

ESCOLA ESTADUAL PROFESSORA HILDA DE SOUZA FERREIRA

GABRIEL CARDOSO DOS SANTOS HERCULINO

DESENVOLVIMENTO DE UM SOFTWARE ESCRITOR DE PARTITURAS  
MUSICAIS MONOFÔNICAS

CAMPO GRANDE, MS

2019

ALEXANDRE BITTENCOURT JORDÃO FILHO  
GABRIEL CARDOSO DOS SANTOS HERCULINO

DESENVOLVIMENTO DE UM SOFTWARE ESCRITOR DE PARTITURAS  
MUSICAIS MONOFÔNICAS

Trabalho de Conclusão apresentado ao curso Técnico em Informática, da Escola Estadual Professora Hilda de Souza Ferreira, como requisito à obtenção do certificado de Técnico em Informática.

Orientador: Prof. Lucas Akayama Vilhagra

CAMPO GRANDE, MS

2019

**FOLHA DE APROVAÇÃO**

ALEXANDRE BITTENCOURT JORDÃO FILHO  
GABRIEL CARDOSO DOS SANTOS HERCULINO

DESENVOLVIMENTO DE UM SOFTWARE ESCRITOR DE PARTITURAS MUSICAIS  
MONOFÔNICAS

Trabalho de Conclusão aprovado como requisito para obtenção do certificado de conclusão do curso Técnico em Informática, da Escola Estadual Professora Hilda de Souza Ferreira, pela seguinte banca examinadora:

---

Prof. Lucas Akayama Vilhagra  
Orientador - Engenheiro da Computação - UFMS

---

Prof.

---

Prof.

---

Prof.

Campo Grande, abril de 2019.

Dedicamos este trabalho a todos os colegas, professores e familiares que contribuíram para nossa formação. Sem eles, e os conhecimentos obtidos durante o curso, o mesmo nunca teria sido possível.

## **AGRADECIMENTOS**

Primeiramente agradecemos a Deus pela oportunidade de estarmos aprendendo e nos capacitando com um curso técnico, em uma escola com excelentes profissionais, e ainda por ter nos dado saúde e força para superar todas as dificuldades encontradas durante ele, inclusive a realização deste trabalho.

A nossa família por ter nos apoiado e sofrido juntamente conosco todas as dificuldades passadas, sempre fazendo o melhor para nos ajudar e nos incentivar a dar o nosso melhor.

A nossos professores e coordenadores (que se tornaram muito mais do que isso, são nossos amigos e companheiros) por cada conhecimento transmitido, não somente sobre informática, mas conhecimentos que poderão ser levados para o resto de nossas vidas. E ainda, pelos momentos de dificuldade em que eles nos auxiliaram, muitas vezes em finais de semana e durante a madrugada.

Aos nossos colegas que viveram com nós essa experiência, muitas vezes perdidos juntamente conosco mas sempre tentando nos ajudar. Infelizmente, não foram muitos que conseguiram ir até o final, mas acreditamos que esse curso e essa turma ficarão marcados na vida de cada um.

Aos nossos amigos, que também nos apoiaram e muitas vezes nos distraíram quando a cabeça parecia que iria explodir. Ainda mais àqueles que muitas vezes sem entender nos ouviam explicar detalhes, problemas e soluções para nosso trabalho, e até mesmo nos dando dicas e sugestões de como aperfeiçoá-lo.

A todos os autores que disponibilizam o conteúdo utilizado, de fato, poucos se voluntariam a transmitir tais conhecimentos. Que eles possam ser recompensados por todo o esforço e dedicação a cada uma das obras, que na maioria das vezes não têm fins lucrativos.

Por fim, a todos que contribuíram com nosso processo de aprendizagem, que nós possamos retribuir isso tudo a cada um de vocês futuramente, muito obrigado!

“Qualquer tecnologia suficientemente  
avançada é equivalente à mágica”.

(Arthur Charles Clarke)

## RESUMO

Uma forma de representação gráfica para as partes constituintes de uma música é a partitura. A partitura representa o registro de como uma música deve ser executada por um músico e é uma parte importante na documentação de composições, utilizada por pequenas bandas musicais e indispensável para orquestras conhecidas mundialmente, que buscam atingir a perfeição em suas execuções. Apesar disso, a escrita de partituras é um processo complexo e estirado, principalmente devido o conhecimento necessário para que seja realizado, o que resulta em um pequeno acervo disponibilizado a quem busca um auxílio durante as etapas de aprendizagem e execução de uma música. Neste contexto, este trabalho propõe a criação de um software automatizador de escrita de partituras musicais monofônicas que seja eficiente e preciso, de forma que ele passe a ser utilizado e se torne indispensável na vida dos músicos.

Palavras-chave: Partitura. Música. Composições. Software.

## **ABSTRACT**

One form of graphic representation for the constituent parts of a song is the score. The score represents the record of how a song is to be performed by a musician and is an important part in the documentation of compositions, used by small musical bands and indispensable for world-known orchestras, who seek to achieve perfection in their performances. In spite of this, the writing of scores is a complex and drawn process, mainly due to the knowledge necessary for it to be performed, which results in a small collection available to those who seek help during the stages of learning and performing a song. In this context, this work proposes the creation of an automation software for the writing of monophonic musical scores that is efficient and precise, so that it becomes used and indispensable in the life of the musicians.

Key-words: Sheet music. Music. Compositions. Software.



## LISTA DE FIGURAS

FIGURA 1 - SINAL ANALÓGICO DE TEMPO CONTÍNUO.....	13
FIGURA 2 - SINAL DIGITAL DE TEMPO DISCRETO.....	13
FIGURA 3 - SINAL ANALÓGICO DISCRETO NO TEMPO.....	14
FIGURA 4 - SINAL DIGITAL CONTÍNUO NO TEMPO.....	14
FIGURA 5 - SINAL A SER AMOSTRADO.....	16
FIGURA 6 - SUPERPOSIÇÃO ENCONTRADA EM SINAL AMOSTRADO.....	16
FIGURA 7 - AMOSTRAGEM COM FREQUÊNCIA ADEQUADA.....	17
FIGURA 8 - AMOSTRAGEM COM FREQUÊNCIA BAIXA.....	17
FIGURA 9 - “FAMÍLIA” FOURIER.....	19
FIGURA 10 - FREQUÊNCIAS DAS NOTAS DO TECLADO .....	22
FIGURA 11 - CLAVES UTILIZADAS PARA ESCRITA ATUALMENTE.....	23
FIGURA 12 - POSICIONAMENTO DAS NOTAS EM CADA CLAVE.....	23
FIGURA 13 - TRANSIÇÃO DA CLAVE DE FÁ PARA CLAVE DE