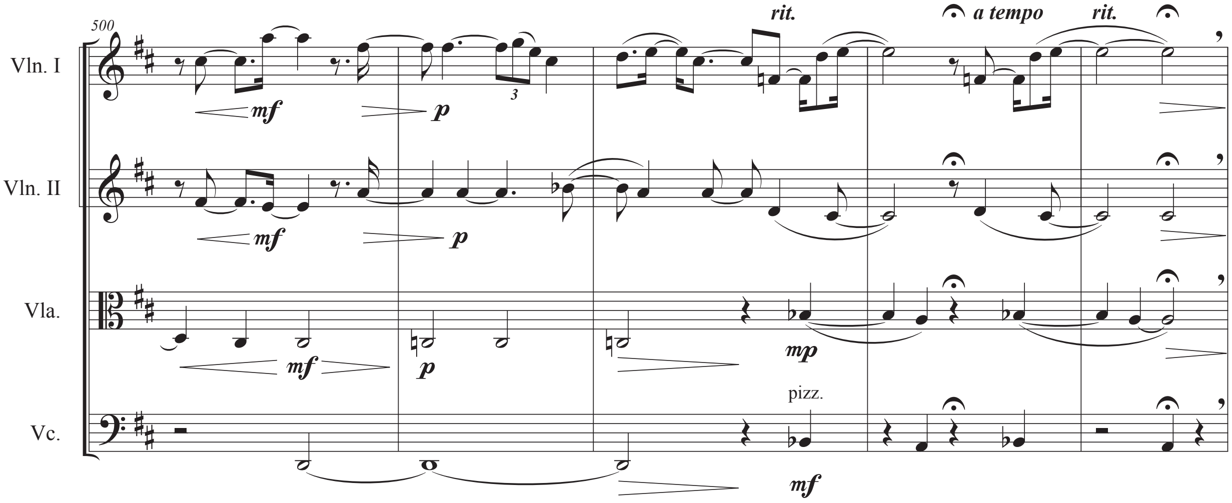

476
Vln. I
Vln. II
Vla.
Vc.

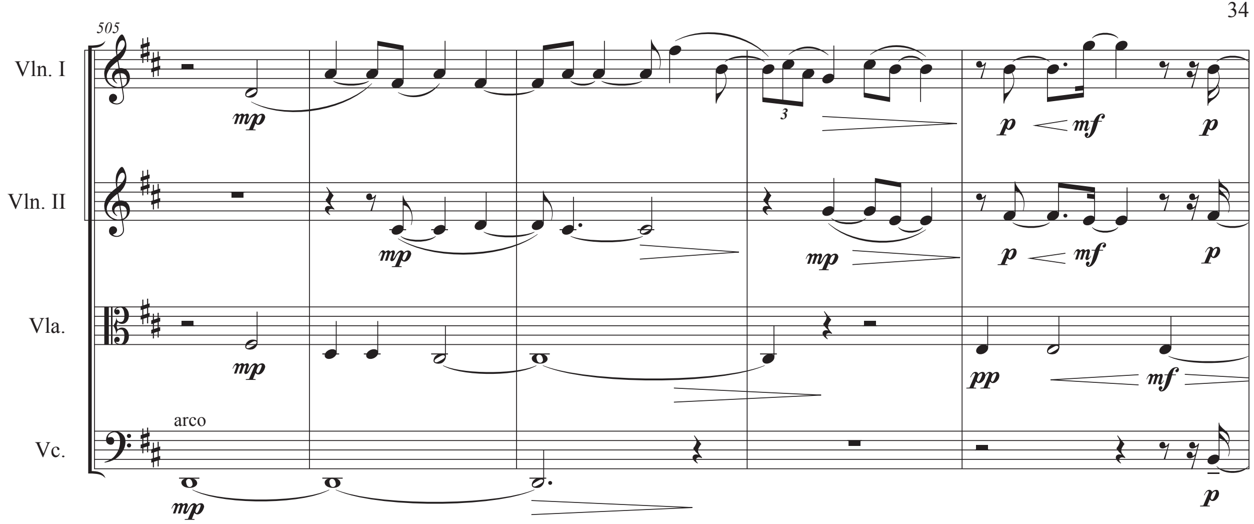

482
Vln. I
Vln. II
Vla.
Vc.
p
ord.
p

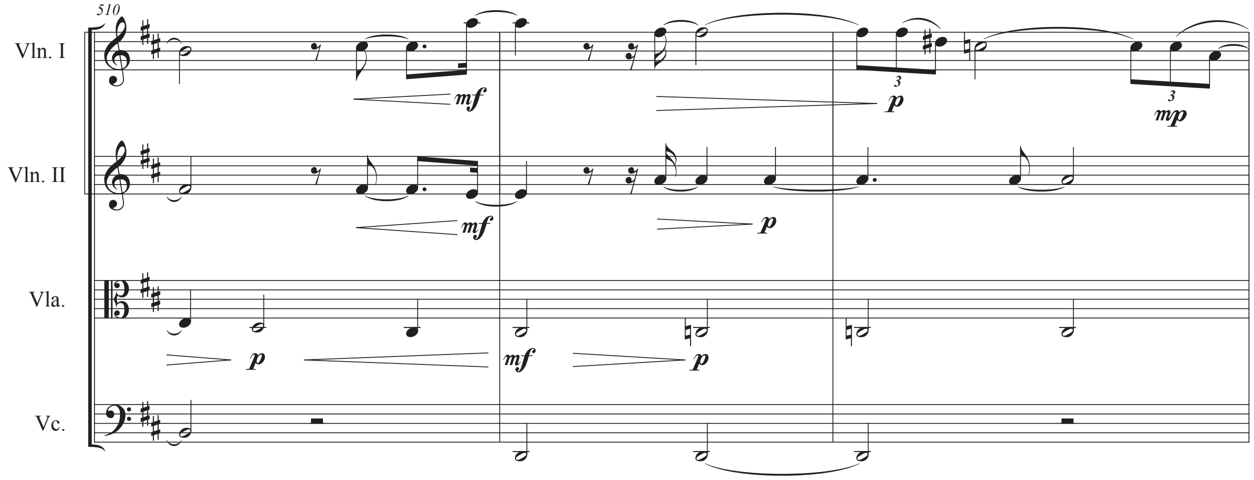

488
G.P.
Vln. I
Vln. II
Vla.
Vc.
attacca

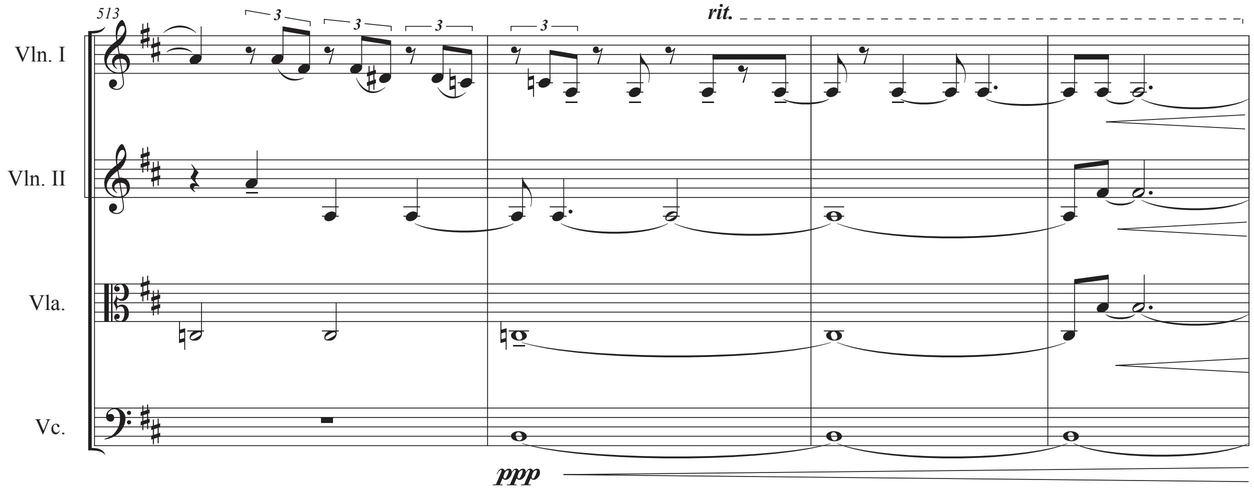
V
Hyperparameter Search

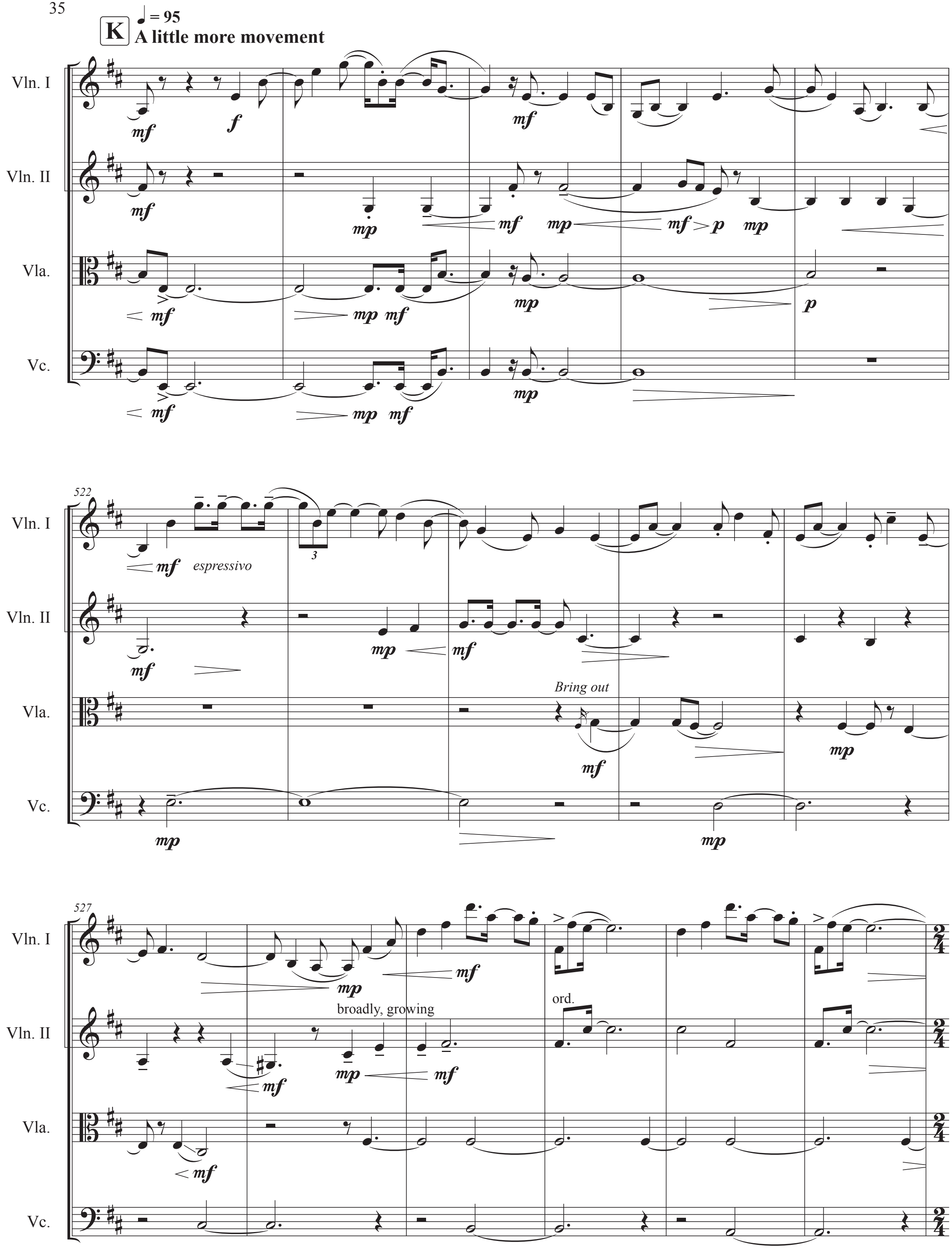


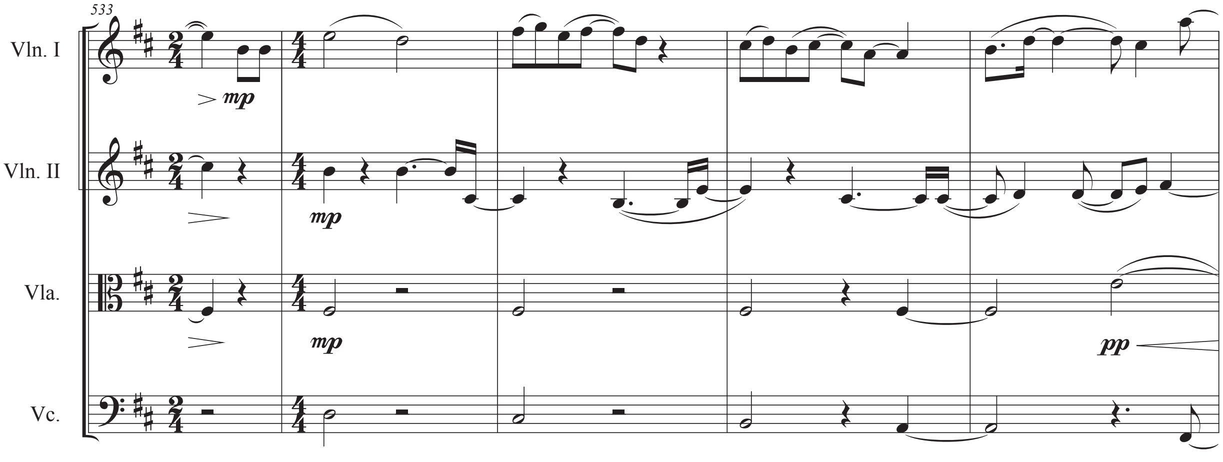


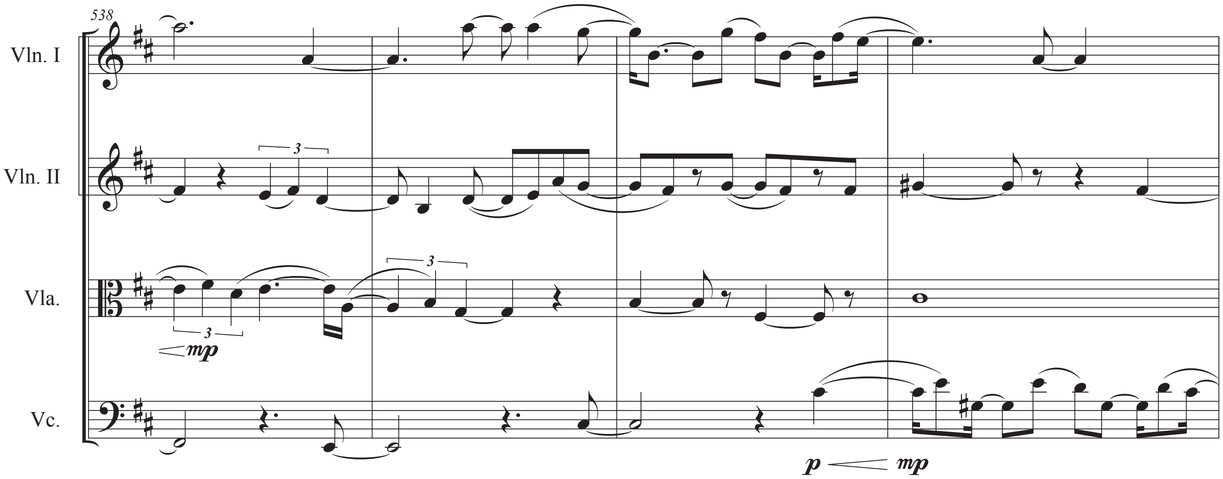

505
Vln. I
Vln. II
Vla.
Vc.
arco
mp
mp
mp
mp
3
mp
p
mf
p
p
mf
p
pp
mf
p

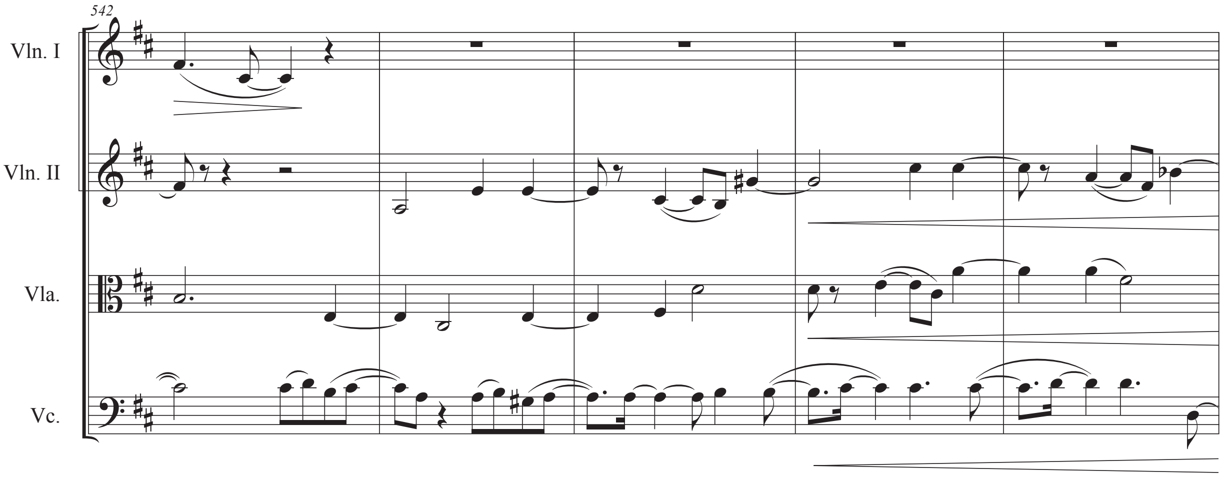

510
Vln. I
Vln. II
Vla.
Vc.
mf
3
p
3
mp
mf
p
p
mf
p

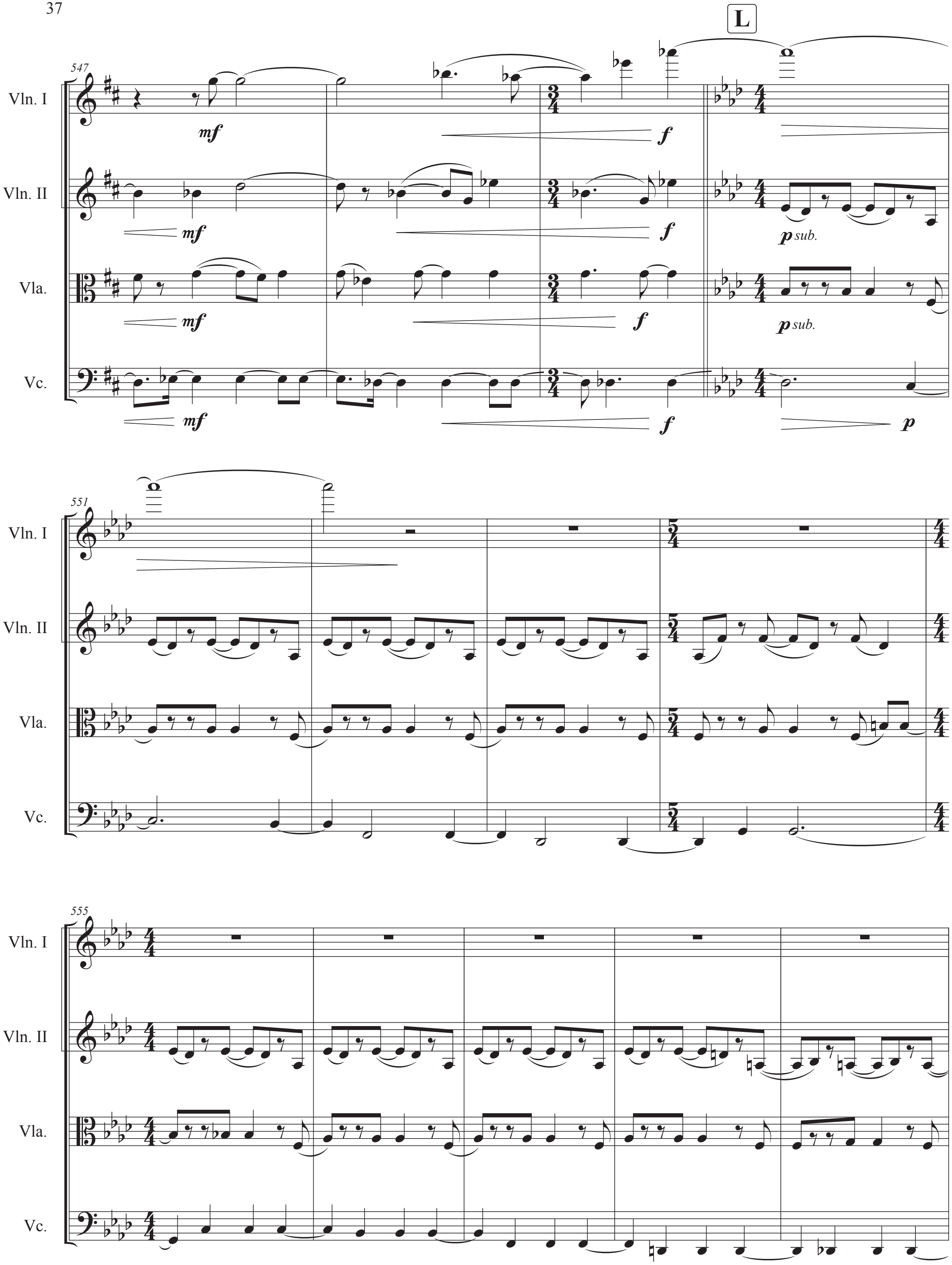

513
Vln. I
Vln. II
Vla.
Vc.
3
3
3
3
rit.
ppp

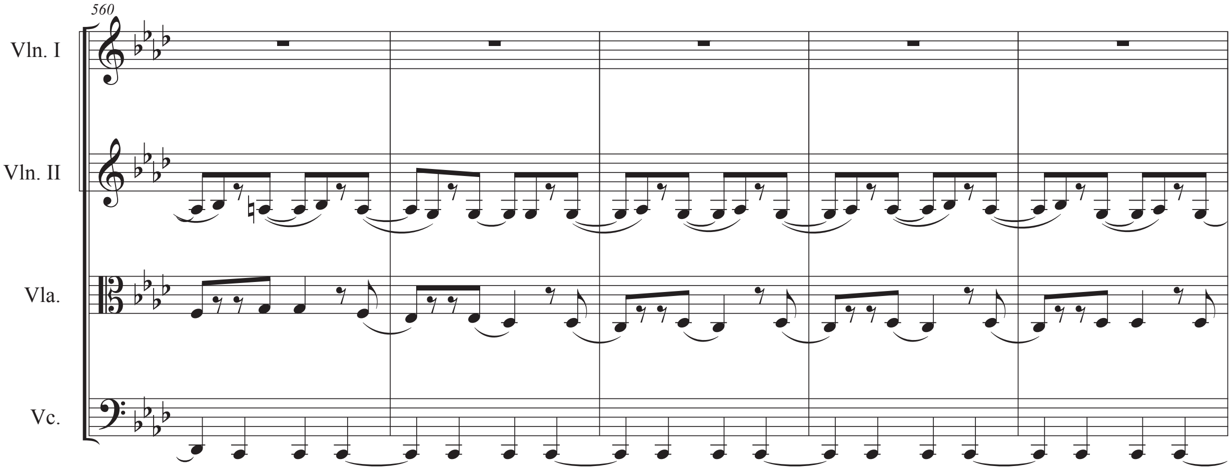

K
♩ = 95
A little more movement
Vln. I
Vln. II
Vla.
Vc.
mf
f
mp
p
522
espressivo
Bring out
3
527
broadly, growing
ord.
2
4

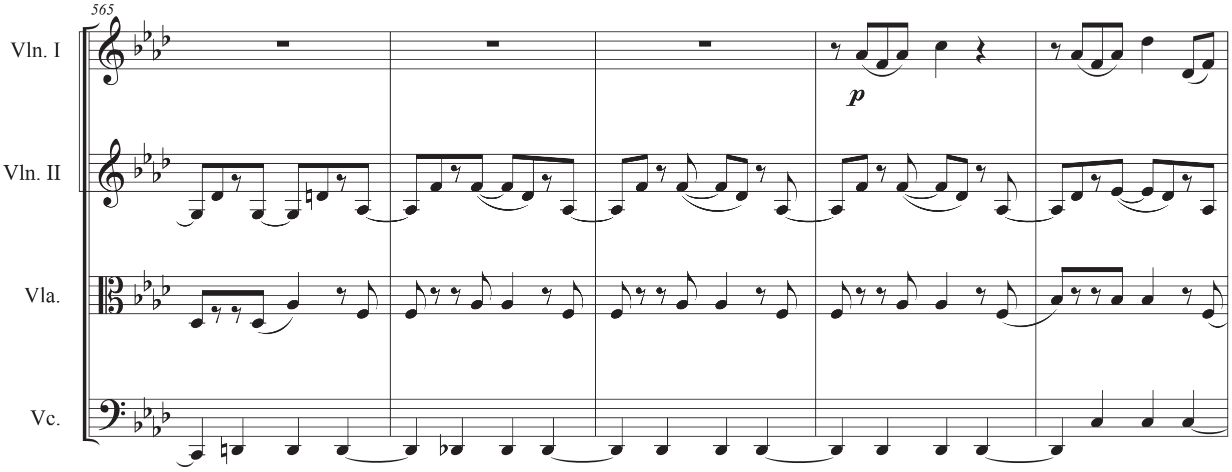

533
Vln. I
Vln. II
Vla.
Vc.
mp
mp
mp
pp

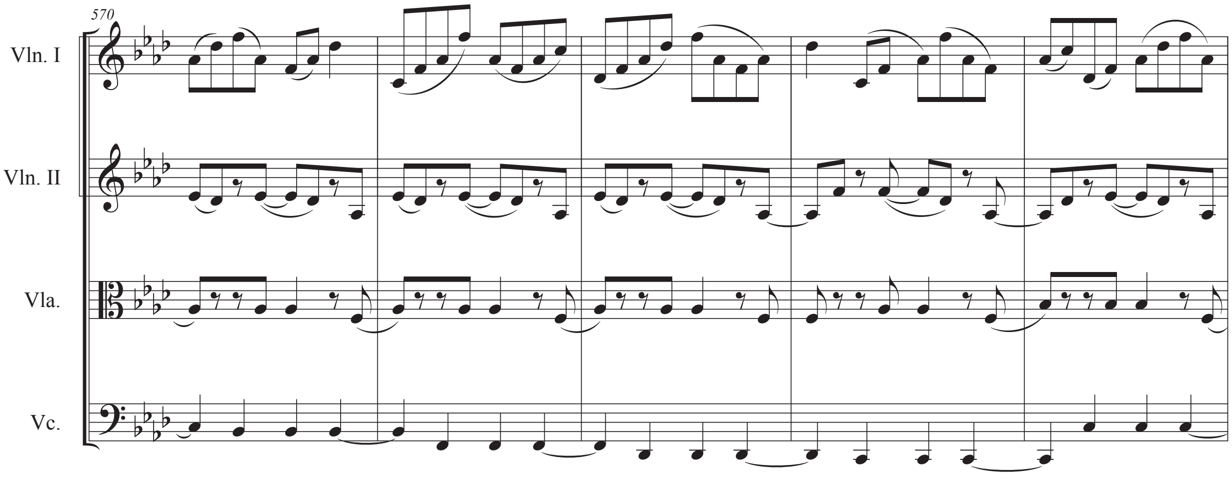

538
Vln. I
Vln. II
Vla.
Vc.
3
3
3
mp
p
mp

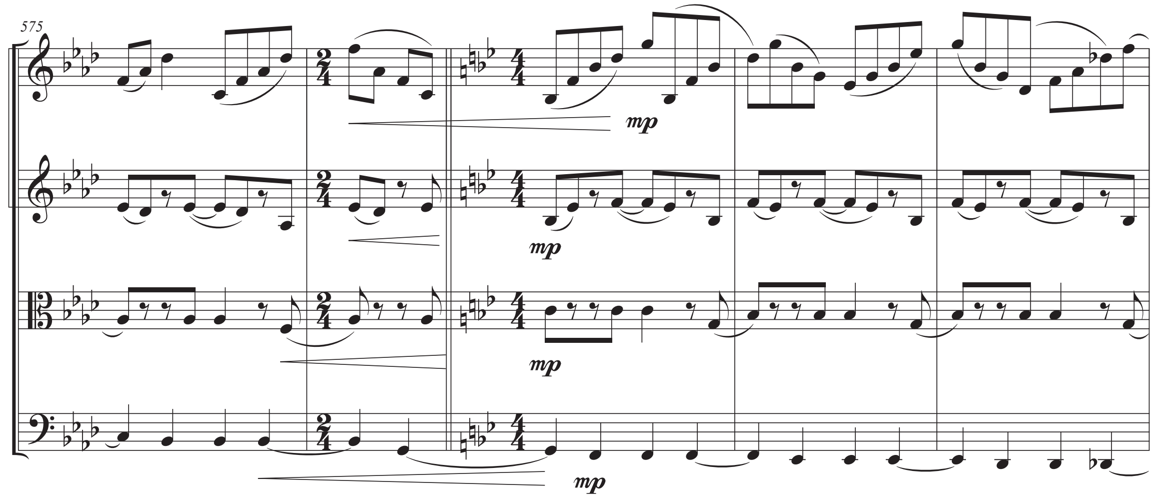

542
Vln. I
Vln. II
Vla.
Vc.

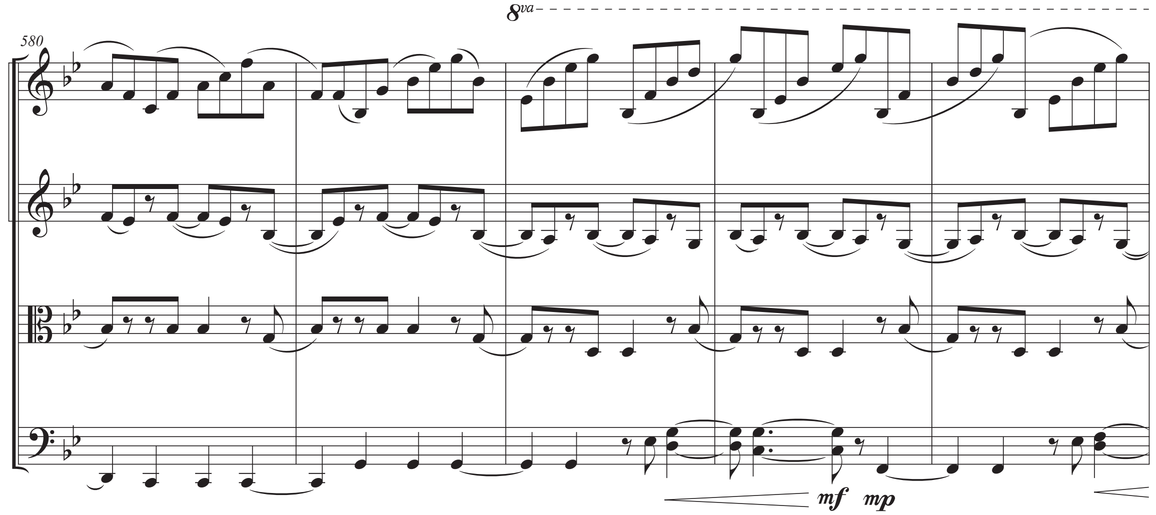

L
547
Vln. I
Vln. II
Vla.
Vc.
mf
f
p sub.
p
551
555

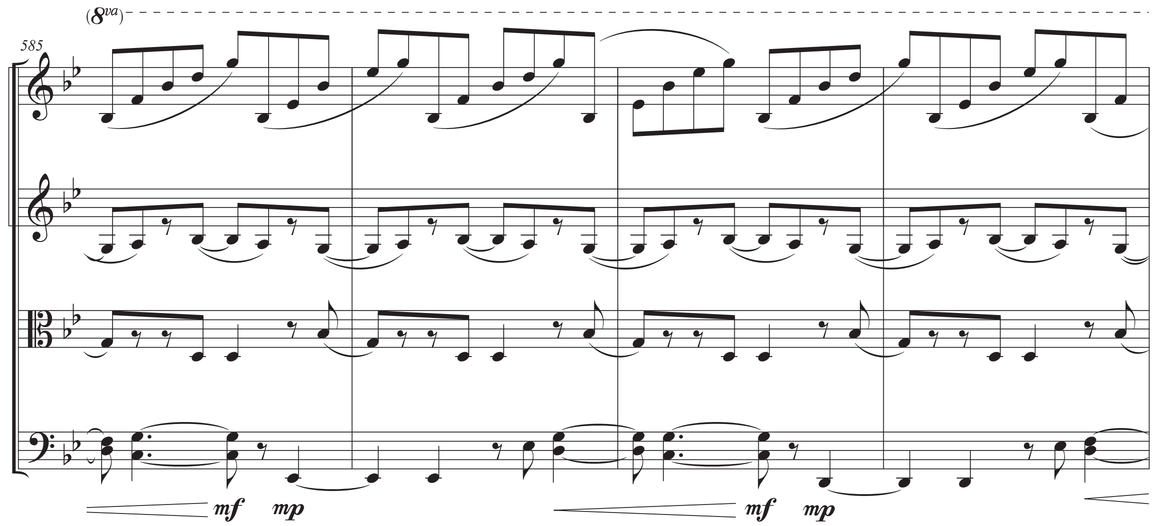

560
Vln. I
Vln. II
Vla.
Vc.

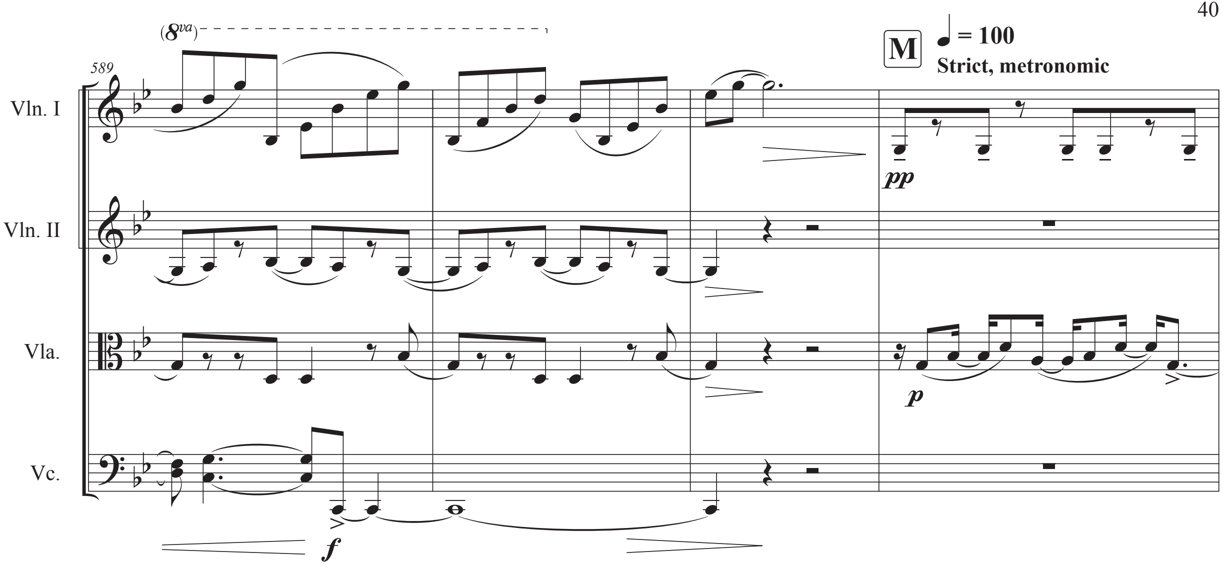

565
Vln. I
Vln. II
Vla.
Vc.
p

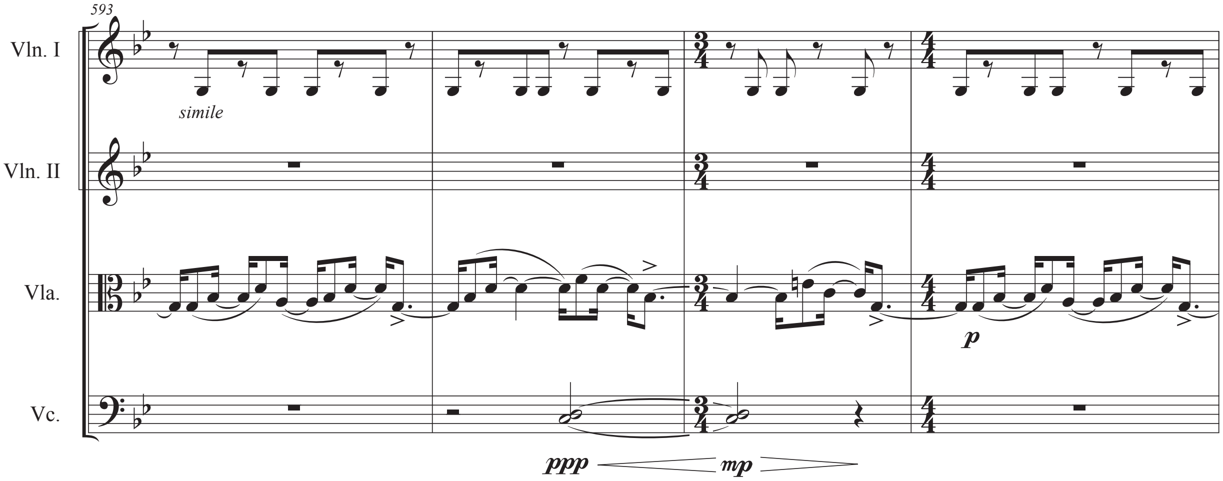

570
Vln. I
Vln. II
Vla.
Vc.

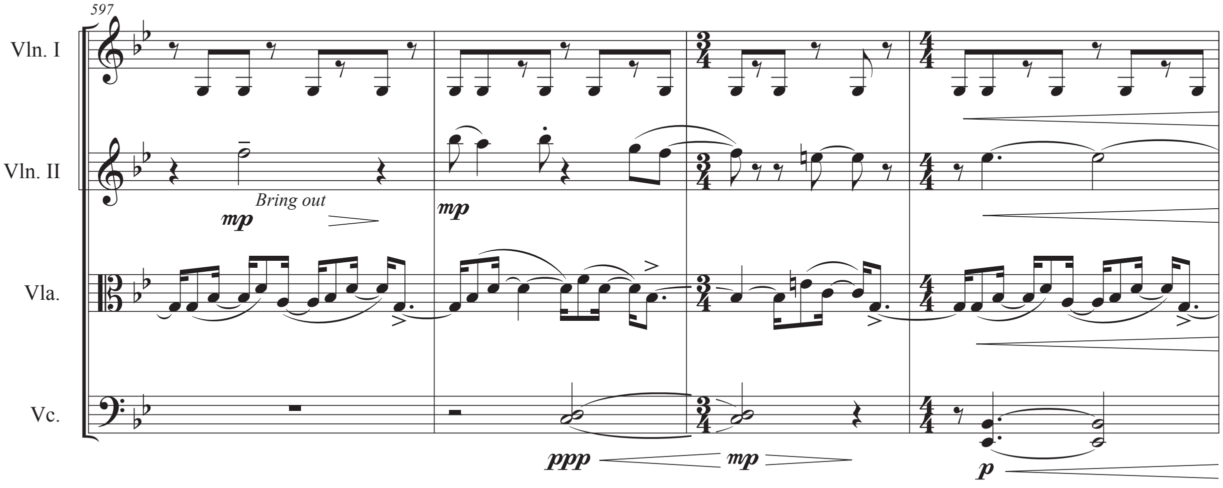

575
Vln. I
Vln. II
Vla.
Vc.
mp
mp
mp
mp

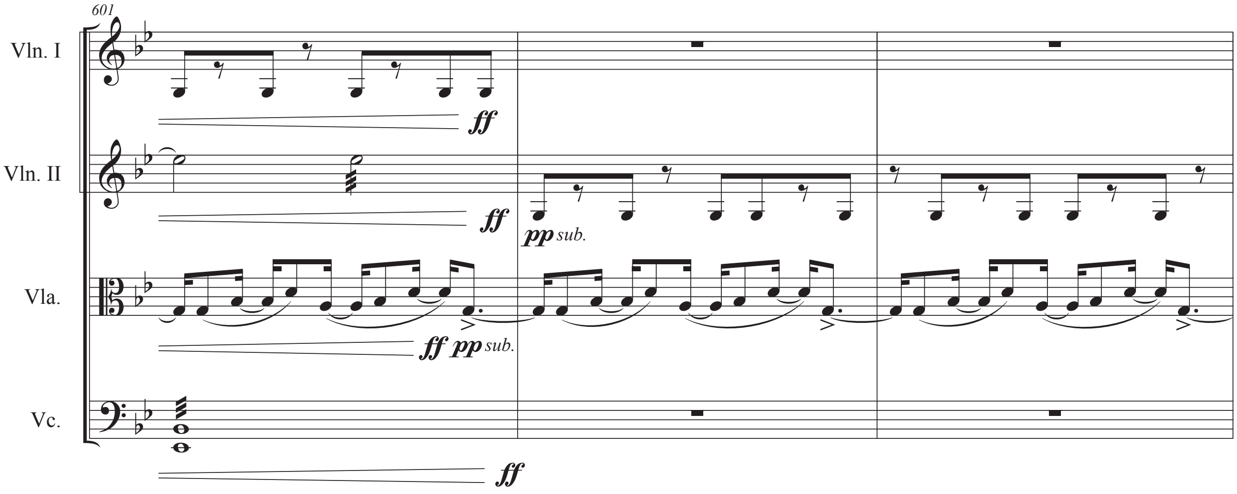

580
8va
Vln. I
Vln. II
Vla.
Vc.
mf
mp

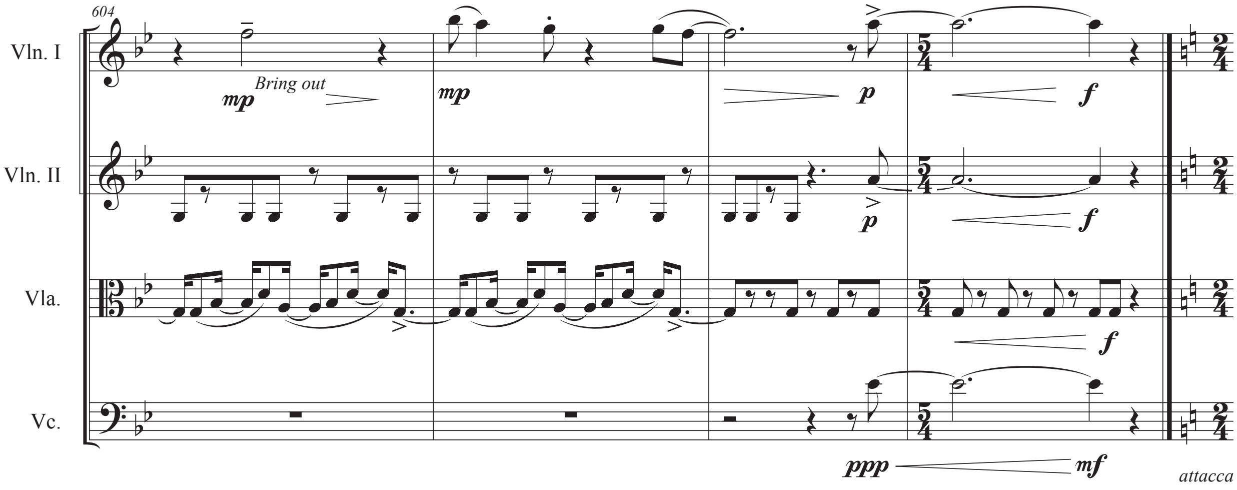

(8va)
585
Vln. I
Vln. II
Vla.
Vc.
mf
mp
mf
mp


589
(8va)
M
♩ = 100
Strict, metronomic
Vln. I
Vln. II
Vla.
Vc.
pp
p
f

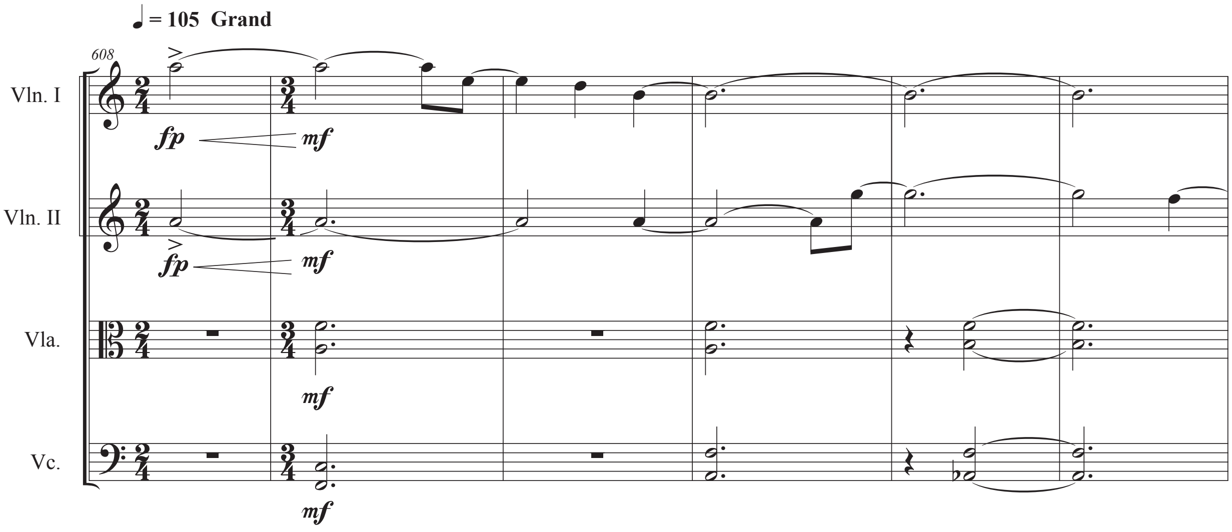

593
Vln. I
Vln. II
Vla.
Vc.
simile
p
ppp
mp

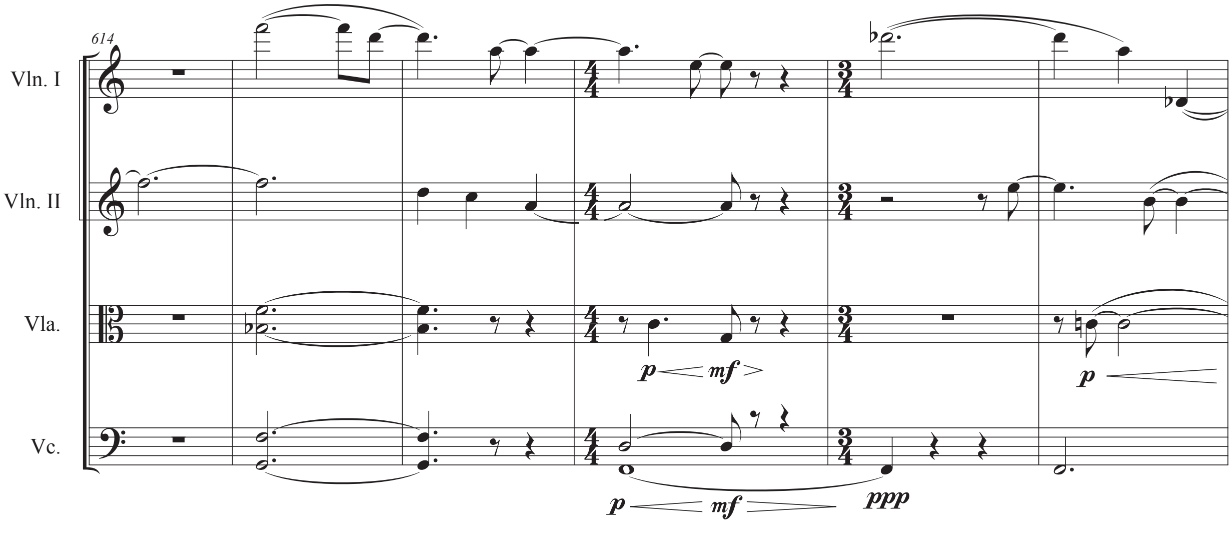

597
Vln. I
Vln. II
Vla.
Vc.
Bring out
mp
mp
ppp
mp
p


601
Vln. I
Vln. II
Vla.
Vc.
ff
ff
pp sub.
ff pp sub.
ff

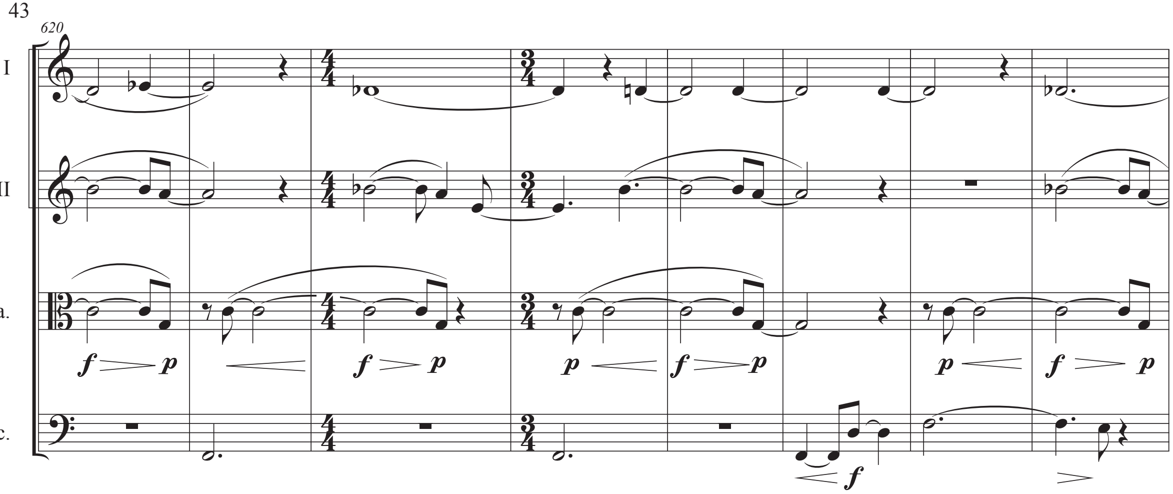

604
Vln. I
Vln. II
Vla.
Vc.
mp
Bring out
mp
p
f
p
f
f
ppp
mf
attacca

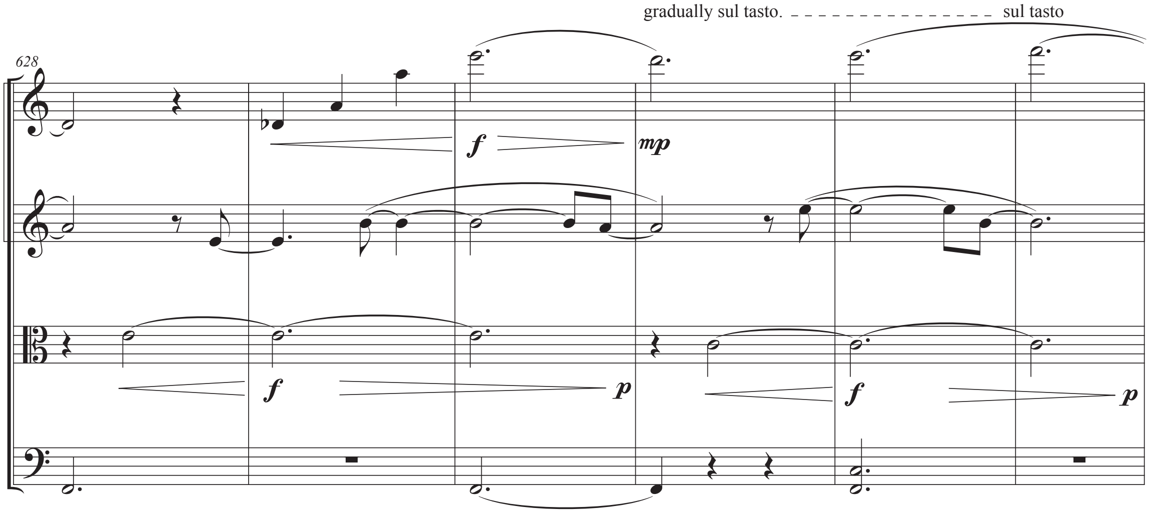
VI
Generation

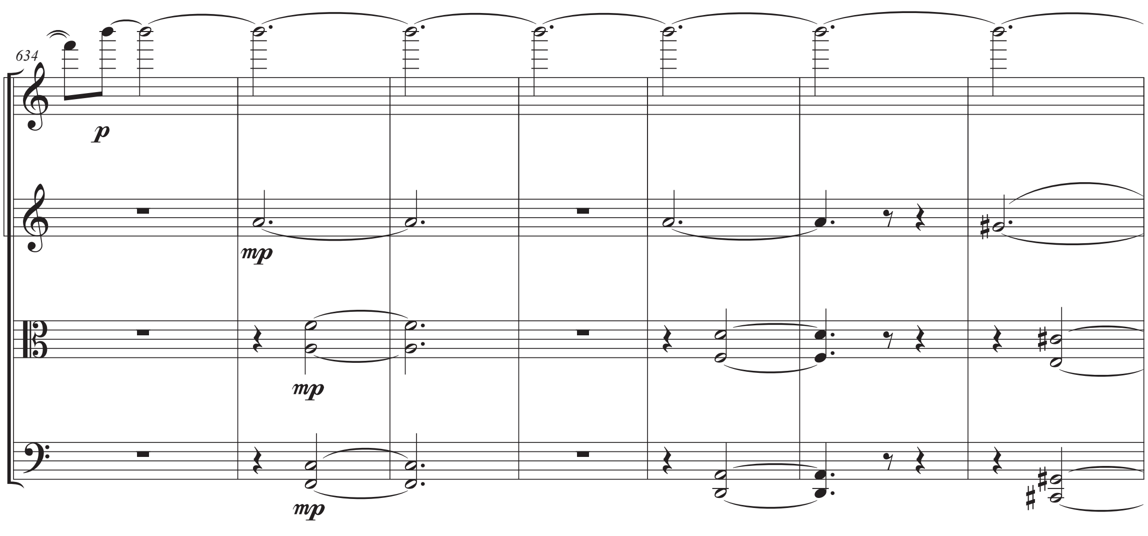


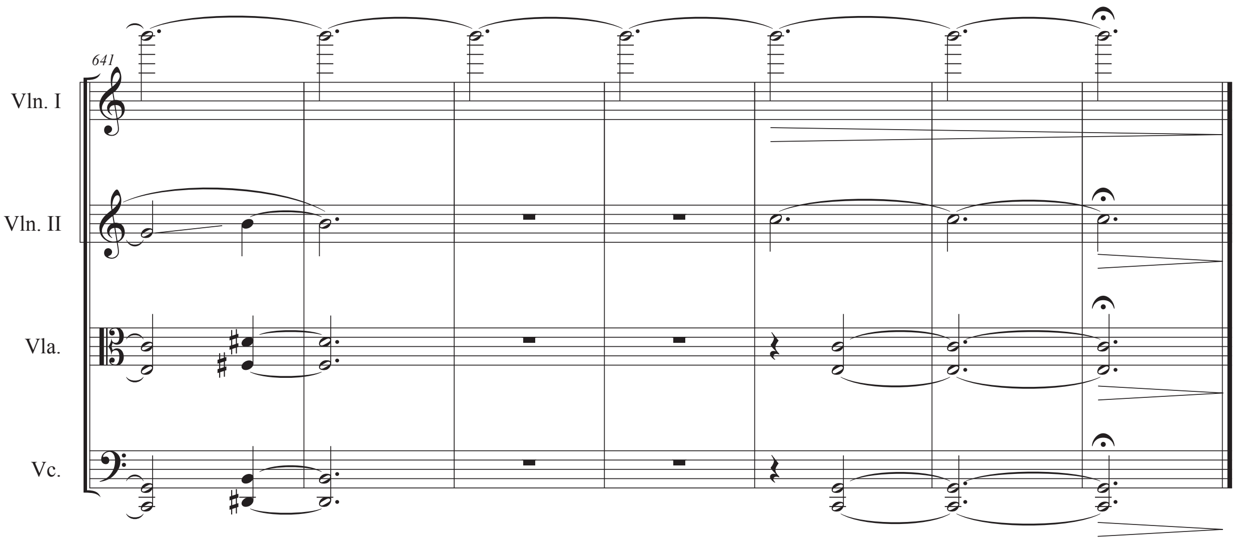


MUSIC FOR SELF-ATTENTION








641
Vln. I
Vln. II
Vla.
Vc.

3.2 Compositional Process & Analysis

3.2.1 Musical Prompt & Overall Form

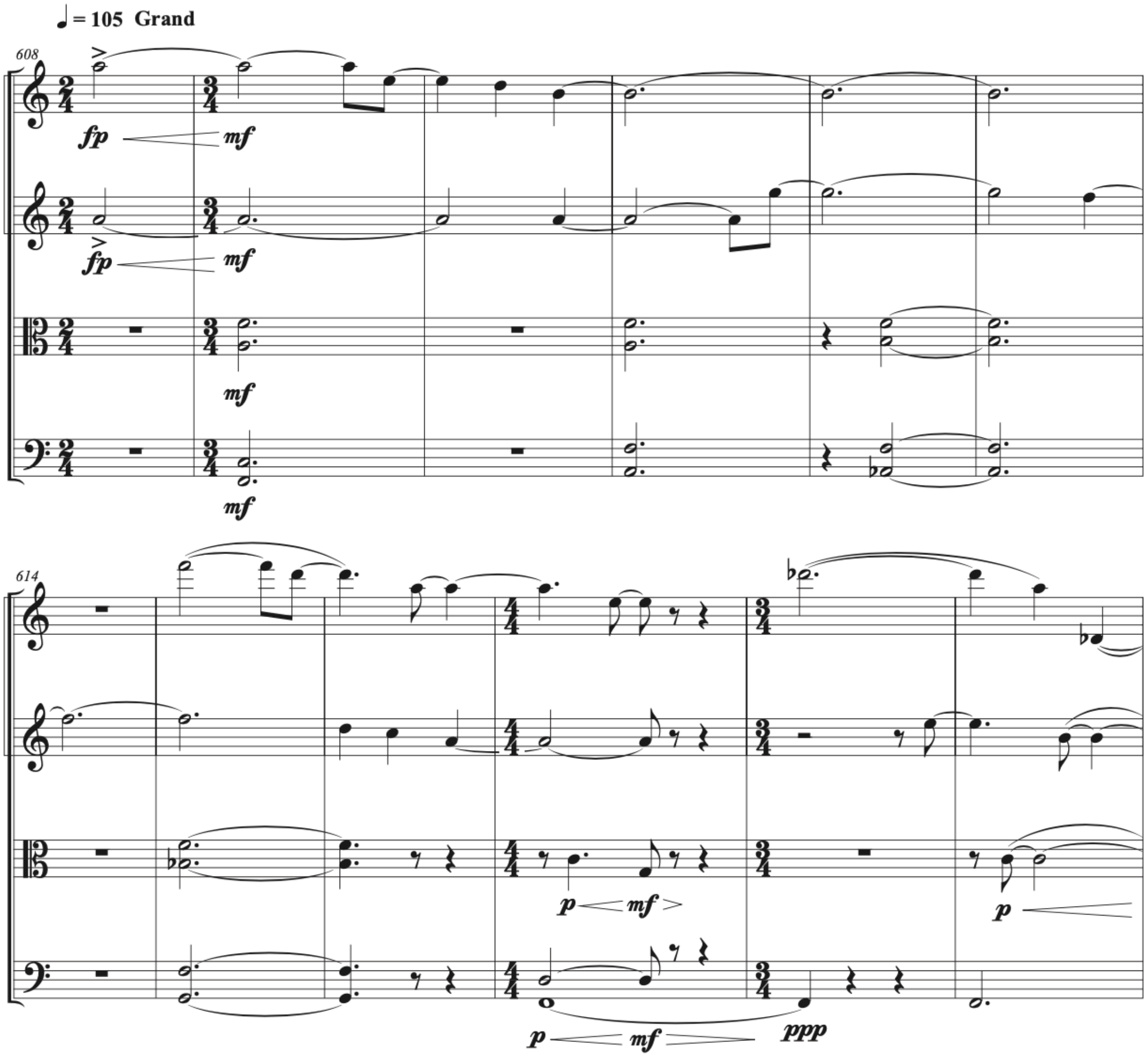
Each of the first five movements of *Music for Self-Attention* begin with music generated by Score-Transformer (ST). In order to give ST some context for that artificially generated output, I supplied ST with a musical prompt taken from the first ten measures of the final movement, "VI – Generation" (Ex. 1)[71]. *VI – Generation* is the only movement to feature no material generated by ST and was the first movement composed for the piece. To prevent these passages from sounding too similar, the sampling parameters were altered for each movement. Furthermore, these alterations were determined specifically to follow an overall form that can be thought of as a loose theme and variations form in reverse. Typically, a theme and variations form features some opening passage of music which is manipulated to become more elaborate as the piece progresses. Within my piece, the musical prompt is considered to be the 'theme' and the 'variations' are ST's artificial continuation of the theme guided by different sampling parameters. This is not a theme and variations in the usual way as the listener won't hear the theme in each variation. Therefore, the reversing of a theme and variations in my piece features music which becomes less elaborate over time by manipulating ST's sampling parameters. Beginning with a movement (variation) that is most unlike the final movement (theme) in terms of pitch collection and rhythmic patterns, as the piece progresses each movement becomes more stylistically akin to the final. To determine sampling parameters for each movement's initial generated content that would demonstrate this overall form, a hyperparameter search (see section 2.2.6) was conducted. A hyperparameter search allowed me to test many combinations of the sampling parameters and their effect on the output (see Appendix B for examples from the hyperparameter search). This search revealed that a *temperature* of 0.8 and a *top p* of 1.0 generated consistent musical output related to the prompt in terms of pitch and

[71] Each example showing every movement's opening passage generated Score-Transformer will have the prompt found in Example 1 removed for redundancy. However, the reader should be aware that this prompt had appeared immediately before each generated excerpt.

rhythmic content. Therefore, each movement had opening musical material generated for it using only the *temperature* and *top p* parameters. The values of 0.8 and 1.0 for *temperature* and *top p* respectively are never used as the intention was to show the parameters getting ever closer to these values (see Appendix C for table of parameters used per movement) and the final movement which would use these parameters instead features no artificial musical material.

This overall form is inspired by the work *Douze Tiroirs de Demi-Vérités Pour Alléger Votre Descente* (1981-82) by Denys Bouliane. In his work, a long unfolding process removes dissonance or "distortion" in his own words (Bouliane, 2018), to reveal a passage from a Bach chorale. This initial passage was then manipulated for use in the earlier sections by adding or subtracting interval amounts by which voices of the chorale were transposed. For example, if a transposition of 0 equals the original Bach theme, a transposition of 1 has the soprano line transpose up a semitone and the bass in the opposite direction down a semitone. The farther away from the original Bach passage (or the closer to the beginning of the movement), the more distortion exists owing to the higher interval used for this displacement. This movement of Bouliane's work simply involves the reduction of distortion over time until the final passage involving an exact representation of the original Bach chorale is found.

In changing the concept of reducing "distortion" over time in my own piece, I define distortion as chromatic and rhythmic displacement from the final movement of the piece. Thus, the reduction of distortion over time in terms of my piece features a gradual shift from the inclusion of chromaticism, non-functional voice leading and varied rhythmic values towards elements more stylistically appropriate to the final movement. While a theme and variations in reverse constitutes an overall form to the piece, each movement is further designed to address and musically represent different processes involved in the training and practical use of ST. Each movement also features these processes in the same order in which they occurred in the program. Thus, ST is theoretically and practically involved in the design and development of this piece. While this document is intended to be read in the order of movements listed, it can be helpful to read "VI – Generation" (section 3.2.7) first which contains an analysis of the prompt used by ST to generate initial musical material for each of the first five movements. This can perhaps provide details on the departure point from which each other movement was developed.



Example 1: Opening measures for "VI – Generation".

3.2.2 I – WTE + WPE

The first four movements of *Music for Self-Attention* incorporate processes found within the transformer-decoder model's architecture (Fig. 8). The first movement delves into positional encoding, an initial process where the model looks up a vector from an input's token embedding (WTE) matrix and adds a vector from the positional embedding (WPE) matrix[72,73]. In short, the meaning of a token (note event, velocity or duration) is being combined with its position in the input batch. Without the addition of a positional embedding, the model could not distinguish between two of the same tokens when used in different contexts. For example, an 'F♯' used as a leading tone will often precede a 'G' which is a characteristic we would want the model to learn. By adding positional embeddings to each of these pitches, the model can learn that this order of notes is quite common in tonal music.

The token embedding matrix (see Fig. 14) contains rows which represent entries of the dictionary set (357 note, velocity and durational entries; section 1.2.1) where each row is an *n*-dimensional dense vector containing rational numbers[74,75]. The size of *n* is called the 'embedding size' in a model and in the case of ST, the value is 384[76]. The positional embedding matrix (see Fig. 14) contains rows for each token in a model's input (for ST, 1024 tokens are input into the model with each training step) where each row is similarly an *n*-dimensional vector (where *n* must equal the embedding size) containing rational numbers between -1 and 1. The creators of the original transformer model used sine and cosine functions of different frequencies to encode these token positions[77,78], thus, why the positional vectors contain rational numbers between -1 and 1. Half of the positional vector is generated using a sine function and the other half using

[72] The 'W' in 'WTE' or 'WPE' refers to 'word.' I retain the acronym found in most literature on transformer models even though it has no meaning within a musical domain.

[73] Géron, p.558.

[74] Chollet, p.184.

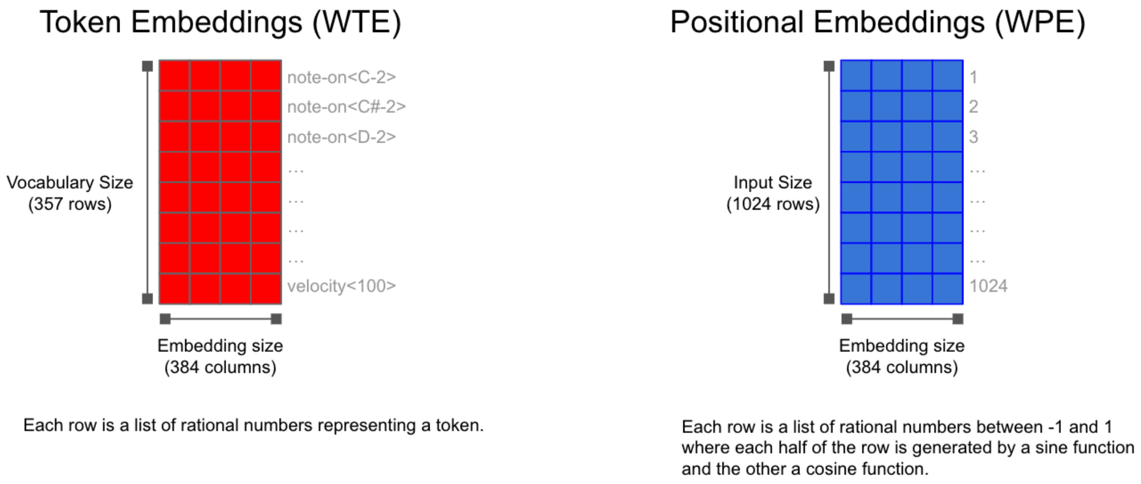
[75] The rational numbers are decided by the model during an initial embedding process in order for it to distinguish one vector from another.

[76] In my model, embedding size is calculated by multiplying the number of attention heads by the number of decoder stack layers (see 2.2.2 & 3.2.3).

[77] Vaswani et al., 2017.

[78] Géron, p.557.

a cosine function. These embedding matrices are fixed and will never change throughout the course of training. To add positional embedding vector to each token embedding vector found within in an input batch (1024 tokens in a batch), the model goes one by one through the batch and first looks up the row (vector) associated with the token in the token embedding matrix, then adds that vector to a vector taken from the positional embedding matrix based on its position in the input batch (see Fig. 15).


Token Embeddings (WTE)
Vocabulary Size (357 rows)
note-on<C-2>
note-on<C#-2>
note-on<D-2>
…
…
…
…
velocity<100>
Embedding size (384 columns)
Positional Embeddings (WPE)
Input Size (1024 rows)
1
2
3
…
…
…
…
1024
Embedding size (384 columns)

Each row is a list of rational numbers representing a token.

Each row is a list of rational numbers between -1 and 1 where each half of the row is generated by a sine function and the other a cosine function.

Figure 14: Token and positional embedding matrices.

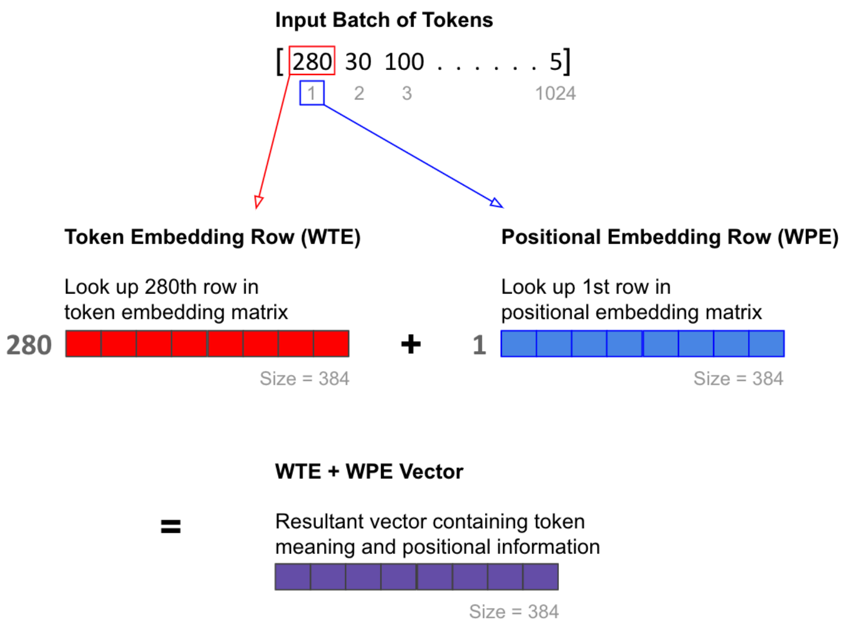
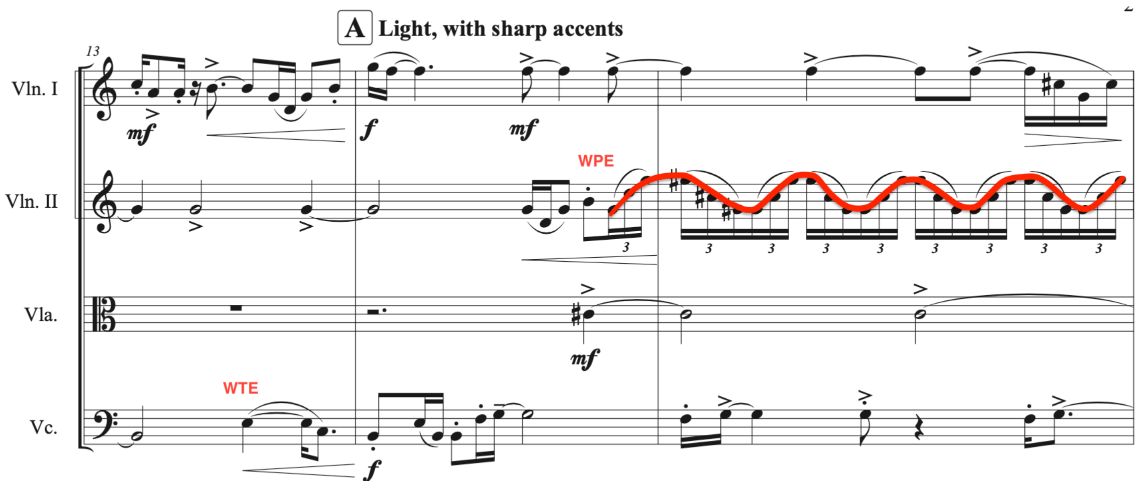


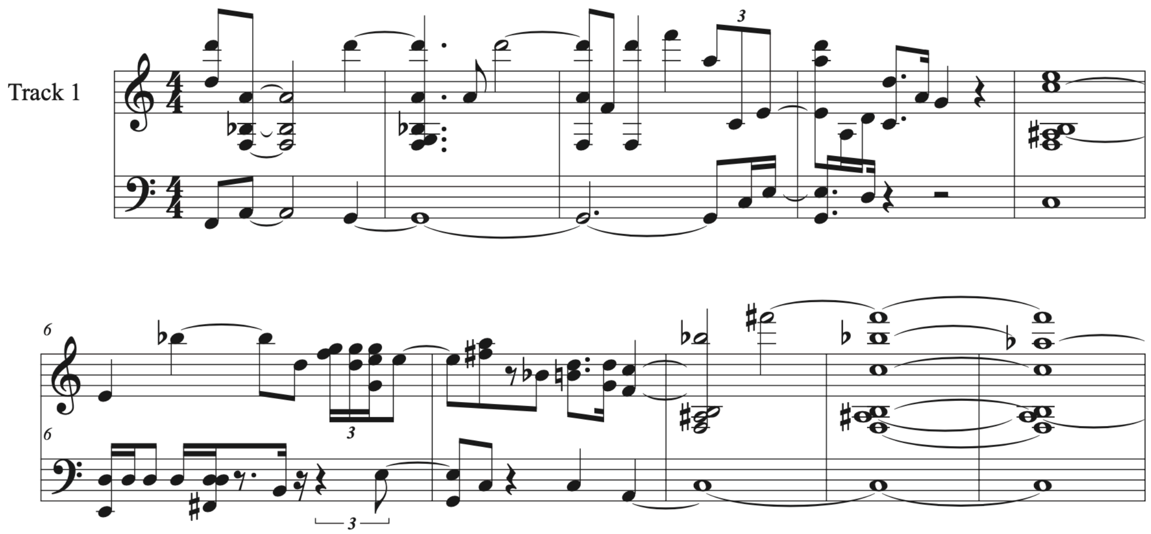
Figure 15: Visualization of a token embedding vector added to a positional embedding vector.

To imitate this process musically, this movement features moments of sweeping lines that run up and down imitating the use of sine and cosine functions for positional embeddings (Ex. 2). These musical representations of positional embeddings are simultaneously coupled with the musical passages representing token embeddings. Token embeddings found throughout the A section of the movement are represented by melodies, while the B section reimagines them as vertical sonorities. A short coda section follows the B section before the conclusion of the movement. Example 2 shows where the token embedding is found in the cello as a melody while the positional embedding occurs in the second violin as it quickly arpeggiates up and down using a ‘C♯’ and ‘G♯.’ This arpeggiated gesture is then passed to the first violin which arpeggiates a chord built on the pitches ‘C♯’, ‘F’ and ‘G’. The viola then appears with a similar up and down gesture between the pitches ‘C’ and ‘F’. This polyphonic texture passes around the sine and cosine representation until measure 24, where the viola and cello alone continue the idea of addition between WTE and WPE. Token and positional embeddings are always found to be occurring at the same time throughout the movement to reflect the addition of the two vectors that occurs within the model before being passed to the next stage.



Example 2: A demonstration of WTE + WPE (mm. 13-15).

The initial musical material for the movement was generated by ST (using the prompt from the final movement) with the sampling parameters of a *temperature* of 2.5 and a *top p* of 1.5 (Ex. 3). This constitutes the highest *temperature* and *top p* to be used throughout the piece as each subsequent movement will gradually reduce the value of these parameters. From this sampling parameter combination, the most varied pitch and rhythmic material (in comparison to the prompt) occurs within the piece. Since these parameters created poorly voiced chords, events seemingly out of nowhere, ties going nowhere and other strange events, I made selections to create the resulting opening measures as seen in the score. The first four measures were left intact to some degree with the repeating high D gesture (related to the open voiced 'F' chord in the prompt) and the rhythmic values, which I deemed to be a good opening for the piece. This relation between this opening passage and the prompt gave the piece "book-end" musical gestures. Instead of copying the version generated by ST exactly, I repeated the rhythmic material in mm. 1-4 but transposed the gesture to a much lower register in the violin 2, viola and cello. Afterwards, I include the music found in mm. 6-7 of ST's generated excerpt as the quick harmony passage between the two violins found in mm. 10-11 of the score. This is a technique I used throughout the piece, not literally setting all of the generated passages by ST into my own score but rather by selecting the pieces of it that inspired me or fit the content I wished to further develop.



Example 3: Initial ST generated excerpt for the opening of the first movement.

Some quirky rhythms appear as a result of these sampling parameters that will begin to disappear with each subsequent movement. I have tried to leave unusual patterns where possible to demonstrate the behaviour of different sampling parameters over time and their relation to the final movement. After the initial musical passage was generated (Ex. 3), I continued to use ST in the generation of new passages. However, I did not feel compelled to use the same sampling parameters of the opening excerpt as this was to generate initial context for how the movement would continue. In the artificial generation of new material during the movement, I used sampling parameters more closely related to those determined by my initial hyperparameter search which were found to create content related to the prompt. Thus, throughout the movement I tended to use sampling parameters closer to a *temperature* of 0.8 and a *top p* of 1.0. This was to keep the newly generated content associated with the style of the movement by relating to the opening passage. Some examples of ST generated content throughout this movement are the cello melody in mm. 24-29 (using the whole quartet in mm. 20-23 as the prompt), the viola line in the same measures (using the cello line as a prompt with a low *temperature* to looping patterns) and the high melody in the violin 1 part at mm. 47-56.

The last example of the high violin 1 melody was generated using the violin 1's melodic moment in mm. 43-45. This demonstrates another useful aspect of ST where the *temperature* is reduced to a value closer to 0. If we determine a *temperature* of 0.8 to be the best value for generating content, lowering this value can trick the model into getting stuck in a loop. In terms of modelling natural language, this is often considered to be a negative result as the repetition of the same few words rarely generates a very compelling sentence. However, in a musical domain, this can create ostinati or looping patterns which can be desirable output for a particular section of a piece a composer is working on. As expected, I found that if I raised *top p* or *top k* values with a low *temperature*, I could still keep the loop but affect its pitches or durational values. Through experimenting with this, I managed to create the very quirky violin 1 part that makes up the majority of the B section (beginning at rehearsal marking 'B'). As I wanted to again demonstrate the addition of token and positional embeddings within the B section, I decided to make this melody in the violin 1(mm. 47-58) represent positional embeddings and therefore manipulated the quasi-ostinato pattern to run down and up again a few times before its completion. Each time the melody runs downwards, it only passes through pitches which have occurred so far in the line. The vertical sonorities in the remaining instruments are there to imitate token embeddings that are being paired up with positional embeddings created from sine and cosine functions.

3.2.3 II – Multi-Head Self-Attention

Perhaps the most important aspect of the transformer-decoder model is the attention mechanism (Fig. 8)[79]. The specific type of attention mechanism used within most transformer-decoder models, including my own, is the self-attention mechanism[80]. Given some token in a sequence, the self-attention mechanism is used within the decoder stack to "pay attention" to all tokens which include and precede that particular token. Thus, the name self-attention refers to the fact that the mechanism attends to itself in order to encode all of the relationships between the tokens it has seen. The mechanism will also

[79] Bahdanau et al., 2014.

[80] Géron, p.549-552.

pay more attention to tokens found to be more significant[81]. For example, when used in a natural language context, if the model is supplied with the following tokens “They welcomed the Prime Minister of Canada,” the attention mechanism will more likely focus on the tokens “Prime,” “Minister” or “Canada” rather than “They” or “welcomed.” As the model is autoregressive or works sequentially, self-attention mechanisms only look backwards and never forwards by masking all future tokens[82,83]. While part of the reason for using self-attention is due to minimizing computational expense over past methods, authors Vaswani et al. also show improvements in the ability for the model to learn longer-range sequences[84]. For a music-based model, this is an imperative feature to generate sequences longer than a few bars or containing many tracks or staves.

Two other important aspects involved in this stage of the model are multi-head attention and concatenation. Multi-head attention involves the same process as mentioned above, however, the self-attention process is repeated many times in order to extract extra information about the sequence[85]. Let’s again use the example “They welcomed the Prime Minister of Canada.” The self-attention mechanism may have caught and encoded that “welcomed” is a verb. However, on another run through the mechanism, it might encode that the token “welcomed” is also a past-tense verb. Increasing the amount of ‘heads’ or how many times the self-attention mechanism process is repeated is more likely to capture all of the possible characteristics than if too few attention heads (or a single head) were used. In the final stage, all of these encoded characteristics are

[81] Ibid, p.556.

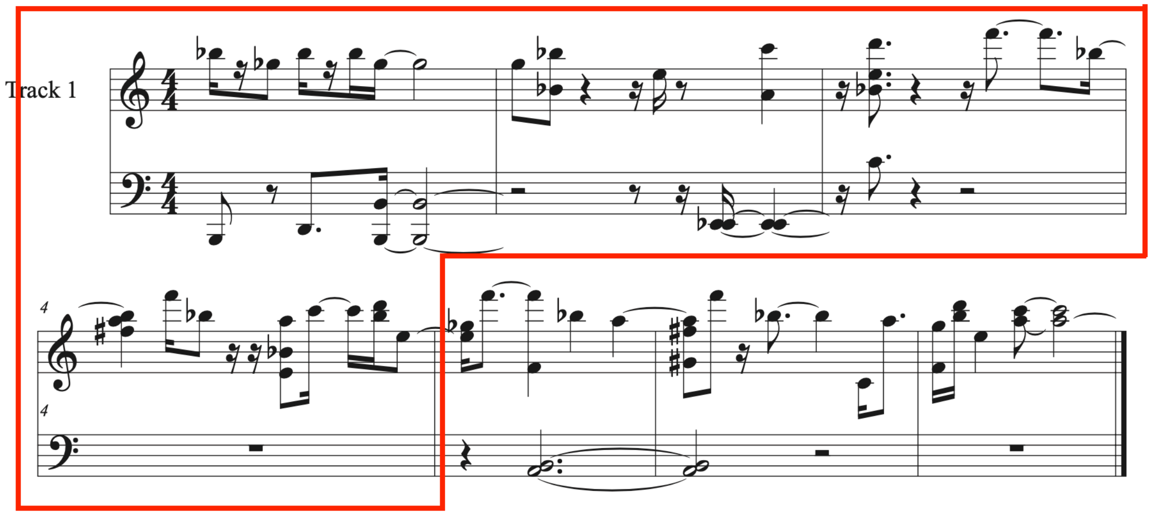
[82] Hence why the self-attention mechanism is also referred to as masked self-attention mechanisms.

[83] Vaswani et al., p.3.

[84] Ibid.

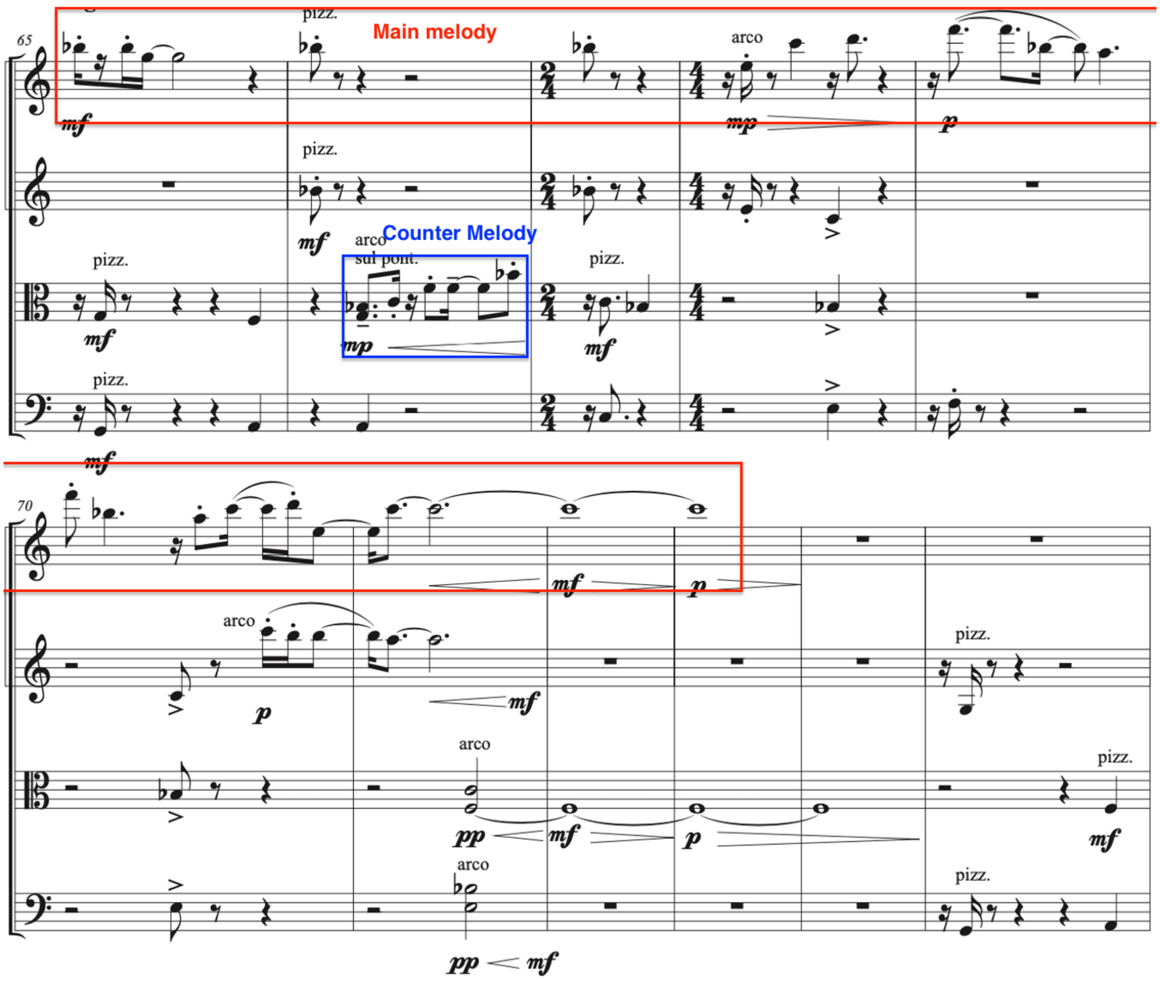
[85] Géron, p.562.

concatenated into a single vector which continues along towards the next stage of the model.



Example 4: Initial material generated by Score-Transformer with sampling parameters: *temperature*=2.0, *top p*=1.4.

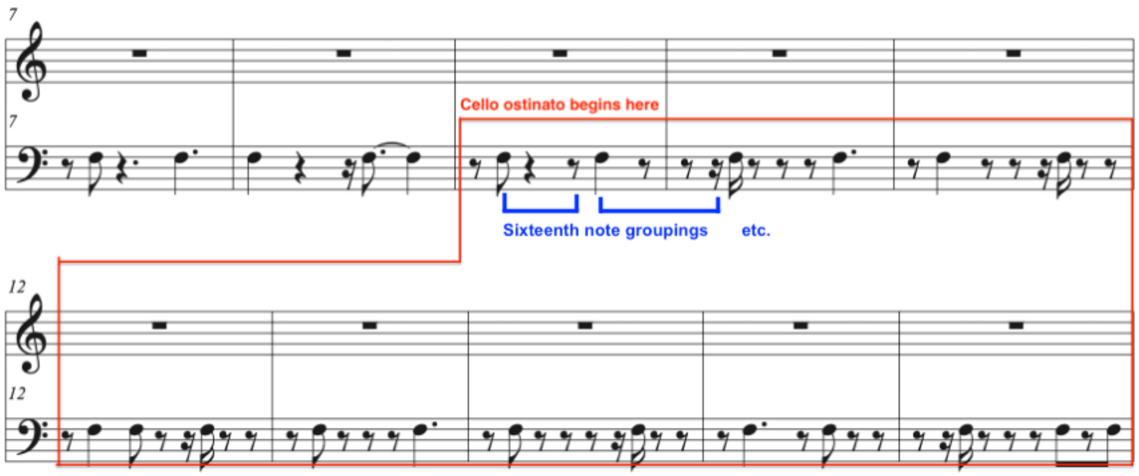
"II – Multi-Head Self-Attention" involves two contrasting sections (referred to here as the A and B section) with the latter featuring a musical representation of the multi-head self-attention mechanism. The short introductory section includes initial material generated by Score-Transformer (Ex. 4) with the sampling parameters of a *temperature* of 2.0 and a *top p* of 1.4. While I cleaned up this passage by removing the left-hand material, altering the 'G♭' to a 'G' natural and orchestrating it for string quartet, the rhythmic content is largely left untouched (Ex. 5). Here I consider the violin I part in mm. 65-73 (Ex. 5) as the main melody and I added a brief counter melody (m. 66) in the viola which is later repeated and extended throughout mm. 76-79. In treating the accompaniment from ST, I found that the sporadic harmonies found in the artificially generated passage would be well articulated as pizzicato and staccato pitches. These light, more sparse textures of the movement are a reprieve from the denser textures found within the opening movement and the even thicker textures yet to come in the next movement.



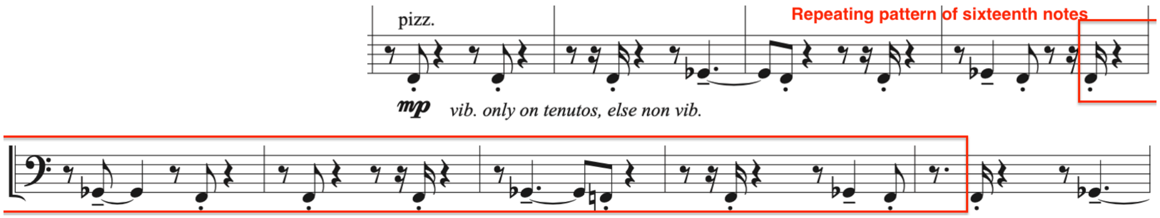
Example 5: Opening of "II - Multi-Head Attention" using the artificially generated material from ST.

One important aspect to note within this section is the reoccurring dominant ninth chord with unresolved voice leading. This C^9 chord first occurs in mm. 68 on the third beat and is followed by an octave 'F.' The treatment of the bass voice which moves from 'E' to 'F' makes sense but the 'B♭' and 'C' drop out. Furthermore, the chord reappears three more times within the section and is never resolved. Common practice conventions, prior to the 20th century, would have us expect some type of voice leading treatment should occur but here the artificially intelligent software ignores it and treats it as a standalone chord. Most likely, this has to do with the sampling parameters involving a high *temperature*, so the model is making selections outside of what it really learned during training. As will be seen later with the lowering of the *temperature* parameter, cadences and dominant chords are more likely to be treated according to common practices of the classical period. Since this piece is demonstrating varied sampling parameters over time, it is important to take note of these earlier moments where the model offers alternatives to the functional harmony found later in the piece.

The following B section beginning at rehearsal C, starts with a cello ostinato taken from an excerpt of an artificially generated excerpt (Ex. 6). As mentioned within the earlier section, changing the *temperature* to a value closer to 0 can cause the model to get stuck in a loop. While this is seen as a problem in natural language, it can be a desirable effect used in a musical domain where repeated gestures are more common. In the case of this movement, I was attempting to elicit the feeling of a model sequentially working through each token within its input. The five measures (including all 4 staves) preceding rehearsal C were used as the prompt this time with the sampling parameters of *temperature*=0.6 and *top k*=50. In using *top k* instead of *top p* (as was used throughout most of the piece), I further force the model to limit the available tokens to pick from in the hopes of generating a single line ostinato. The rhythms here are a bit unorthodox as I left the model's rhythmic decisions untouched. This is a continuation of methods seen in the previous movement which occur from the manipulation of the *temperature* parameter, allowing the model to make decisions less likely to occur with a lower *temperature*. Here, the quirky resultant rhythm includes irregular groupings where individual attacks are separated in time by durations measured in varying multiples of 16th notes (7, 8, 9 sixteenth notes per group) which appear in no particular pattern until reaching two groupings of 4 and 5 sixteenth notes respectively. After which, a pattern of irregular sixteenth-note groups (7, 8, 8, 9, 7, 8, 9, 7, 4, 5) repeats until the quartet unison passage at rehearsal 'D.' A few modifications to the artificially generated excerpt include transposing it down an octave and the addition of a new pitch ('G♭'). A performance note was included to bring out each 'G♭' instance by sustaining the pitch and by adding vibrato. Before arriving on the theme of this section in m. 109, there is a false start in the violin at mm. 93 and further some textural material in the violins and viola. Both of these are foreshadowing what is to come for the duration of this section.

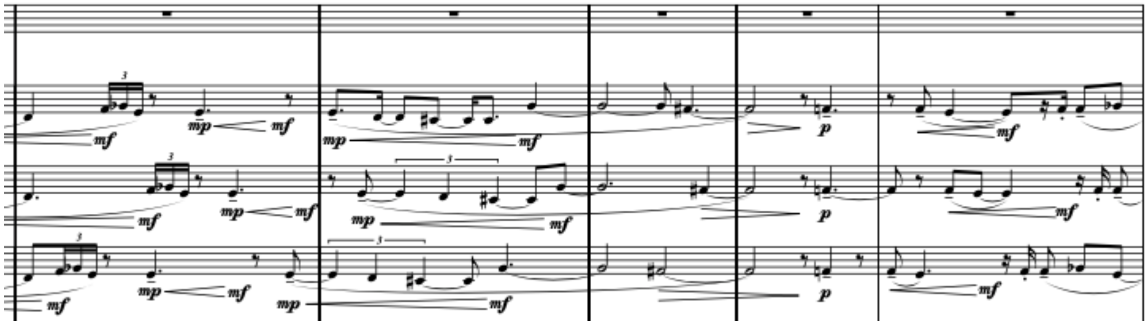


Example 6: ST generated material for the cello ostinato.


pizz.
mp
vib. only on tenutos, else non vib.
Repeating pattern of sixteenth notes

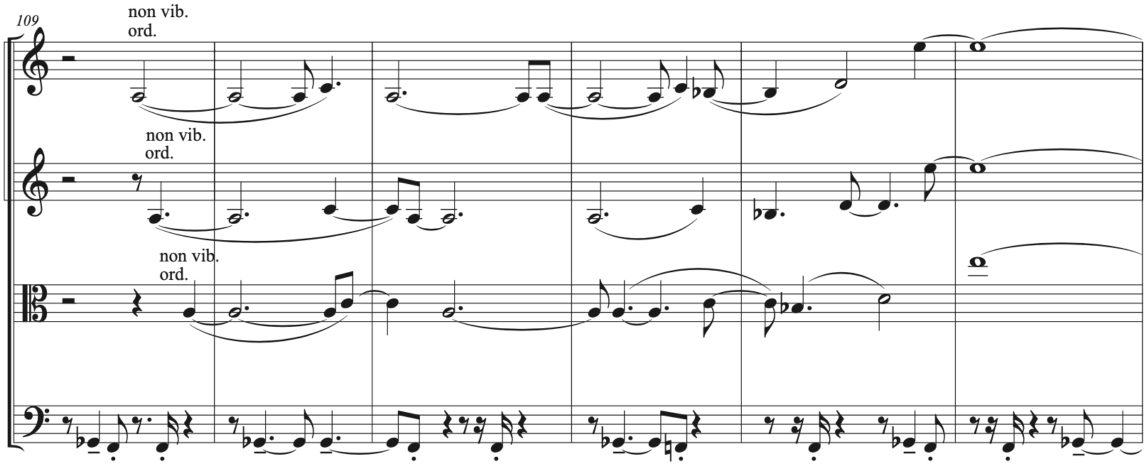
Example 7: Opening measures of the cello ostinato (mm. 88-96).

In m. 109, an initial musical passage is copied and set one eighth note apart with each new entry in a different instrument (the violins and viola). This is a technique I utilized in my master's thesis piece *Two Movements for Orchestra* which observed and altered the phasing techniques of Steve Reich. In *Two Movements for Orchestra*, I was attempting to demonstrate Reich's phasing technique used with longer and more complex melodies as opposed to his shorter and simpler loop patterns (Ex. 8). In borrowing this technique from my old work, I wanted to musically represent the multi-head attention process where some input is sent through the attention mechanisms multiple times in order to gain different perspectives of its characteristics. By having a melody repeated amongst the violins and viola and offset by an eighth each time, it's as though the listener is peering into a stack of attention mechanisms working at the same time. If I had set this material as a single melody at unison, it would harder to convey this premise to the listener. This continues throughout the section and slowly builds a longer melody piece by piece, (each of these pieces begin at m. 109, m. 118 and m. 123 respectively). Some artistic liberties were taken in order to show how multi-head attention might learn different characteristics by having these chunks of the melody add one extra note with each out-of-phase repeat. For example, in measures 109-117, the repeated melody in the viola has two more pitches (‘E♭’, ‘B♭’; mm. 115-116) than the violin 1 and similarly, the violin 2 part contains one extra pitch (‘E♭’; m. 115).



Example 8: Excerpt from *Two Movements for Orchestra* (2016)[86] demonstrating out of phase melody setting amongst different instruments.

[86] Jeffrey A. T. Lupker. *Two Movements for Orchestra*. ()



Example 9: Mm. 109-114 from the second movement demonstrating the phasing technique.

Finally, at rehearsal D, all three chunks of the building melody are put together to represent the next stage of the process which is concatenation. As we have moved beyond the multi-head attention stage, the out-of-phase process has ceased and instead all four instruments of the quartet play the entire melody in unison. The melody includes every note found within the three melodic segments played earlier in order to demonstrate that all “characteristics” of this melody have been learned and put together. ST created the melody, to which I made some minimal adjustments in order to fit it to the time signature. After the melody has concluded, the cello ostinato and textural out-of-phase gestures (violins and viola) found at the beginning of the section return. However, they have all being transposed on their return to begin on their lowest open string. Furthermore, the unorthodox rhythms found in the cello at the beginning of this section have been ‘corrected’ (through simplification) by lining them up with eighth notes. Having this pattern align with eighth notes was to give the feeling that the model has learned some information from the inputted musical material, demonstrating how with each pass through the multi-head attention stage the model will improve its ability to notate music.

3.2.4 III – Neural Networks[87]

Artificial neural networks (ANNs) or simply neural networks (NNs) are the most common form of deep learning which extracts information through multiple stacked layers commonly known as hidden layers. These layers contain neurons (also called units as they truly have little in common with biological neurons), which are connected independently via weights to neurons in other layers beside it. Often neural networks involve dense, or fully connected layers meaning that each neuron in any given layer will connect to every neuron of its preceding layer (Fig. 18). This allows the network to learn increasingly intricate details or to be trained by the data passed through each subsequent layer. Part of what separates deep learning from other forms of machine learning is its ability to work extraordinarily well with unstructured data. This refers to any data that comes without prearranged labels or features such as audio, images or video. Using many stacked layers, a deep learning algorithm can actually learn to organize and associate features and labels itself when introduced to unstructured datasets. This is accomplished by the hierarchical manner in which a deep multi-layered learning algorithm provides progressively intricate details with each passing layer, allowing for it to break down a highly complex problem into a series of simpler problems.

[87] The first paragraph contains an excerpt adapted from my contribution “Deep Learning” in Philip Frana and Michael Klein (eds.), *Encyclopedia of Artificial Intelligence: The Past, Present, and Future of AI.* (Santa Barbara: ABC-CLIO, 2021), p.112-114.

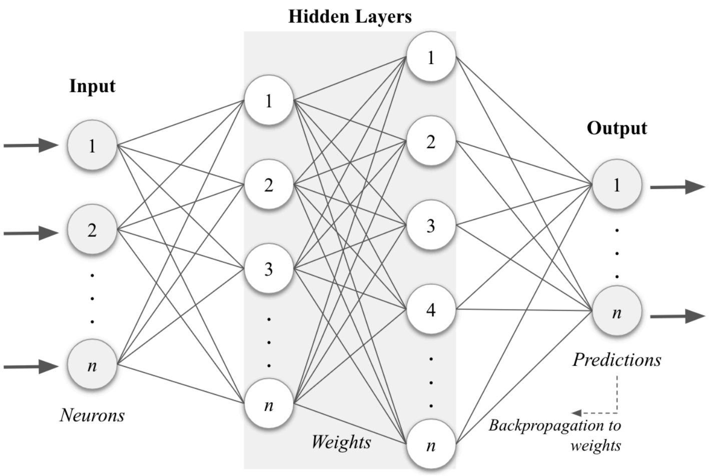


Figure 16: Example architecture of an artificial neural network.

In terms of its use within a transformer-decoder model context, the feed-forward neural network (information only flows in one direction) contains weights which are initially random since the true values are unknown; these weights are updated as the model trains. The goal is to find appropriate values for all of the weights allowing the model to best make sense of the data passed through it, often called 'fitting the model to the training set'[88]. Information learned during the previous section through the multi-head attention mechanism is 'stored' in these weights as a type of memory. As will be discussed in the next section (2.2.5), the weights are tested and changed as required by the model as it gains a better understanding of the dataset. In the transformer-decoder model, this means the purpose of the NN is to process the output from the concatenated attention layer to better fit the data on the next pass through, or the next training step.

Different aspects of NN processes were metaphorically featured within the third movement. Here, the movement considers the notion of stacked, densely connected layers and the process of drawing information from unstructured data. Imitating stacked layers was accomplished by structuring the overall form to include three repeated A and B sections (Fig. 17). The first repeat (mm. 160-229) is imagined as some unstructured input

[88] Trevor Hastie et al., *The Elements of Statistical Learning: Data Mining, Inference, and Prediction (2nd Ed.)*. (New York: Springer Science+Business Media, LLC, 2009). p.395.

data being passed to the NN in order for patterns and representations to be learned. The following repeated sections (mm. 230-299 & mm. 300-363), metaphorically representing a two-layered NN (comparable to that found within a transformer-decoder model), exhibit successive modifications to pitch and rhythmic content. These modifications over time imagine some aspect of the input dataset is being uncovered, represented musically by the pitch and rhythmic changes. Thus, the NN processes of stacked layers and how they might draw representational information from unstructured datasets are musically represented by form, pitch collection modulations and rhythmic alterations.

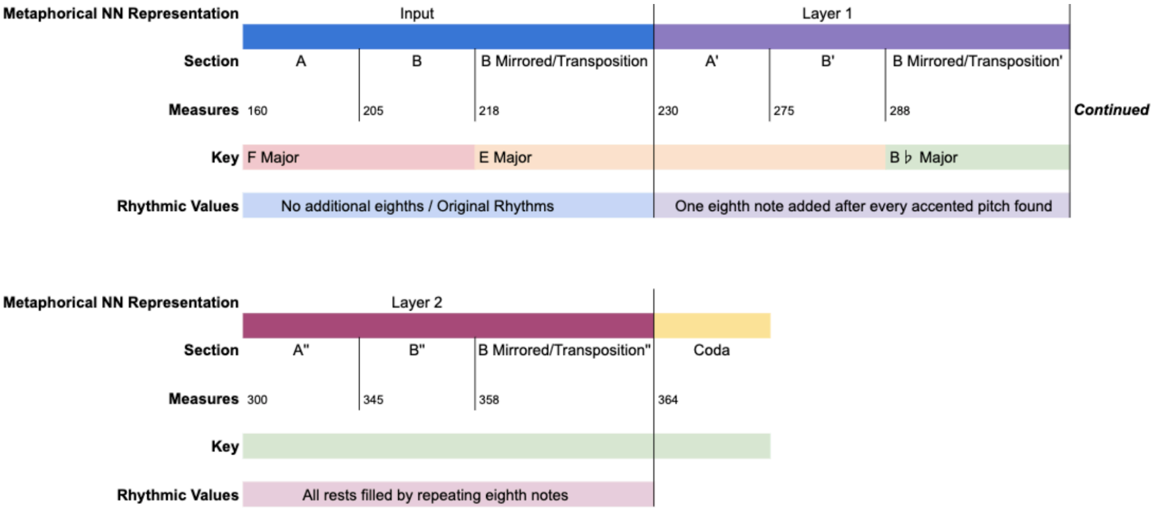
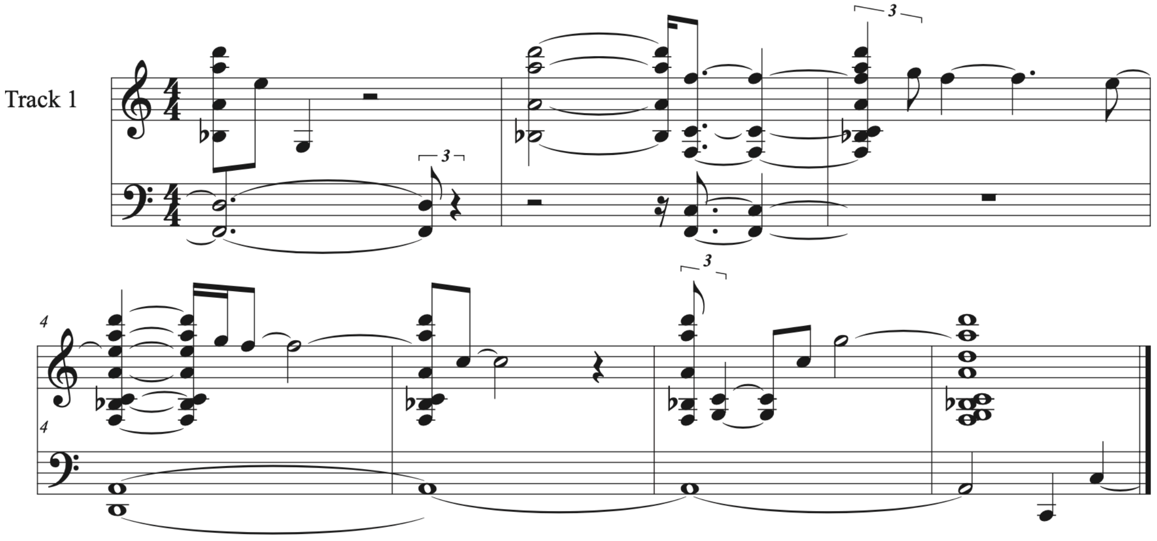

Metaphorical NN Representation
Input
Layer 1
Section
A
B
B Mirrored/Transposition
A'
B'
B Mirrored/Transposition'
Measures
160
205
218
230
275
288
Continued
Key
F Major
E Major
B♭ Major
Rhythmic Values
No additional eighths / Original Rhythms
One eighth note added after every accented pitch found
Metaphorical NN Representation
Layer 2
Section
A''
B''
B Mirrored/Transposition''
Coda
Measures
300
345
358
364
Key
Rhythmic Values
All rests filled by repeating eighth notes

Figure 17: Overall form of the third movement.

Rhythmic and textural densification, through the filling of rests with each repeated A and B section, are used throughout the movement to represent the process of uncovering patterns from unstructured data in reverse. The initial musical content therefore is “buried” amongst repeated pitches added with each repeat of the A and B section. Owing to the accented pitches used in the initial A and B section setting, this music can still somewhat be discerned by the end of the piece. This represents the idea that amongst the unstructured data, some worthwhile content exists that the NN is intending to find (although reversed within this movement). The opening material for the movement (representing the hypothetical material the NN is looking to uncover) was generated by ST using a *temperature* of 1.7 and a *top p* of 1.3 (Ex. 10). I took this generated excerpt and diminished the value of the majority of rhythmic content to create the quick moving passage as seen in Example 11. Having been inspired by this initial content generated by ST, I composed the B section in a similar style without any input from ST (Example 14). Both examples show the space in between each accented staccato chord which will be filled over time with the two repeats of these sections. Furthermore, in an effort to avoid the movement sounding stale, initial instrumental parts are swapped at each new repeated section with other instruments. By swapping parts, the repeated material undergoes timbral and register changes (parts are transposed to fit new ranges as needed) and becomes less obvious to the listener. Figure 18 charts the passage of material amongst the quartet over each repeat of the A and B sections. I have colour-coded these exchanges to show examples from the score (Ex. 11-16)



Example 10: Initial musical passage generated by ST for "III – Neural Networks".

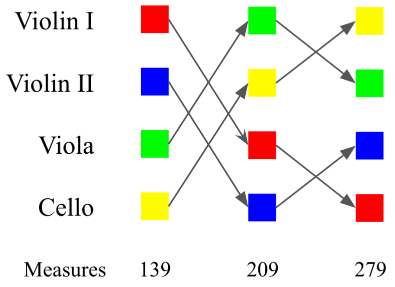
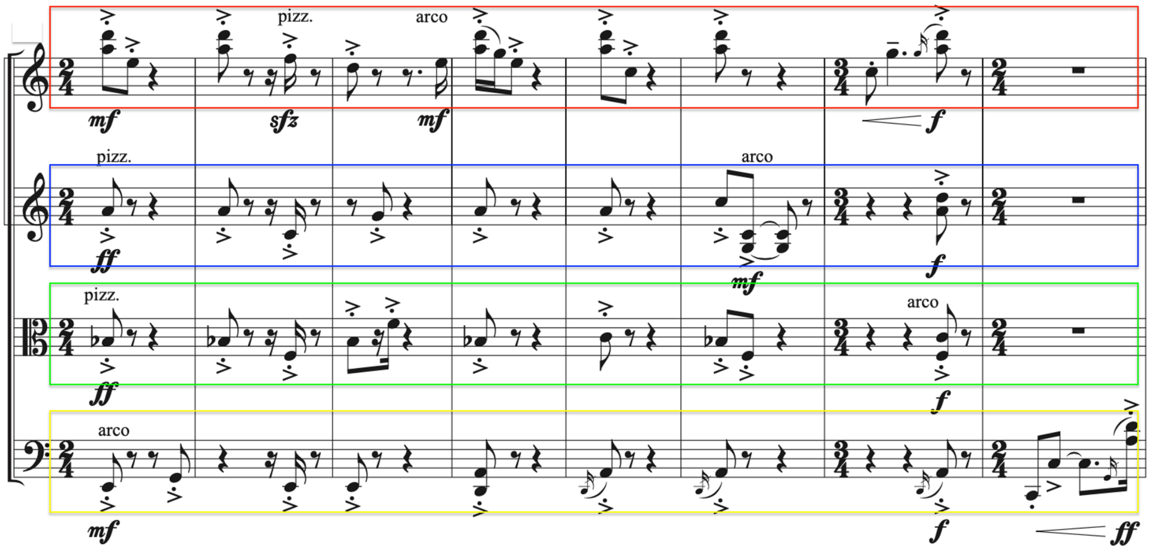
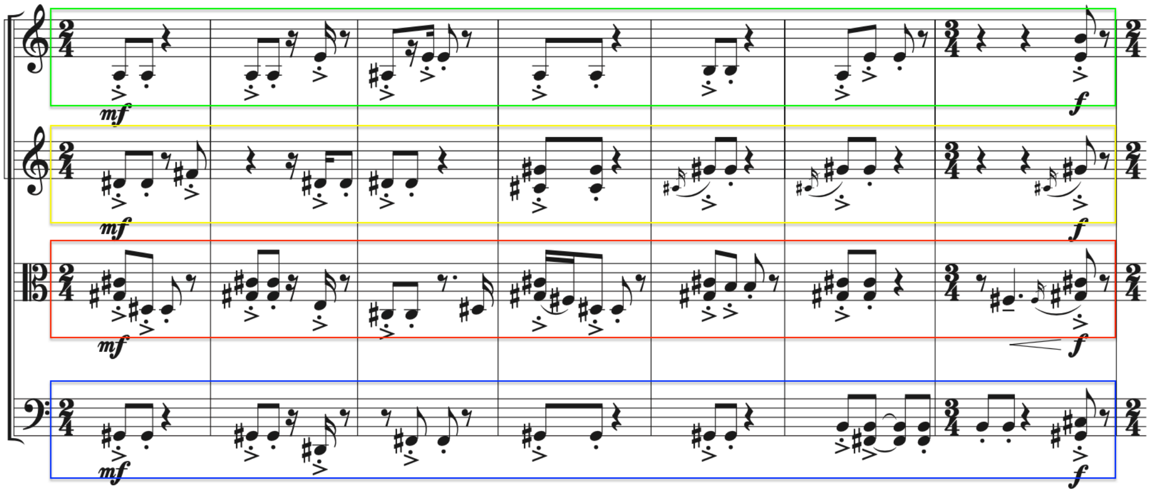


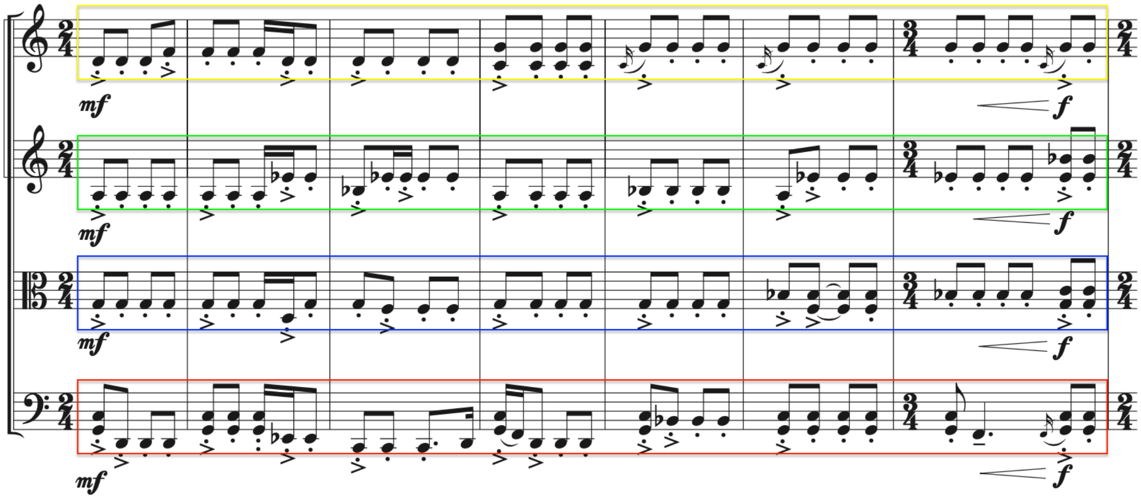
Figure 18: The trading of parts over the movement.



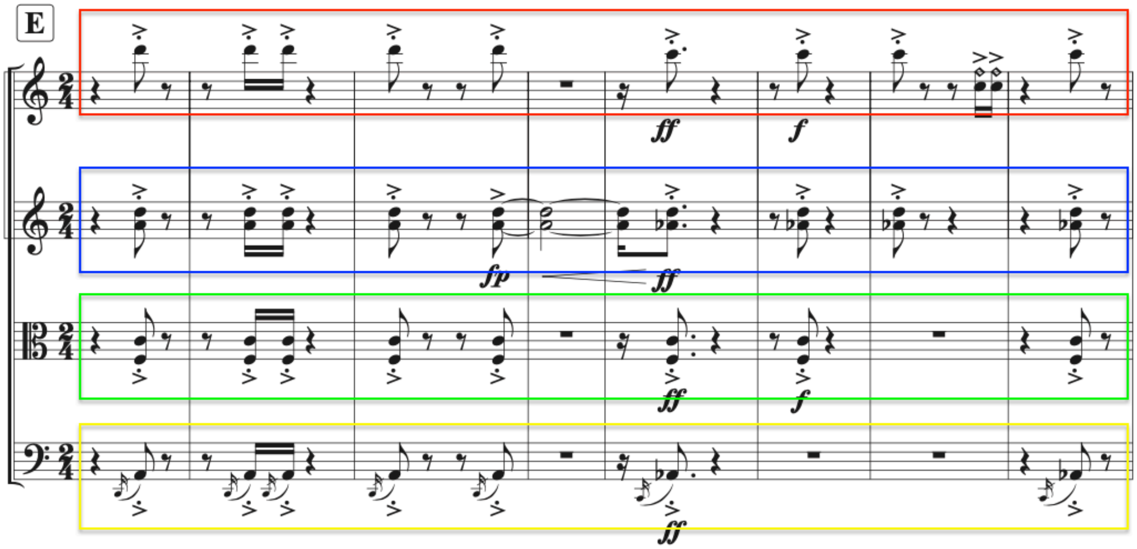
Example 11: Beginning of the original A section.



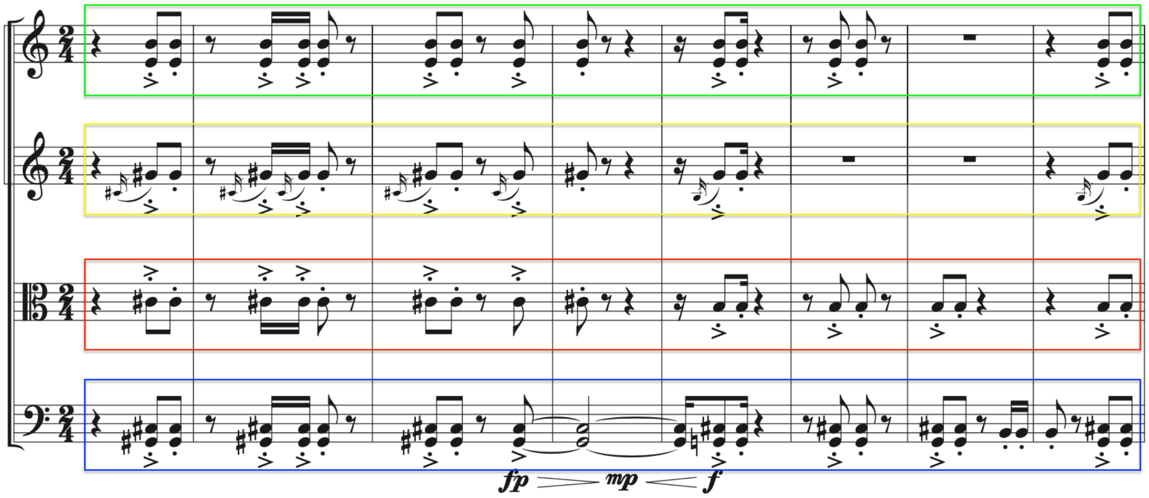
Example 12: Beginning of the first repeated A section.



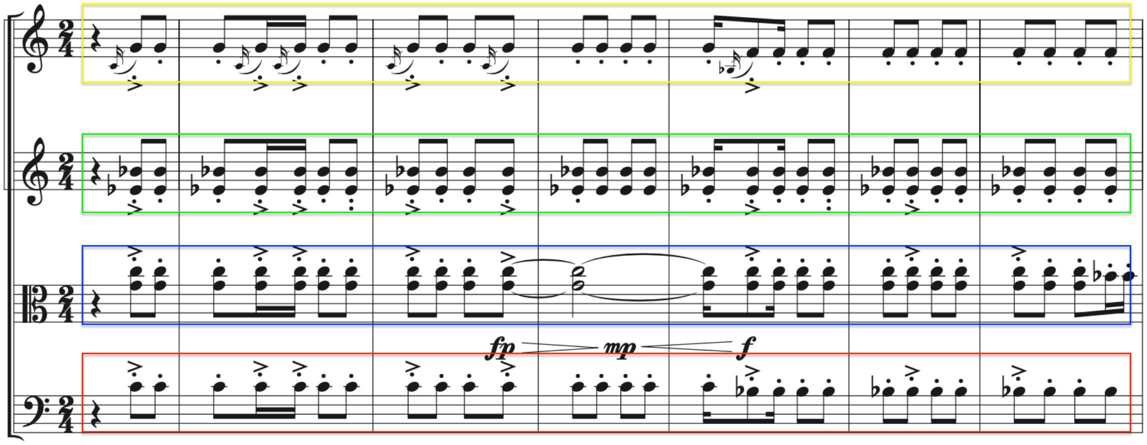
Example 13: Beginning of the second repeated A section.



Example 14: Beginning of the original B section.



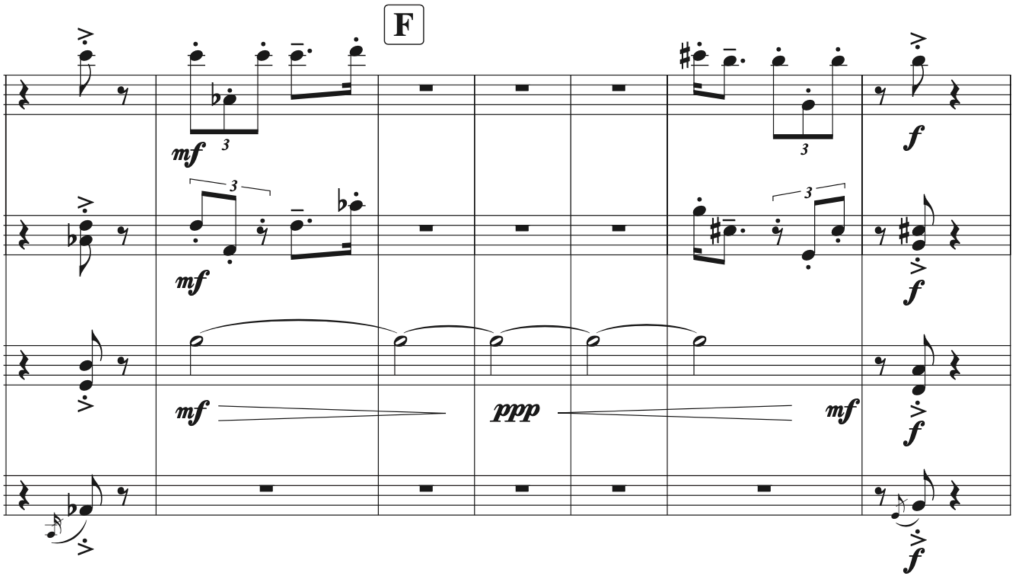
Example 15: Beginning of the first repeated B section.



Example 16: Beginning of the second repeated B section

To keep the movement from feeling too square by having all musical changes occur with each restatement of the A section, pitch collection modulations do not follow this pattern. As seen in Fig. 17, modulations occur within the original B section and the first repeated B section. An overall process unfolds where the music attempts to move towards a target pitch collection made up of the same pitches found in the beginning of the next movement. The first one-flat pitch collection (F, G, A, B♭, C, D, E), was determined by the artificially generated excerpt from ST (Ex. 10). The first modulation (mm. 218) transposes the initial pitch collection down a minor second to obtain a complementary set that when combined makes up the 12-tone aggregate (‘E, F#, G#, A, C, C#, D#.'). Encompassing all of the pitches within the 12-tone aggregate is the second way this movement metaphorically represents the passing of unstructured data through a NN where some pattern is uncovered by the end, in this case the pitch collection belonging to the next movement. This is exactly what occurs during the second modulation (mm. 288) with the move to the pitch collection B♭ major, which is indeed the same collection occurring at the beginning of the next movement. In summary, the movement involves two overall processes (one forwards and one in reverse) to imitate a NN deciphering patters within unstructured data through changing pitch collections and an unfolding rhythmic pattern.

To musically represent the idea of fully connected layers within a NN, I have attempted to demonstrate this in two ways throughout the movement. The first occurs by pitch collections that are connected via a pivot pitch, some pitch common to both collections. Both instances of the pivot pitch can be found in m. 213 and later in m. 283 where a literal connection (a long held sustain note over 5 measures in first the viola and then in the first violin) occurs between the two halves of the B section. The second attempt to illustrate this process again occurs between these two halves of the B section where after modulation, the first half of the B section is mirrored exactly in the second half. To summarize, the B section consists of two halves where the second is a transposed and mirrored copy of the first. The attempt here was to link or draw connections between every pitch found in both halves as their mirrored and transposed counterpart exists on either side of the sustained pivot pitch. Thus, these halves are metaphorically "fully-connected".



Example 17: Mirrored rhythms in mm. 212-219.

In addition to these processes, the texture of this entire movement which consists of a dense setting of many staccato vertical sonorities with few solo moments for individual instruments is inspired by the many neurons within a neural network. After the final repeat of the B section, a brief coda section follows which fragments and repeats rhythmic material from measures 364-366 as an ostinato. This material breaks away from the strict repeats of everything preceding it to connect to the next movement. Beginning with the first violin, one by one each instrument switches from playing the rhythmic loops to hold a long-sustained pitch which matches the pitch that they will play in the next movement (although in a different octave). With each entry of the sustained note, the dynamic of the ostinato drops, continually getting softer and fading out the rest of the movement.

3.2.5 IV - Loss & Gradient Descent

Directly influenced by Iannis Xenakis' work, the fourth movement reimagines the functions of calculus as a means of selecting pitches, register and durations. "IV- Loss & Gradient Descent" is the final movement in which a process belonging to the model architecture is emulated, in particular by the model's ability to "learn", or optimize itself, through each training step by minimizing a loss function[89] through gradient descent[90].

The loss function for a transformer-decoder model is categorical cross-entropy (Appendix A). Briefly explained, the equation calculates the distance between a model's output probability distribution (its guesses) and the correct targets with each subsequent training step[91]. Targets consist of a vector of binary yes (1) or no (0) answers and guesses consist of a vector of normalized probabilities[92] (see Fig. 19). To summarize, minimizing the loss function can be thought of as the model learning by reducing the distance between each guess value and its intended target value. A model that returns a loss function of 0, can be said to be reaching each target prediction every time.

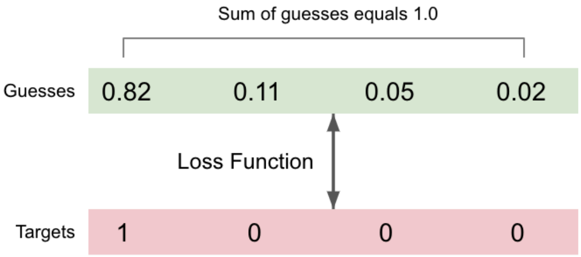


Figure 19: A visualization of guess and target vectors being compared. The sum of all guess probabilities must equal 1.

To communicate the results of the loss function with itself and whether or not these results are continuing the desired minimization each time, gradient descent is applied. Gradient descent comes from calculus where the derivative (i.e., rate of change) of the loss function is calculated at each training step in order for the model to determine

[89] Sometimes called the cost or objective function (Goodfellow et al., 2016, p.79). I will refer to it as the loss function throughout.

[90] Goodfellow et al., 2016, p.79.

[91] Géron, 2019, p.149.

[92] The softmax function "calculates the exponential of every score, then normalizes them (diving by the sum of all the exponentials)" to return "logits" or normalized probabilities (Géron, 2019, p.148).

in what direction to incrementally change its weights (parameters which can affect the resultant guesses) to return a slightly minimized output from the loss function[93]. Generally, a model runs through as many training steps as required to reach a plateau called the “global minimum,” when the derivative no longer demonstrates a negative slope (Fig. 20). That is, its rate of change is zero. While not always resulting in a loss function equaling 0, this is where the model has likely reached a point where it can no longer derive enough new information from the dataset in order to continue reducing the loss function. In other words, training generally is complete at this time and any further training is unlikely to benefit the model (Fig. 21).

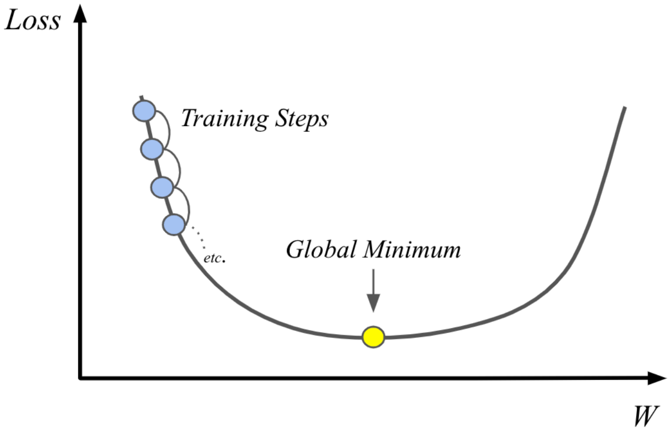


Figure 20: A visualization of gradient descent. Loss is minimized with each training step until the global minimum is reached. *W* is for weights, the parameters the model updates with each step.

[93] Goodfellow et al., 2016, p.80.

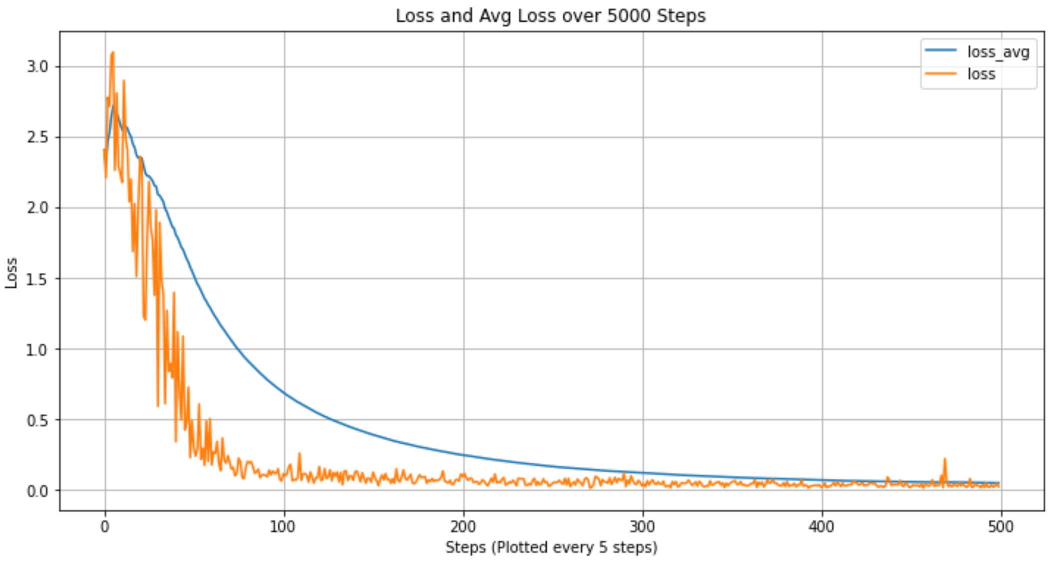


Figure 21: Example of loss and average loss plotted during the training of a model. Loss has reached ~0.1 over 500 training steps meaning the model is making very few errors at this point.

The inspiration for the best way to reflect these mathematical formulas musically, came from the pieces *Achorripsis*[94] and *Metastasis*[95] by Iannis Xenakis. In his piece *Achorripsis*, Xenakis uses Poisson's probability formula, the law for the appearances of rare random events in order to determine the sequence of sonic events and their probability of occurring from a matrix of these events that he had established[96]. In *Metastasis,* Xenakis sketched out how to shape glissandi using geometric models such as the golden ratio (Fig. 22). A combination of these two pre-compositional methods provided the basis for the overall form of "IV – Loss & Gradient Descent" (Fig. 23). Instead of applying many musical elements to a matrix which can then be manipulated using a mathematical function as in *Achorripsis*, I elected to simply control the pitch register of the quartet to mimic the process of gradient descent. A pitch register chart was

[94] Xenakis, 1958.

[95] Iannis Xenakis, *Metastasis.* (London: Boosey & Hawkes, 1954).

[96] Xenakis, 1992, p.29-32.

first sketched out according to a possible visualization of gradient descent over time as it passes through local minima in search of the global minimum (Fig. 23), in this case, the lowest register possible for the quartet. There is a nice symmetry between the final and lowest possible pitch in the quartet, a ‘C’ or 0 in set theory notation, since a 0 in regard to loss would mean a model is making all correct predictions. A loss of 0 is what the model strives for during training.

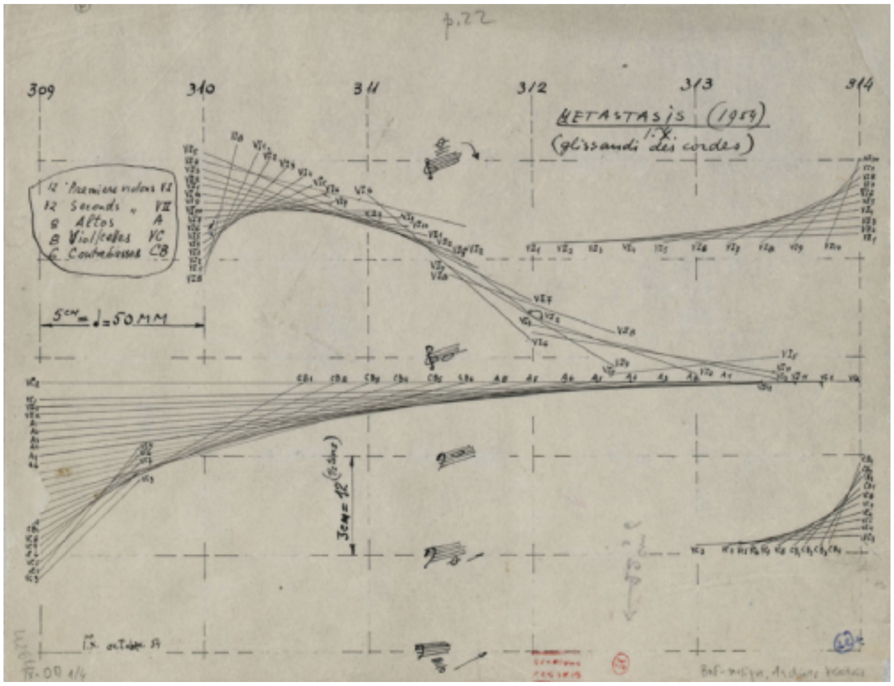


Figure 22: Pre-compositional sketch for Xenakis' *Metastasis*[97].

[97] Varga, 1996.

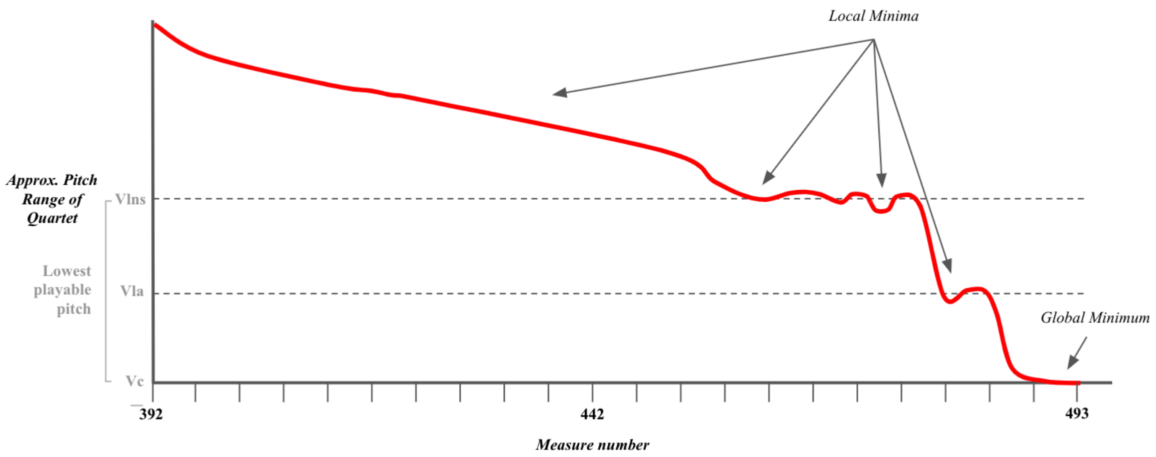


Figure 23: Pre-compositional sketch for the fourth movement demonstrating loss minimized over the duration of the movement.

As the output of the loss function changes incrementally over some period of time, a slow-moving process akin to methods used by minimalist composers, specifically Arvo Pärt, provided the means for the quartet to follow the pre-composed register shape (the gradient descent line) over the length of the movement. Pärt's *Stabat Mater* (1985) features 2 violins and a viola starting in their upper registers which slowly descend around two octaves over 54 measures before the introduction of the choir. This introduction of the string is governed by a slow canon (played 4 times) where the stepwise descent of the strings outlines the tonic triad of A minor[98]. My fourth movement was inspired by Pärt's process to slowly govern the descent of an artificially generated musical passage in the upper register of the violins and viola over time.

This movement involves the introduction of a 6-measure passage of music created by ST (Ex. 14) which is continuously repeated and transposed downwards until it becomes too low for each instrument before finally landing on the final low 'C' in the cello. When the transposition features a subset of playable pitches and unplayable low pitches for a given instrument, that instrument will continue to play the playable ones

[98] Stephen Gregory John Penton. (1998) "The compositional processes of Arvo Pärt: a survey and comparison of two musical styles". *Durham theses*, (Durham University: 1998).

while substituting rests for unplayable pitches. Once all notes become unplayable, that instrument remains tacet until the end of the movement. Each instrument (violins are grouped together) dips below the range of playable pitches at the exact time specified by the graph in Figure 23. In summary, the repetitive transposition of the opening passage represents minimizing the loss function while the pitch register shape over time represents gradient descent.

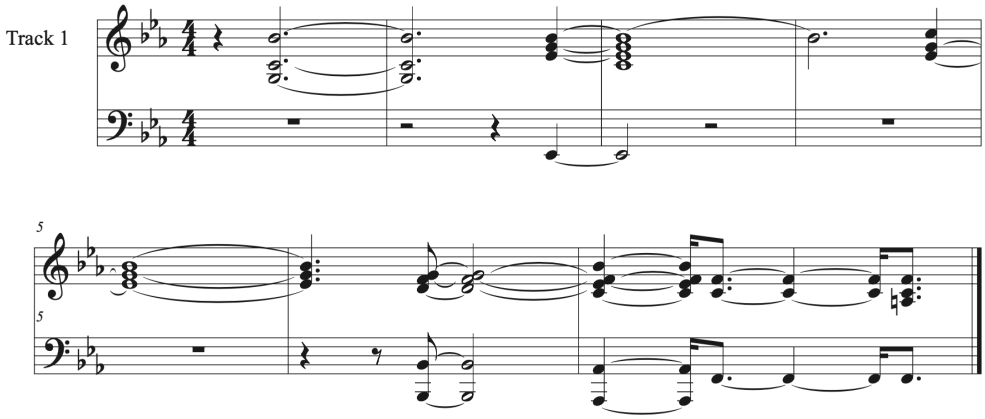
The introductory passage of movement IV was generated using ST (Ex. 14; sampling parameters: *temperature*=1.3 & *top p*=1.2) with some interference by myself. My edits to the autonomously created passage involve modifications to the rhythms, reduction of the passage from 7 measures to 6 (with one in 2/4) and the addition of a simplified cello line. The transposition process for each instrument are as follows:

- Violin I: Transpose down a perfect fourth with every repeat of the introductory material (transpose every six measures).

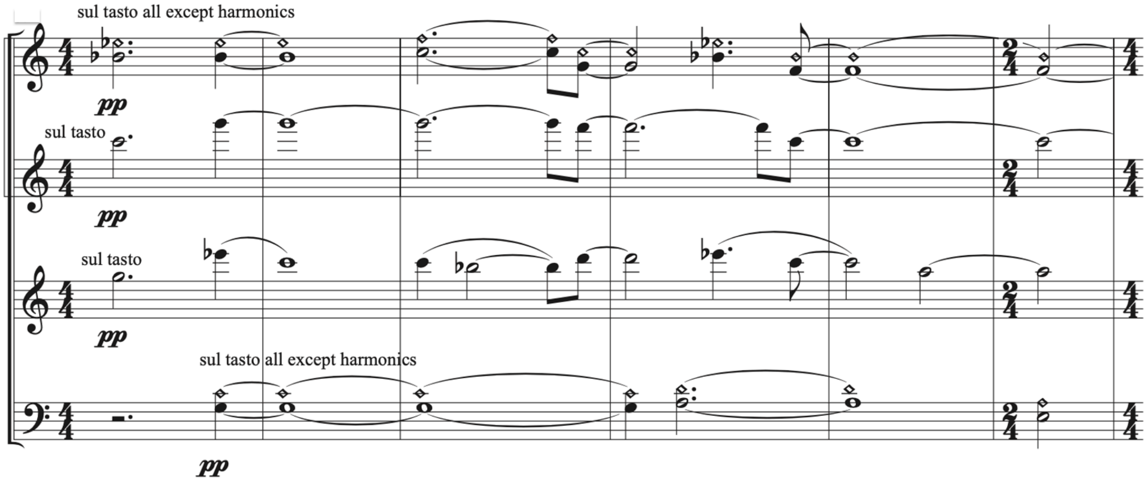
- Violin II: With one repeat of the introductory material, transpose the second half (the last two notes of the four-note passage) down a perfect fourth. With the second repeat, transpose only the first half down a perfect fourth. Every other repeat of the introductory material features a full transposition down a perfect fourth (transpose fully over 12 measures).

- Viola: Transpose down a perfect fourth every other repeat of the introductory passage (transpose every 12 measures).

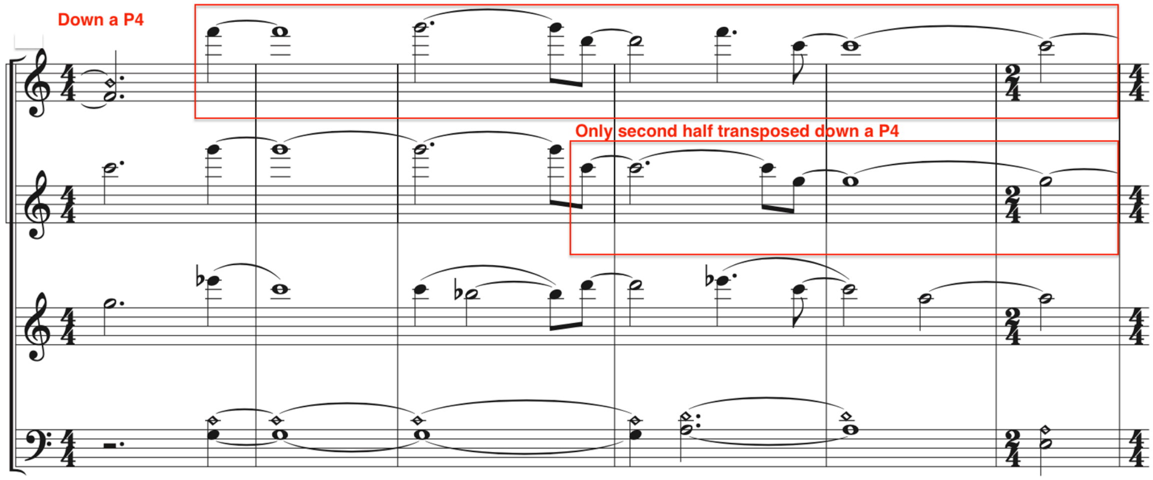
- Cello: Transpose down a perfect fourth every other repeat of the introductory passage (transpose every 12 measures). Cello begins higher in its range as well to allow for it to continue longer than the viola in the movement.



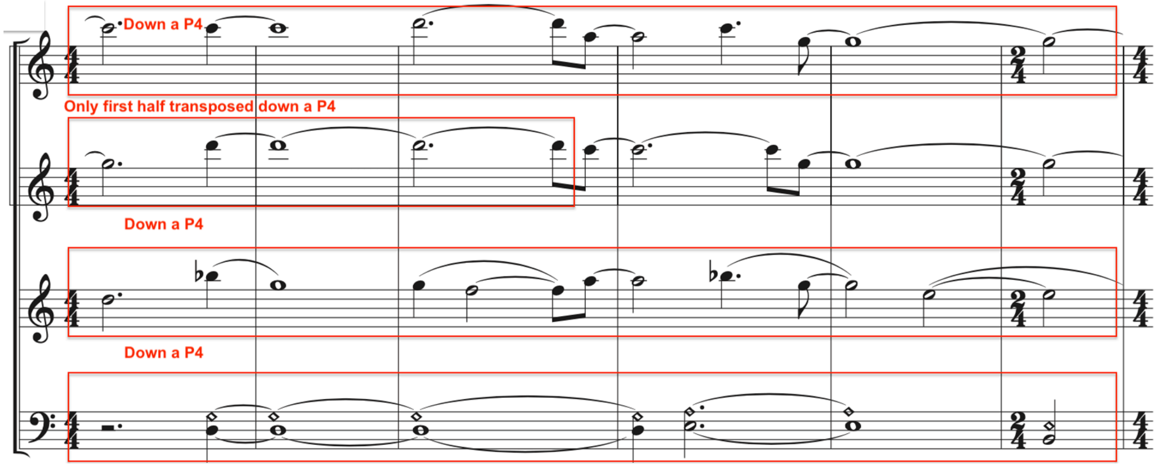
Example 18: Opening passage of "IV – Gradient Descent & Loss" as created by Score-Transformer.


sul tasto all except harmonics
pp
sul tasto
pp
sul tasto
pp
sul tasto all except harmonics
pp

Example 19: Introductory material (m. 293) generated by ST as used in the fourth movement.



Example 20: First repeat of introductory material. Demonstrating the unfolding transposition process.



Example 21: Second repeat of the introductory material. All parts have transposed down at least a perfect fourth at this point.

In an effort to highlight the process itself over any individual moments throughout, the movement features only a single initial dynamic of "pianissimo" and one performance note to play "sul tasto" for the duration of the movement (with the exception of some harmonics in the violin 1 and cello part early in the movement). This draws further inspiration from Pärt's setting of the strings in the opening 54 measures of *Stabat Mater* where the uniform timbre of the strings allows the process to be the focus of attention. Any textural, registral or harmonic changes come solely from the unfolding process. However, there are a few cases where the rules of the process have been intentionally broken. The first involves the final moments of the cello part which should repeat the final six measures one more time. A grand pause follows instead with the conclusion of the movement as I did not feel that a solo line within the movement needed to be repeated. The second rule-break has the violin 1 return in measure 442 which occurs outside of the pitch register graph (Fig. 23) that the process strictly follows. The violin 1 brings back one melodic passage from each of the three preceding movements as a metaphorical return to the start of the model which is what occurs with each training step. The model will loop back to the start many times during the training process and I attempted to briefly nod to this cycle with the return of some melodic moments from earlier in the piece.

3.2.6 V – Hyperparameter Search

The fifth and final movement to include artificially generated musical materials is influenced by the process of a hyperparameter search. A hyperparameter search can occur during initial stages in designing model architecture, where it is used to select optimal parameters in order to achieve the best results from training (lowest possible loss) and, after training as a means to generate musical excerpts most like the supplied prompt. In this discussion, I focus only on the latter. In this case, a hyperparameter search is a systematic and automatic way of testing a range of possible parameter combinations[99] to find a combination that is well suited to the task at hand. As mentioned earlier in section 1.1, best-performance or model creativity during musical generation is subjective, so a manual inspection of each generated excerpt from the search was required to determine a range of parameter combinations that were in my opinion, the best suited to aid the composition of *Music for Self-Attention*[100]. For example, in section 2.2.1, the sampling

[99] Chollet, p.263-264.

[100] See Appendix B for excerpts generated during the hyperparameter search. Note that none of these were used in the piece itself, the hyperparameter search was purely an experiment to gauge the output generated

parameters involving a *temperature* of 0.8 and a *top p* of 1.0 were mentioned as having generated the most closely associated musical material based on the prompt taken from the final movement. However, while composing within a given movement, I could again employ a hyperparameter search with a smaller range of possibilities to see if some other combinations perhaps were better suited when supplied with a different prompt. A user can either decide upon which of ST's parameters to use by trial and error as they become accustomed to the software, or with more time, they could automate this process of testing parameter combinations to obtain a best example from the search. In short, the hyperparameter search in this context allowed me to observe a range of possible generated excerpts, from which I could determine how best to use ST in the creation of *Music for Self-Attention.*

Since this process occurs outside of the model architecture, I felt as though this departure point should be represented musically by highlighting some change from preceding movements. Therefore, this is the first movement to feature functional tonality at times which will eventually be seen to be linked with the final movement. Furthermore, as this is the last movement to feature artificially generated musical materials, the sampling parameters used were: *temperature* of 0.9 and *top p* of 1.1. This combination is only slightly different (higher *temperature* and *top p*) from the best-case parameters. The best-case parameters are a combination of parameters I determined to generate excerpts most closely associated to the prompt in terms of pitch, register or style. Hereafter, the set of parameters referred to as the "best" or "optimal," refer to this combination of parameters (temperature: 0.8, top p: 1.0). Since the overall form of *Music for Self-Attention* is to eventually stylistically blend into the final movement, using the best-case sampling parameters here would have created a musical passage too closely associated with the final movement.

To emulate the process of a hyperparameter search throughout the fifth movement, the movement consists of two main contrasting sections, each involving a

by different sampling parameter combinations. The "best-case parameters" of *temperature:* 0.8 and *top p:* 1.0 were selected owing to pitch, register, gestural and rhythmic similarities to the prompt.

mosaic of many themes which surface, connect with one another and eventually fade back into accompanying material (Fig. 24). None of these themes last for a significant period of time as the movement explores different ideas but never settles on one. Similar to the hyperparameter search, many parameter combinations are explored in attempt to determine the best possible combination. The culmination of this hypothetical hyperparameter search is that the opening of the sixth and final movement was the successful combination chosen which is then fully realized in the final movement. These last two movements connect without pause to musically reflect this connection.

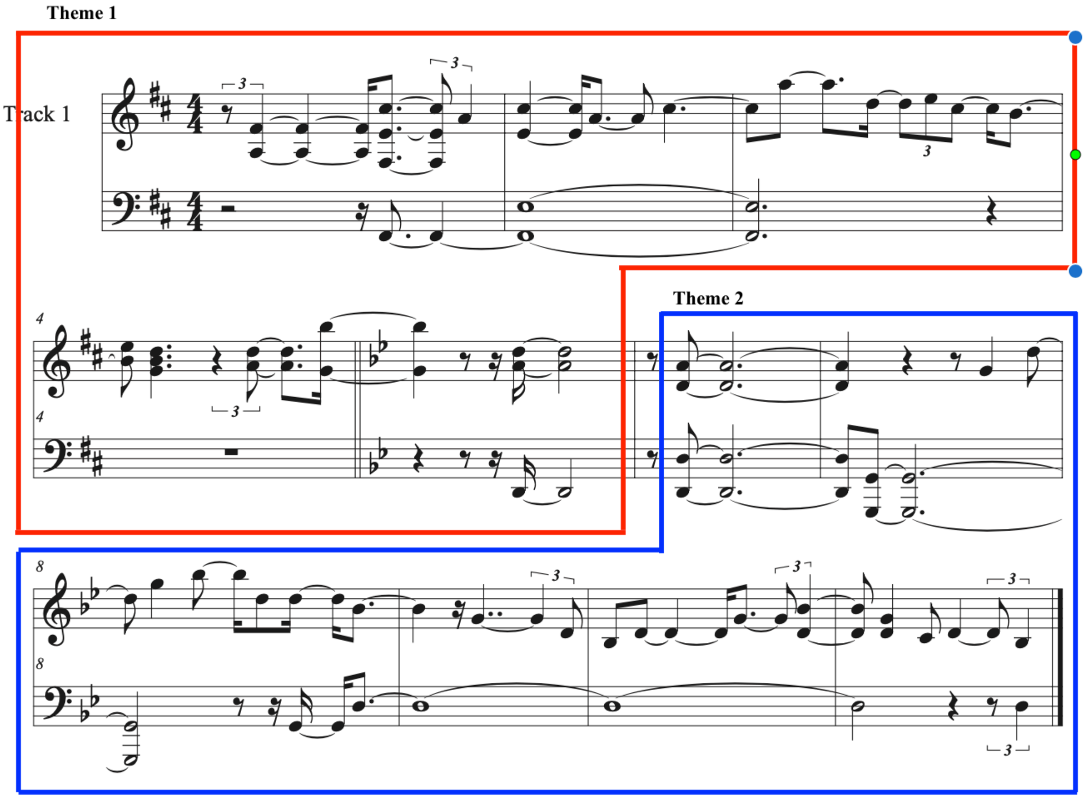
									7	8	8’		10
Themes	1	1’	2	3	4	4’	5	6 (accompaniment)				9 (acc.)	
Measures	495	505	517	522	533	538	543	550	568	577	582	592	597
Sections	**A Section** - *More syncopation, more polyphonic*							**B Section** - *Steady pulse, some generative music*					
Key	**D**							**A♭**		**B♭/Gm**		**Gm**	

Figure 24: Overall form of the fifth movement.

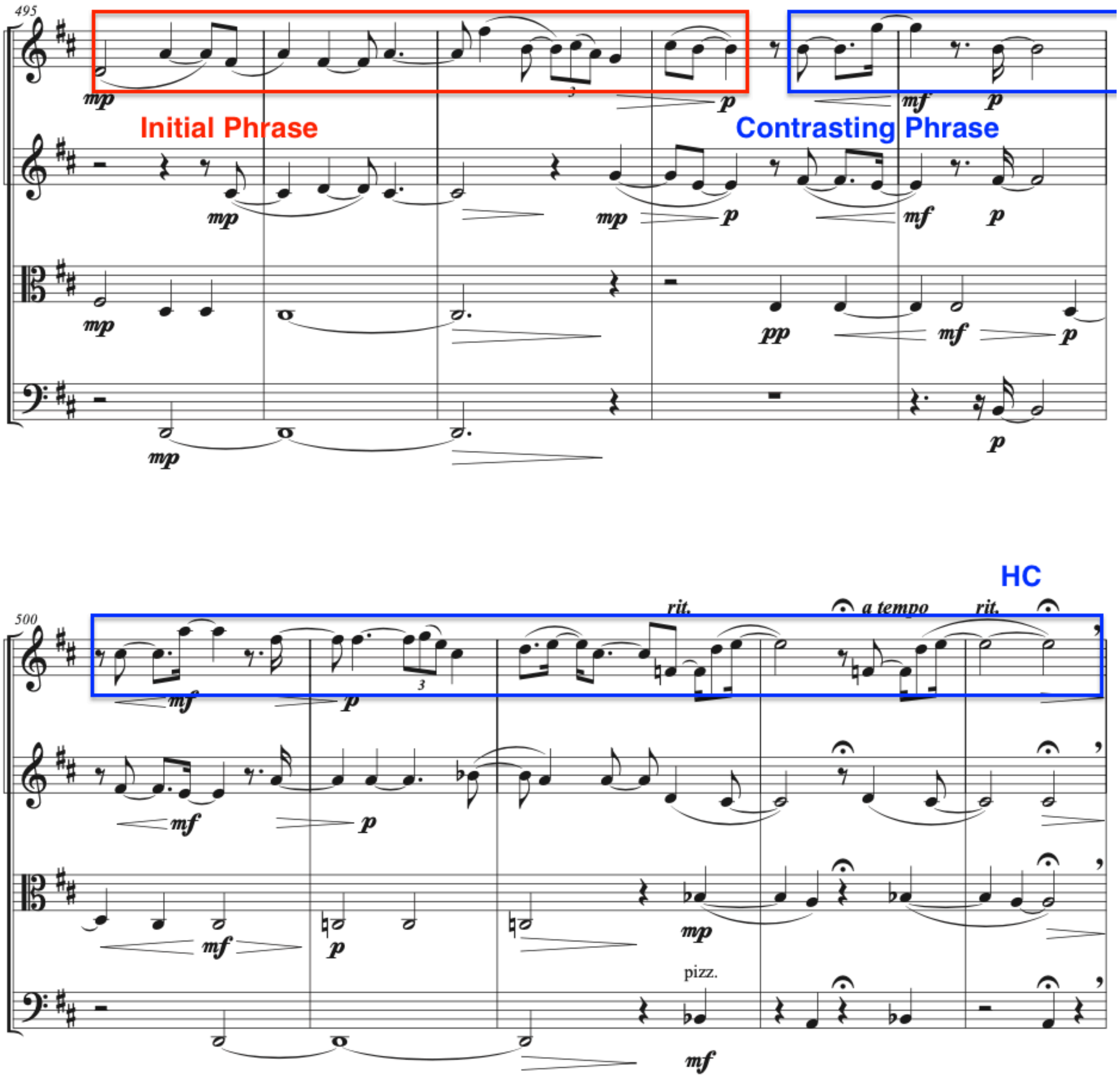
Texturally, section A of the fifth movement generally has more syncopation and polyphony as opposed to section B which features a steadier rhythmic pulse, often driven by the cello and short loops found in the rest of the quartet. This textural change over the course of the movement is as though these hypothetical early excerpts generated by a hyperparameter search (musically represented by themes 1-5) included higher *temperatures*, *top p* or *top k* and which were then slowly lowered over time (themes 6-10). As we have seen throughout *Music for Self-Attention*, increasing the sampling parameters allows the model to make choices that more freely interpret the music of the prompt, often by allowing for less likely pitch or durational choices to be made. The opening melodic passage, which I have noted as ‘Theme 1’[101] (Ex. 22), was generated by ST using a *temperature* of 0.9 and a *top p* of 1.1. As demonstrated in Example 23, I took the first five bars from the artificially generated excerpt in Example 22, transposed it down a diatonic third and treated it as an initial phrase for the resulting opening theme.

[101] I refer to each as theme, however, the term is used loosely. Themes in this case can encompass melodies, accompaniment, sequential material and fragments of past themes.

The excerpt generated by ST was smoothed out in terms of rhythmic values where syncopated rhythms starting on weak triplet or sixteenth note beats were simplified to eighth notes or a half note as is the case with the opening 'D' pitch. The rising gesture at the end of m. 4 was then composed into a contrasting phrase (attaching to the aforementioned initial phrase) by repeating it and moving it upwards as a quasi-sequential moment before ending it with half cadence (m. 504). In a change from the excerpt generated by ST, the theme is repeated but shifted one half note to the right, allowing the cello to sound first instead of the violins and viola. This theme is repeated and referred to as 'Theme 1`' in Figure 24, where the initial phrase returns in exactly the same way except for the rhythmic shift. However, the ending of the contrasting phrase is instead altered by repeating a fragment of the phrase (the triplet in m. 501) which becomes transitional material leading into next theme. This transitional material from mm. 512-516 also includes the long-sustained note from m. 6 in the ST generated excerpt ('D' transposed down a third to 'B' in the cello) which will connect with 'Theme 2' upon its arrival at rehearsal marking 'K'. To emphasize the syncopation occurring in themes 1 and 1', I used the viola and cello to ground the listener's understanding of where the beat lies. If treated exactly as it appears in ST's version without giving context to the beat, the listener will struggle to detect the syncopation.



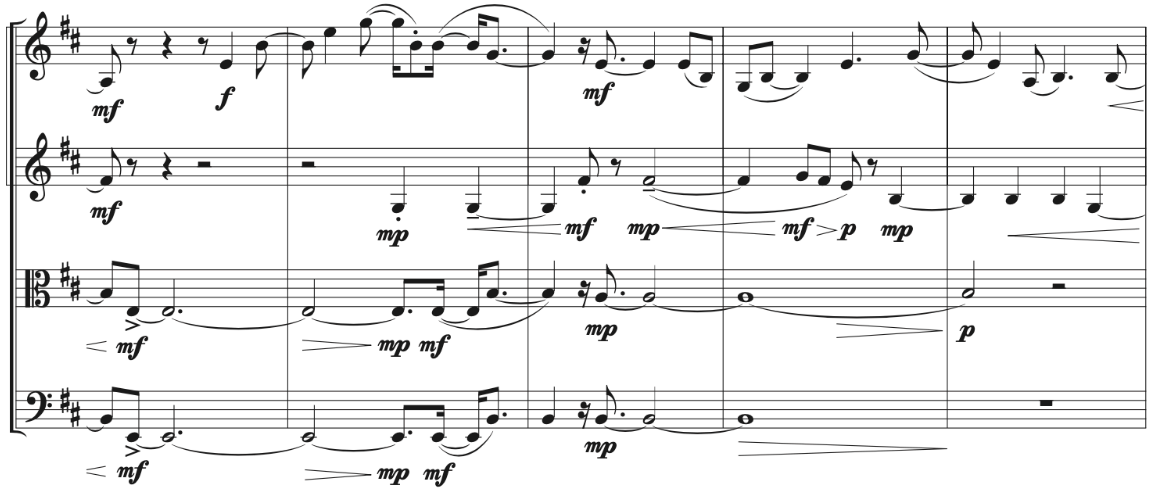
Example 22: Initial ST generated passage used to create two of the early themes found in "V – Hyperparameter Search."



Example 23: Transposed opening measures of “V - Hyperparameter Search.”

A cadence leading back to D major might be expected after ‘Theme 1``’, however, instead this transitional material sitting on a diminished chord moves towards a cadence landing on E minor (rehearsal marking ‘K’). To avoid a fully realized perfect cadence, which didn't seem appropriate at this point in keeping with the overall feel of the entire piece, I removed the third from the supposed B^{7} chord. ‘Theme 2,’ which comes from mm. 7-11 in the generated excerpt from ST, now is used within the piece. Whereas the key in ST’s excerpt points towards G minor, I transposed all pitches down a third (ignoring the two flat key signature) which changes it to E minor. The following themes

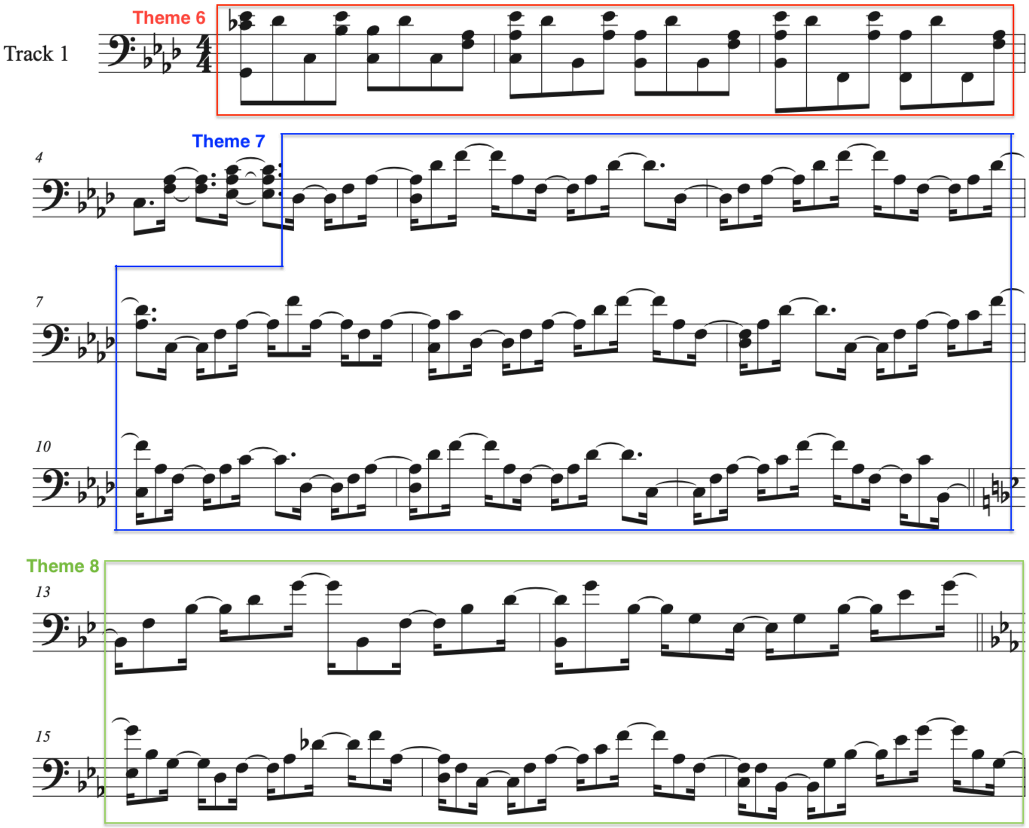
(3, 4 and 5) found within this movement were composed by myself as I had been inspired by the initial passage generated by ST.



Example 24: Theme 2 as realized in the score.

In keeping with the style of the A section, 'Theme 3' features a melody set in the first violin with syncopated rhythms of the accompaniment material found in the second violin and viola. Harmonically, the music has moved away from E minor and is again associated with the opening key of D major. The third theme begins a descending stepwise cello line beginning on an 'E' which continues until it reaches 'A,' signaling the end of this theme. 'Theme 4' begins on a pickup to measure 534 where the stepwise descent continues in the cello from 'D' to 'E' (skipping only 'G' in the process) where the final note of the descent coincides with the beginning of 'Theme 4`' (m. 538). 'Theme 4`' takes a fragment of the preceding theme in the first violin at m. 535 and stretches the primarily eighth note gesture into a quarter note triplet gesture. The viola and second violin play this stretched gesture in m. 538 and again in the viola at m. 539 before the last instance found in the cello at m. 542 where the stretched version has returned to its original eighth note setting concluding 'Theme 4`'. 'Theme 5' consists of a chord progression used to transpose the key from D major to A♭ major. The move from D major to A♭ Major occurs in m. 546 by altering the fifth of the D chord ('A') up to a 'B♭' which creates an augmented dominant of 'E♭'. The passage from m. 547 (beat 3) – m. 549 can overall be analyzed as an extended E♭ major chord while some alterations occur such as the raised seventh in m. 547 (second violin), the included flat seventh in mm. 548-549 (cello), the removal of the root ('E♭') for one and a half beats in m. 548, and the arpeggiated A♭ chord in m. 549 (first violin).

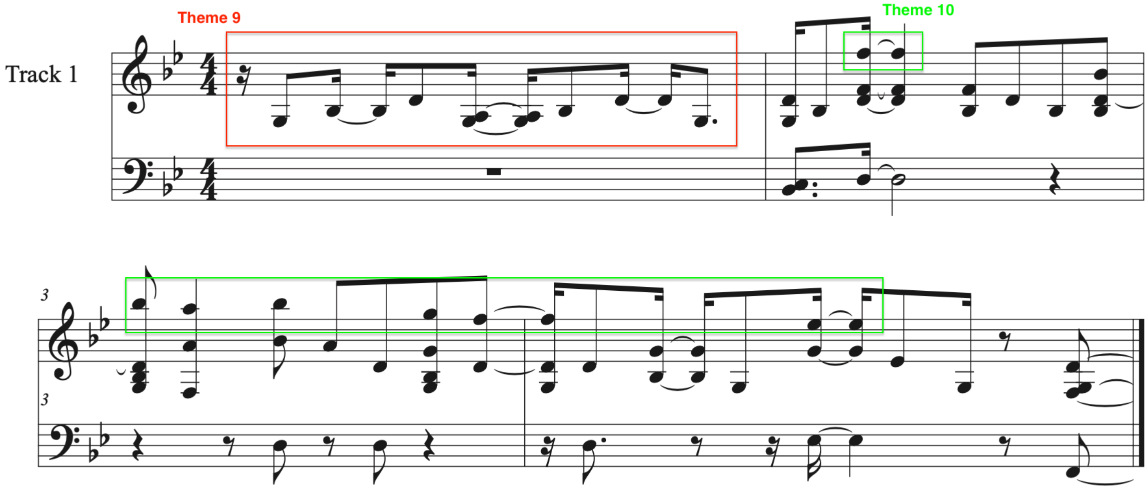
Section B (beginning at rehearsal marking 'L') continues the descending cello line with single sustained notes as seen throughout the end of section A before breaking the sustain into repeated quarter notes as seen in m. 555. One of the more apparent changes in section B is the music now aligns itself more with the beat. 'Theme 6', which will eventually become accompaniment material for themes 8 and 9, was generated by ST as shown in Ex. 25. The lowest voice in these measures (mm. 1-3 in Ex. 25) was used as the cello line from mm. 550-556 beginning after the 'D♭' which was left over from the preceding measure. The upper lines' pitches were modified to adapt the chords to the cello line but largely the feel and rhythmic pattern from this artificially generated pattern has been left as is.



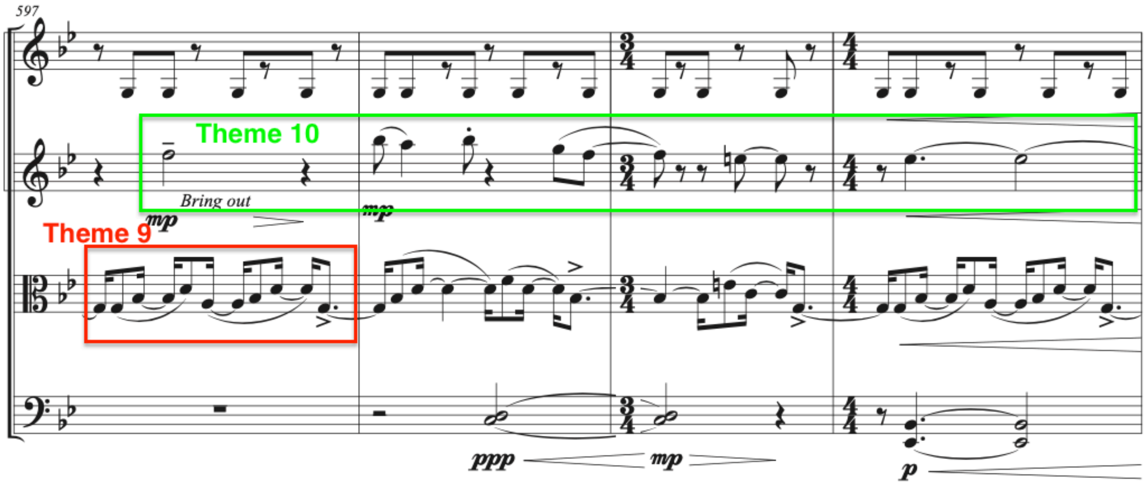
Example 25: ST generated passage showing where themes 6-8 originated.

‘Theme 7’, which also comes from the same artificially generated excerpt (Ex. 25), was implemented in the piece at beat 4 in m. 569 in the first violin part after having been shifted over a sixteenth note. The beginning was altered to feature a false start, m. 568, before leading into the passage created by ST. One aspect of this material that I liked was that the pattern changes at rate different to that of both the cello line and the accompaniment material in the second violin and the viola. Whereas the rhythmic content of the second violin, viola and cello repeats every measure, the first violin part repeats every 18 eighth notes (beginning on beat 4 of m. 569), meaning every time the pattern repeats, it lines up differently with the underlying accompaniment. This overlay of different pattern lengths is similar to the generative music of Brian Eno, where he sought to create ever-changing music that took days before the voices lined up again. I decided that there was a good symmetry between Eno’s work involving early generative music and music artificially generated by ST (a modern type of generative music), so I kept this nod to his work throughout the rest of the movement where possible. ‘Theme 8’ which I have denoted as being in two parts (Fig. 24) breaks from the generative music inspired loops initially (m. 577) to incorporate the material generated by ST exactly as is (Ex. 25) but returns to the generative music loops in measure 582 as ‘Theme 8`’. In measure 582, the first violin spreads the total pitch register further by going up an octave while playing a loop between an arpeggiated B♭ major and E♭ major chord lasting 10 eighth notes before repeating. Furthermore, this section involves the final key change of the movement which includes a B♭ major pitch collection, however, the emphasis on the ‘C’ in the cello throughout and the lack of any dominant chord points towards C Dorian.

The final section (rehearsal marking ‘M’) acts more of as a transitional section into the final movement rather than a dedicated section like the two which precede it. Here, the earlier syncopation is brought back to contrast with the beat-aligned music which will follow it in the sixth movement. The arpeggiated G minor chord in the viola and repeated ‘G’ in the first violin establishes G minor at this point, however, the arpeggiated C major chord in the viola and the later addition of ‘Theme 10’ in m. 597 introduce an ‘E’ which points towards G Dorian. My final homage to Brian Eno’s generative music occurs in this section between the ST-generated viola part (Ex. 26) which makes up ‘Theme 9’ and the first violin part. The viola pattern repeats an arpeggiated G minor chord for two measures before including a B♭ and C major chord, and the first violin repeats a pattern of open 'G' notes and rests every 7 eighth notes. This pattern in the first violin is later swapped to the second violin in m. 602. While Eno's loops often lasted longer and rarely lined up together at all within his pieces, I am trying to show this procedure in a smaller scale where the repeats occur much more often. The final 'Theme 10' comes from mm. 2-4 of Ex. 26 where selections from the top voice which seemed to stand out above the other accompaniment were used to create a melody (highlighted in green in Ex. 26). I used this disjointed melody to connect with the opening melody of movement 6. The first setting of the melody (m. 597) occurs with almost the same pitches as found in Example 26 except for the addition of the 'E' natural to match the G Dorian inflection. The varied repeat of the melody in measure 604 changes the final note to an 'A' (the same pitch the following movement begins on) and includes another 'E' natural. The addition of the 'A' and 'E' were used so that the chord in m. 606 was built using the same pitches ('G', 'A', 'E') as the end of the first movement (m. 62). This parallel was intentional and meant to convey the end of artificial intelligence usage, as ST was only used to aid in the creation of movements 1-5.



Example 26: Excerpt created by ST containing themes 9 & 10.



Example 27: Themes 9 & 10 as realized in the fifth movement.

3.2.7 VI – Generation

The final movement of the piece, of which aspects have been previously mentioned in section 2.2.1 regarding the prompt material, contains music composed without artificial intervention. By removing the artificial element, one intent was to musically represent a model which has learned to such a degree that it appears human-like. At the same time, this movement refers to critiques of AI that suggest that humans' willingness to bestow intelligence on machines often requires them to disregard the significant extent of their own input. Thus, by composing a movement meant to metaphorically convey a trained model's preparedness in generating its own materials without ST, I also attempt to represent this human element of disregard for human contributions to machine intelligence (albeit, taken to extreme by removing all artificial input).

> "This fundamental property of interaction with machines is described by Collins and Kusch (1999) as Repair, Attribution and all That (RAT) – human users constantly 'repair' the inadequacy of computer behaviour, then attribute the results to intelligence on the part of the machine, while

> discounting the actual intelligence that was supplied in the process of repair."[102]

Here the author makes the point that machines still require the presence of humans to be involved in their success and the attribution of success solely to the machine itself is currently unwarranted. Addressing this critique further, a quote by Suchman argues that machines even require human interaction in the form of making sense of errors or "moving the bar" for what might constitute as an error. Simply, humans may observe an error the model makes and determine "it probably meant to do this," then fix the error while still promoting the success of the machine.

> "Human interaction succeeds to the extent that it does, however, due not simply to the abilities of any one participant to construct meaningfulness but also to the possibility of mutually constituting intelligibility, in and through the interaction. This includes, crucially, the detection and repair of mis- (or different) understandings. And the latter in particular, I argued, requires a kind of presence to the unfolding situation of interaction not available to the machine."[103]

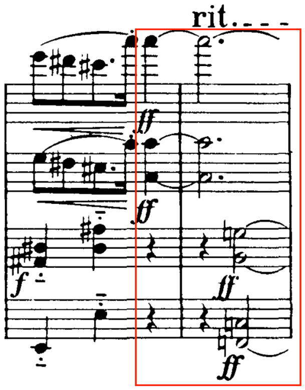
Reflecting on my own intervention both with the model itself and its generated output, I believe these quotes to be very applicable. If my own AI system failed to do something required of it, I attempted to fix it by altering the architecture (for example the addition of relative positional representations, section 1.2.2). Furthermore, when some output was generated and involved poor musical gestures or notational mistakes, I might regard this as a sampling parameter combination failure, or I could edit any small errors that seemed stylistically incorrect. As mentioned earlier, *Music for Self-Attention* is an attempt to demonstrate a symbiotic relationship between human and AI to benefit music composition. However, this relationship really began from the initial conception and architecture of Score-Transformer and not just during the creative development of *Music*

[102] Alan F. Blackwell, 'Objective Functions, Deep Learning and Random Forests', Contribution to *Science in the Forest, Science in the Past*. Needham Institute, Cambridge, 2017, 11.

[103] Lucy Suchman. *Human-Machine Reconfigurations: Plans and Situated Actions* (2nd ed., Learning in Doing: Social, Cognitive and Computational Perspectives). (Cambridge: Cambridge University, 2006), 12.

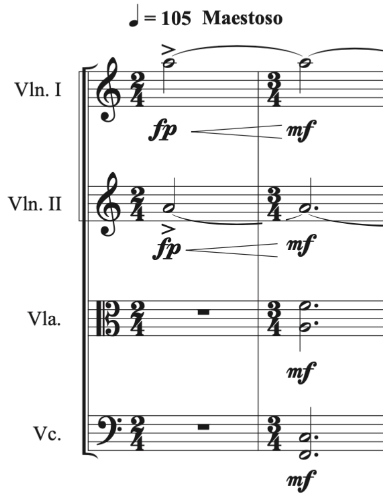
for Self-Attention. In summary, currently music composition stands to benefit more from working in tandem with deep learning technologies rather than relying upon autonomous creation without human intervention.

Two homages occur within this final movement as bookends. The first pays tribute to Schoenberg with a short musical quote from his *String Quartet No. 2 in F♯ Minor, Op. 10* (1908; Ex. 28). Example 29 shows how I have retained his voicing for each instrument except the upper octave found in the first violin as I wanted a reduced register space in comparison to Schoenberg. While some differences in dynamics and rhythmic placement exist, the excerpt from Schoenberg remains largely untouched. This brief excerpt from Schoenberg's string quartet inspired the whole of movement VI. Schoenberg's use of non-musical systems as an aid in music composition provided a significant amount of inspiration for my own non-musical system and therefore an homage to his work seemed appropriate.



Example 28: Excerpt from measures 10-11 of Schoenberg's *String Quartet No. 2 in F♯ Minor, (I), Op. 10* (1908) [104].

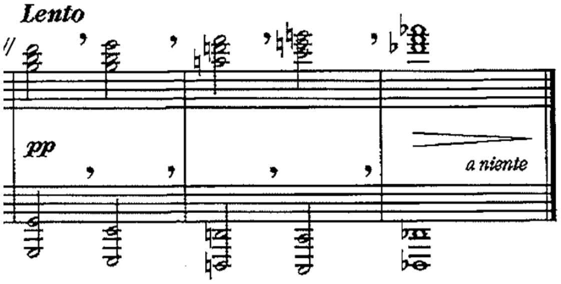
[104] Arnold Schoenberg, String Quartet No. 2, Op. 10. (Vienna: Universal Edition, 1921), p.1.



Example 29: Opening two measures from movement VI.

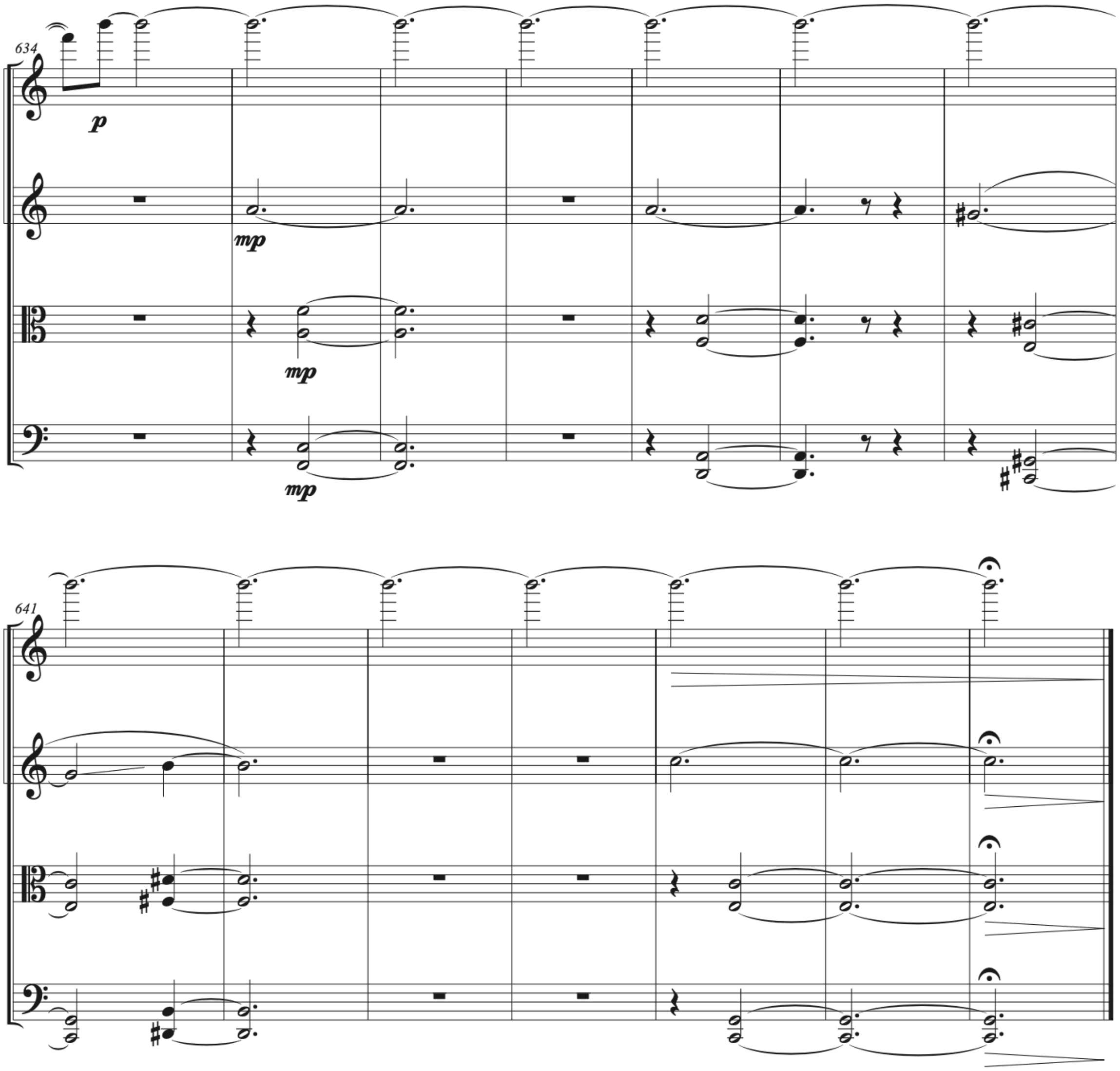
The movement continues in F Major with F Lydian inflections throughout with the inclusion of ‘B’ natural in measures such as 610, 619 and 623. At m. 618, an ‘F’ in the cello becomes a pedal tone underneath the introduction of chromatic pitches (‘D♭’, ‘E♭’) above. This also is the end of the section used as the musical prompt as identified in section 2.2.1. The addition of the chromatic pitches coincides with a reduction in registral space one measure later as the ‘D♭’ in the first violin drops down two octaves and the second violin reaches no higher pitches than the ‘B’. This section (mm. 619-628) retains this more closed-off registral space until m. 629 where the violin arpeggiates an A major chord until it lands on a high ‘E’ which spreads the pitch register space back to relatively how the movement began. The ‘F’ pedal continues until m. 635 with the return of a root position F major chord, demonstrating that this whole passage from mm. 618-635 was just a prolongation of what will be shown to be a IV chord in the context of the movement as a whole. In m. 634, the pedal tone is then replaced from the lowest register (‘F’) in the cello to a pedal tone in the highest (‘B’) in the first violin which lasts throughout the end of the movement, eventually becoming a major seventh in the final tonic C major chord. The predominant prolongation is then continued with the addition of the second homage, this time (mm. 635-642) to Andrew Lloyd Webber with the use of the final chord progression from *Phantom of the Opera* (1986; Ex. 30).

I chose this moment as this musical was one of the first large instrumental scores that piqued my interest in composition and studying music. As one of my favourite chord progression moments from the musical, I decided to quote it at the end of my work signifying a closure to a major milestone in my career. The final chord progression of *The Phantom of the Opera* (Ex. 30) is IV - ii - ♭ii - VII before resolving to the tonic (the unseen key signature in Example 30 is D♭). In my piece, the progression is transposed down a minor second with a few further adjustments. First, instead of voicing the chords of the progression in closed position as Webber had done, I based it on the voicing of the Schoenberg chord from the beginning of the movement (Ex. 28). Therefore, the final progression features a sixth double stop in the viola and a perfect fifth double stop in the cello where possible (mm. 641-642 a perfect fifth is not possible and altered). The second violin entry before the viola and cello (mm. 635-646) similarly alludes to the voicing and rhythm of the Schoenberg excerpt and is used for each chord in the progression except with the B major chord (mm. 641-642). Using this voicing, the Webber chord progression (IV - ii - ♭ii - VII) is then retained exactly with a tonic of C major. Overall, this movement acts as a large-scale plagal motion with the predominant prolongation lasting from mm. 618-642 and the tonic occurring in m. 644.



Example 30: Final three measures of Andrew Lloyd Webber's The Phantom of the Opera (1986)[105].

[105] Andrew Lloyd Webber, *The Phantom of the Opera.* (London: The Really Useful Group Ltd., 1986), p.272.



Example 31: Final thirteen measures of movement VI.

Part 4

4 Conclusions and future work

Throughout the development of *Music for Self-Attention*, the only strict usage of Score-Transformer can be found at the beginning of each movement. Afterwards, ST was used freely when it was deemed necessary and without any sense of obligation. ST was designed to be used in this way as an extra tool at the composer's disposal. Thus, throughout each of the first five movements, there is no specification for how frequently or infrequently ST was used past the initial material. For each of the movements with artificially generated content, I had no idea how the music was going to sound before this material was generated. The freedom to use ST when I wanted and having no preconception for how each movement was going to begin demonstrates a possibility for human and artificial interaction in the creation of new music. I believe this to be a positive early experience towards a future where more composers adopt the use of artificially intelligent software in their own works.

The title, *Music for Self-Attention,* serves two purposes as it relates to the self-attention process within the transformer-decoder model but also towards my own reflexive methods during the composing of this piece. In composing with an artificially intelligent aid (that I also developed), there were many aspects to which I had to pay attention to in order to produce a compelling piece of music. I had to determine what I wanted the AI software to bring to my own compositional methods, pay attention to the music it generated (whether or not it would fit the piece) and make sure I didn't become too reliant on the artificially generated excerpts. In demonstrating a symbiotic relationship between human and machine composers, attention has to be paid to the output of each and how they will fit together. Thus, "self-attention" becomes more than a stage within the model but rather a metaphor of "self-awareness" reflecting throughout the entire process of my dissertation work.

Some future goals for Score-Transformer includes an ability to clean up the generated excerpt before Finale (or other notation software) attempts to quantize it. As can be seen from some of the examples throughout, there are some imperfections which occur from training a model on a dataset involving human MIDI recordings. Anything recorded by a human without quantization is likely to include notes which don't line up with the beat. For example, the MAESTRO dataset used within the model's corpus is entirely made up of human MIDI recordings of piano works (see Appendix D for a Beethoven score compared with Finale's rendering of a MIDI recording of the same piece). Some method of quantizing the excerpts generated by ST before notation software attempts to quantize them could be a beneficial addition. Another possible benefit to the model would be to encode data about how many staves for which the excerpt is intended. This would need to be included in the initial dataset but could be a very useful feature. These examples are a part of my larger goal to incorporate as much musical data as possible into the model without it becoming too unwieldy for use on standard commercial computers running solely off of a CPU without GPU support[106]. Some examples of musical data to include are the addition of microtonal pitches, performance instructions, dynamics and even instrument selections. These additions would allow the model to learn new types of music such as learning contemporary idioms that go beyond the equal tempered system, taking into account timbral considerations and possibly extended techniques.

I also hope to continue this work by adapting the model further into a musical domain. Currently, the model is based upon research utilizing it within a natural language context. Experiments involving the manipulation of the architecture such that it can better interpret the complex characteristics of music and music composition could be important for this type of aid to gain traction amongst composers. This work could go hand-in-hand with the above-mentioned future goals for the model by allowing for more information to be encapsulated during training. Regardless, I believe this existing work already demonstrates the benefits of using deep learning within music composition and I intend to further create music using Score-Transformer or other artificially intelligent aids.

[106] For my dissertation, I ran Score-Transformer off of a 2015 Apple Macbook Pro. See section 2.2 for training resources.

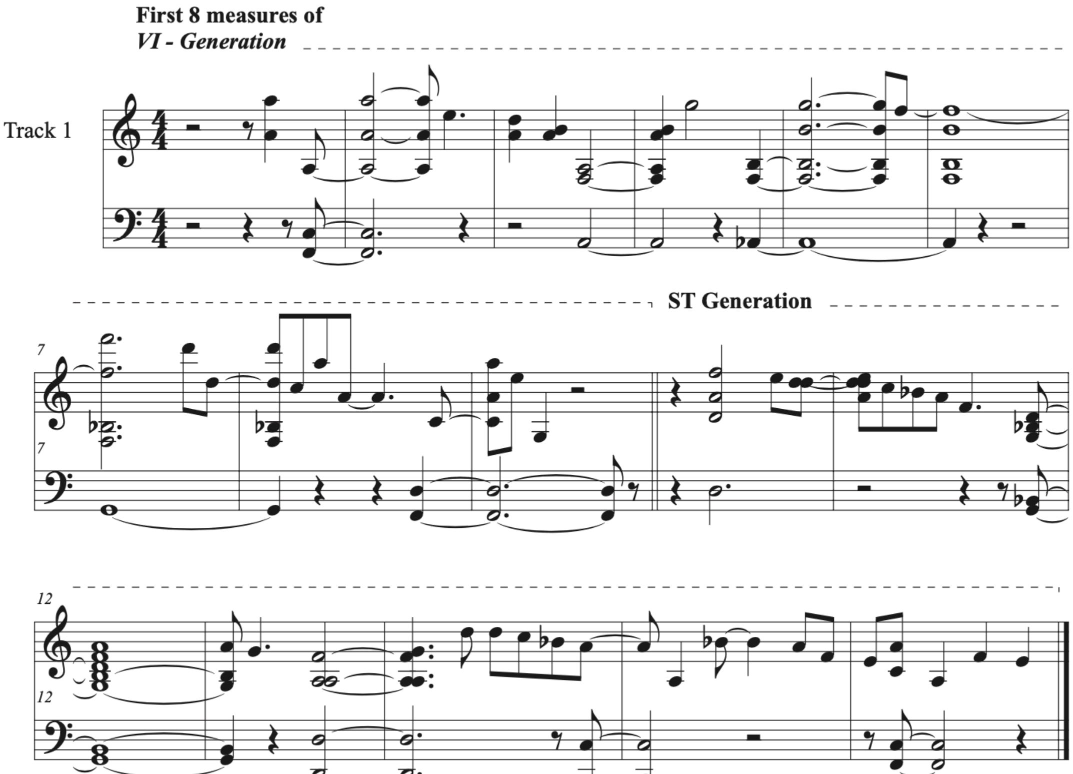
Appendices

Appendix A: Cross-entropy loss equation where T_i are the target values and S_i are the guess values.

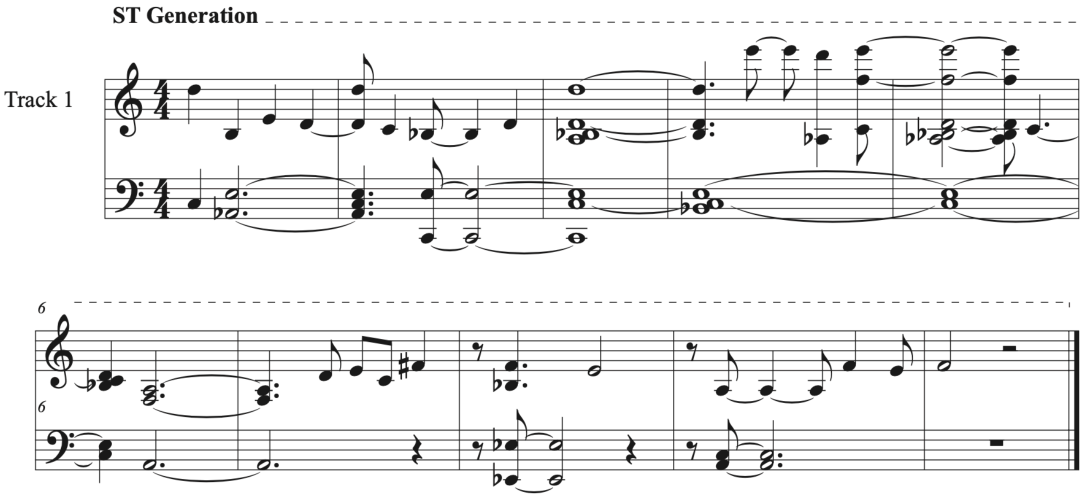
$$L_{CE} = -\sum_{i=1} T_i \log(S_i)$$

Appendix B: Examples from a hyperparameter (sampling parameter) search to find the most stylistically similar material to the prompt. Finale's MIDI quantization was set to eighth notes.

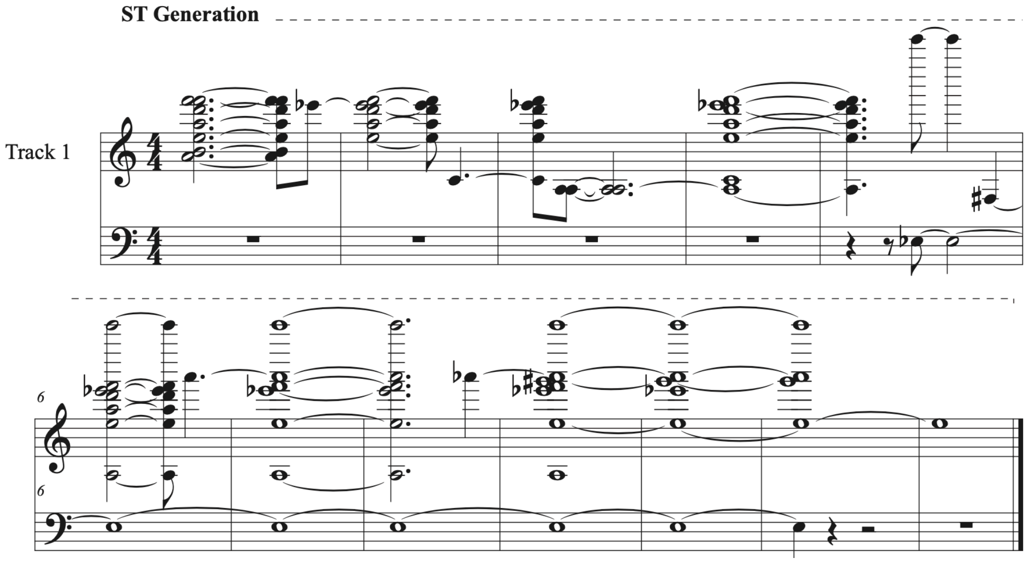
Excerpt #1: *Temperature*: 0.8 | *Top p*: 1.0



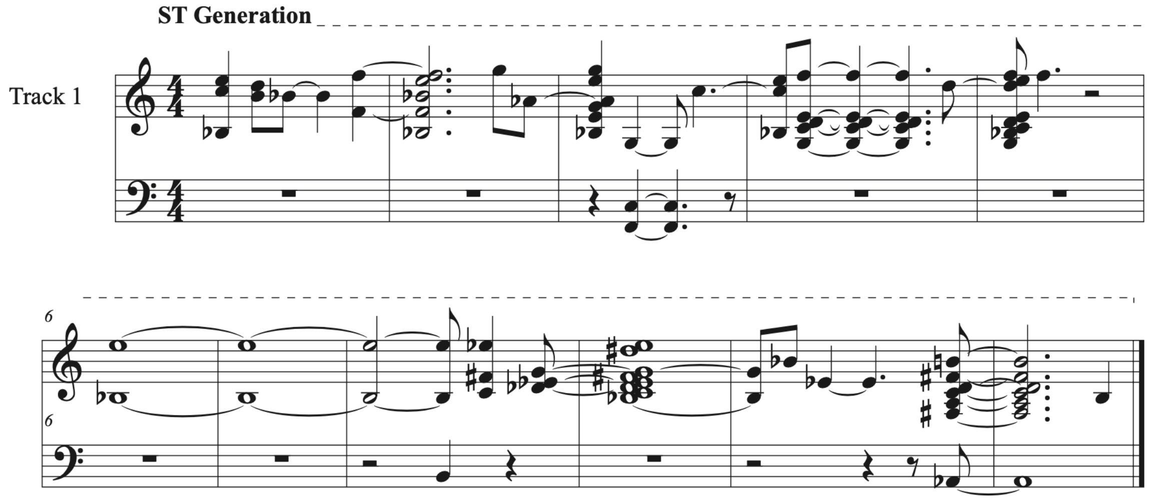
Excerpt #2: *Temperature*: 1.2 | *Top p*: 1.0



Excerpt #3: *Temperature*: 2.1 | *Top p*: 1.0



Excerpt #4: *Temperature*: 1.0 | *Top p*: 1.5



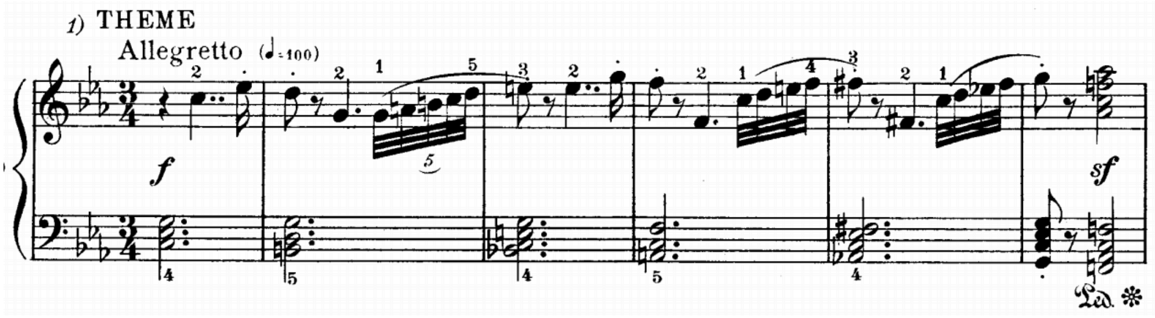
Appendix C: Table of sampling parameters used to generate artificial material used in the beginning of each movement.

Movement	*Temperature*	*Top P*
“I – WTE+WPE”	2.5	1.5
“II – Multi-Head Attention”	2.0	1.4
“III – Neural Network”	1.7	1.3
“IV – Loss & Gradient Descent”	1.3	1.2
“V – Hyperparameter Search”	0.9	1.1
“VI – Generation”	N/A	N/A

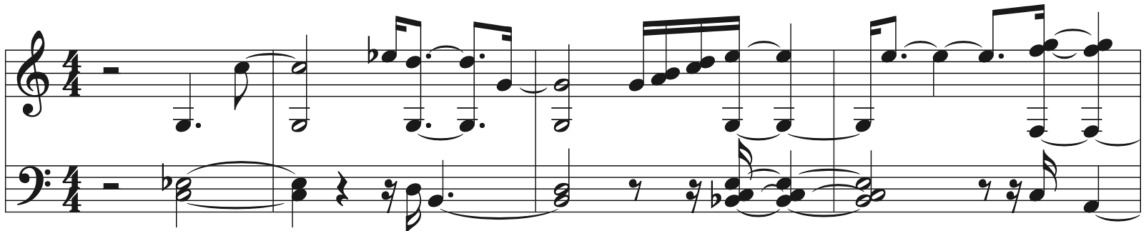
Appendix D: Excerpt from the MAESTRO dataset.

Comparison of the score from Beethoven's *Thirty-Two Variations in C minor*, and a Finale rendered version of the MIDI recording found in the MAESTRO dataset.

Score:



Finale rendered MIDI recording:



Glossary of Terms

Model

Algorithms which are trained to imitate human (or an expert's) decisions.

Natural Language Processing (NLP)

Any computer use of human languages (i.e., English, Spanish & Italian).

Sampling

When a language model attempts to predict the next token in a sequence, it picks (or samples) it from a distribution.

Temperature

A sampling parameter by which the probability distribution is divided by the temperature amount to increase or decrease the probability of next potential tokens.

Tokens

In natural language processing, this refers to elements such as words, characters or bytes. In music, this can relate to elements determined by the user such as notes, chords or durations.

Top k

As many of the tokens in a distribution are likely not applicable when a model attempts to predict the next token during generation, we can remove them from the sample space by utilizing the top k parameter.

Top p (Nucleus Sampling)

Similar to *top k*, we can remove tokens from the sample space by calculating their cumulative probability. Any tokens below a threshold are removed from the sample space.

Length

Number of tokens to be generated by the model. For example, 50 tokens would generate 25 pitch, chord or rest events with 25 duration events.

Bibliography

Bahdanau Dzmitry, Kyunghyun Cho & Yoshua Bengio. "Neural Machine Translation by Jointly Learning to Align and Translate". *In Proceedings of the Second International Conference on Learning Representations (ICLR 2014),* 2014.

Beethoven, L. V. *Thirty-Two Variations in C minor.* (Boston: Oliver Ditson, 1909)

Beltagy, I., Cohan, A., & Lo, K. "SciBERT: Pretrained Contextualized Embeddings for Scientific Text", *CoRR*, (2019). abs/1903.10676. Retrieved from http://arxiv.org/abs/1903.10676

Blackwell, Alan F. "Objective Functions, Deep Learning and Random Forests", Contribution to *Science in the Forest, Science in the Past, Needham Institute, Cambridge*. 2017.

Bouliane, Denys. *Douze Tiroirs de Demi-Vérités.* Toronto: Canadian Music Centre (1981-82).

Bouliane, Denys. "X-rated vs. IN-rated music. Anchors, driftage, parallel worlds: The paths to the Anticostian saga," In *Research Alive @ Schulich* (Jan. 31, 2018). Accessed September 22, 2020, https://www.youtube.com/watch?v=dmTd6bWF5i8&t=2778s

Chollet, François. *Deep Learning with Python.* (Shelter Island: Manning Publications, 2017), p.138-142.

Cope, David. "Experiments in Musical Intelligence". Undated. Retrieved from *http://artsites.ucsc.edu/faculty/cope/experiments.htm.*

Edwards, Michael. "Algorithmic Composition: Computational Thinking in Music," *Communications of the ACM* Vol. 54, no. 7 (July, 2011). Accessed August 21, 2019, 2020 http://people.cs.vt.edu/~kafura/CS6604/Papers/Algorithmic-Composition-CT-Music.pdf

Eno, Brian. *Ambient 1: Music for Airports.* (London: Polydor Records, 1978)

Géron, Aurélien. *Hands-On Machine Learning with Scikit-Learn, Keras & Tensorflow: Concepts, Tools, and Techniques to Build Intelligent Systems (2^nd^ Ed.).* (O'Reilly Media Inc., 2019).

Goodfellow, Ian, Yoshua Bengio & Aaron Courville. *Deep Learning.* (Cambridge, MA: The MIT Press, 2016).

Hastie, Trevor, Robert Tibshirani & Jerome Friedman. *The Elements of Statistical Learning: Data Mining, Inference and Prediction (2nd Ed.)*. (New York, NY: Springer, 2009).

Hawthorne, Curtis, Andriy Stasyuk, Adam Roberts, Ian Simon, Cheng-Zhi Anna Huang, Sander Dieleman, Erich Elsen, Jesse Engel, and Douglas Eck. "Enabling Factorized Piano Music Modeling and Generation with the MAESTRO Dataset." In *International Conference on Learning Representations*, (2019).

Holtzman, Ari, Jan Buys, Li Du, Maxwell Forbes, Yejin Choi. "The Curious Case of Neural Text Degeneration," *arXiv.org* (2020). Accessed August 21, 2020. http://arxiv.org/abs/1904.09751

Huang, Cheng-Zhi Anna, Ashish Vaswani, Jakob Uszkoreit, Noam Shazeer, Ian Simon, Curtis Hawthorne, Andrew M. Dai, Matthew D. Hoffman, Monica Dinculescu and Douglas Eck. "Music Transformer." *arXiv.org (2018).* Retrieved from https://arxiv.org/abs/1809.04281

Loy, Gareth. *Musimathics: The Musical Foundations of Music, Volume 1.* (Cambridge: MIT Press, 2011).

Lupker, Jeffrey A. T. "Deep Learning". In P. Frana and M. Klein (Eds.), *Encyclopedia of Artificial Intelligence: The Past, Present, and Future of AI*. (Santa Barbara: ABC-CLIO, 2020).

Lupker, Jeffrey A. T. "Generative Music and Algorithmic Composition". In P. Frana and M. Klein (Eds.), *Encyclopedia of Artificial Intelligence: The Past, Present, and Future of AI*. (Santa Barbara: ABC-CLIO, 2020).

Lupker, Jeffrey A.T. "Score-Transformer: A Deep Learning Aid for Music Composition." In the *Proceedings of the 21st New Interfaces for Musical Expression (NIME)* conference (2021). Accepted.

Lupker, Jeffrey A.T. *Two Movements for Orchestra.* (UWO Electronic Thesis and Dissertation Repository, 3727, 2016).

Mann, Ben. “How to Sample from Language Models.” *towardsdatascience.com* (2019). Accessed April 20, 2020. https://towardsdatascience.com/how-to-sample-from-language-models-682bceb97277

Nierhaus, Gerhard. Algorithmic Composition. (Mörlenbach: SpringerWienNewYork, 2009).

Oore, Sageev, Ian Simon, Sander Dieleman, Douglas Eck and Karen Simonyan. “This Time with Feeling: Learning Expressive Musical Performance.” *arXiv.org.* (2018). Retrieved from https://arxiv.org/abs/1808.03715.

Pärt, Arvo. *Stabat Mater: für gemischten Chor (SAT) und Streichorchester.* (Wien: Universal Edition, 1985)

Payne, Christine. "MuseNet." *OpenAI.* (2019). Retrieved from openai.com/blog/musenet

Penton, Stephen Gregory John. “The compositional processes of Arvo pärt: a survey and comparison of two musical styles”. *Durham theses*, (Durham University: 1998). Retrieved from: http://etheses.dur.ac.uk/4816/

Radford, Alec, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei & Ilya Sutskever. “Language Models are Unsupervised Multitask Learners”. *openai.com,* (2019). Accessed July 16, 2020, https://cdn.openai.com/better-language-models/language_models_are_unsupervised_multitask_learners.pdf

Radford, Alec, Karthik Narasimhan, Tim Salimans & Ilya Sutskever. “Improving Language Understanding by Generative Pre-Training”. *openai.com,* (2018). Accessed June 3, 2020, https://s3-us-west-2.amazonaws.com/openai-assets/research-covers/language-unsupervised/language_understanding_paper.pdf

Raffel, Colin. "Learning-Based Methods for Comparing Sequences, with Applications to Audio-to-MIDI Alignment and Matching". PhD Thesis, (2016).

Roads, Curtis. *Composing Electronic Music: A New Aesthetic.* (Oxford: Oxford University Press, 2015).

Roads, Curtis. Microsound. (Cambridge, MA: The MIT Press, 2001).

Sandred, Örjan & Laurson, Mikael & Kuuskankare, Mika. (2009). Revisiting the Illiac Suite - A rule-based approach to stochastic processes. Sonic Ideas/Ideas Sonicas. 2. 42-46.

Schoenberg, Arnold, *String Quartet No. 2, Op. 10.* (Vienna: Universal Edition, 1921).

Schoenberg, Arnold, *Style and Idea: Selected Writings of Arnold Schoenberg: 60th Anniversary Ed.* (Berkeley: University of California Press, 2010).

Shanken, Edward A., editor. *Systems*. (Cambridge, MA: The MIT Press, 2015).

Shaw, Peter, Jakob Uszkoreit, & Ashish Vaswani. Self-Attention with Relative Position Representations. *arXiv.org.* (2018). Retrieved from http://arxiv.org/abs/1803.02155

Suchman, Lucy. *Human-Machine Reconfigurations: Plans and Situated Actions (2nd ed., Learning in Doing: Social, Cognitive and Computational Perspectives)*. (Cambridge: Cambridge University Press. 2006).

Tenney, Ian, James Wexler, Jasmijn Bastings, Tolga Bolukbasi, Andy Coenen, Sebastian Gehrmann, Ellen Jiang, Mahima Pushkarna, Carey Radebaugh, Emily Reif & Ann Yuan. “The Language Interpretability Tool: Extensible, Interactive Visualizations and Analysis for NLP Models,” *arXiv.org* (2020). Accessed September 22, 2020, https://arxiv.org/abs/2008.05122

Varga, Balint A. *Conversations with Iannis Xenakis*. (London, UK: Faber and Faber, 1996).

Vaswani, Ashish, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser & Illia Polosukhin. “Attention is All You Need”. *ArXiv.org*, (2017). Accessed July 3, 2020, https://arxiv.org/abs/1706.03762

Webber, Andrew Lloyd. *The Phantom of the Opera.* (London: The Really Useful Group Ltd., 1986)

Xenakis, Iannis. *Achorripsis.* (Berlin: Bote & Bock, 1958).

Xenakis, Iannis. *Formalized Music: Thought and Mathematics in Composition.* (Pentagon Press (Revised Edition), 1992).

Xenakis, Iannis. *Metastasis.* (London: Boosey & Hawkes, 1954).

Xenakis, Iannis. *Pithoprakta*. (London, UK: Boosey & Hawkes, 1967).

Curriculum Vitae

Name: Jeffrey A. T. Lupker

Post-secondary Education and Degrees:

The University of Western Ontario
London, Ontario, Canada
2010-2014 Hon.B.Mus.

The University of Western Ontario
London, Ontario, Canada
2014-2016 M.Mus.

The University of Western Ontario
London, Ontario, Canada
2016-2021 Ph.D.

Honours and Awards:

Dean’s Honour List
2011-2013

Paul Akira Ohashi Summit Award
2014

Western Graduate Research Scholarship
2015-2016, 2016-2020

Province of Ontario Graduate Scholarship
2015-2016 (Declined), 2019-2020

Social Science and Humanities Research Council (SSHRC)
Joseph-Armand Bombardier CGS-M
2015-2016

George Proctor Memorial Award
2020

Student Summer Teaching Fellowship Award
2020

Social Science and Humanities Research Council (SSHRC)
Doctoral Fellowship
2020-2021

Related Work Experience

Music Copyist
The University of Western Ontario, Musical Stage Productions
2016, 2017

Teaching Assistant
The University of Western Ontario
2014-2020

Research Assistant
The University of Western Ontario
2016, 2019, 2020

Publications:

“Two Movements for Orchestra” (Master's Thesis Composition), UWO Electronic Thesis and Dissertation Repository, 3727.

“Katajjaq Impressions.” Commissioned composition by The School of Graduate and Postdoctoral studies, UWO for "Celebrating 150 years of Canada and the World," premiered Nov 15, 2017 at The University of Western Ontario.

“Creating New Music with Big Data and Evolutionary Algorithms,” Abstract in *Digital Humanities 2020: Carrefours/Intersections*. University of Ottawa & Carleton University, July 22-24, 2020.

“Music Theory, the Missing Link Between Musical Big Data and Artificial Intelligence” with William J Turkel. *Digital Humanities Quarterly,* 15.1, 2021.

“Observing Mood-Based Patterns & Commonalities in Music using Machine Learning Algorithms” with William J. Turkel. In Justin Paterson, Rob Toulson and Russ Hepworth-Sawyer (eds.), *Innovation in Music.* (New York: Routledge, 2021).

“Algorithmic Composition and Generative Music.” In Philip Frana and Michael Klein (eds.), *Encyclopedia of Artificial Intelligence: The Past, Present, and Future of AI*. (Santa Barbara: ABC-CLIO, 2021).

“Deep Learning.” In Philip Frana and Michael Klein (eds.), *Encyclopedia of Artificial Intelligence: The Past, Present, and Future of AI*. (Santa Barbara: ABC-CLIO, 2021).

“Score-Transformer: A Deep Learning Aid for Music Composition.” In the *proceedings of the 21st New Interfaces for Musical Expression* (NIME, 2021).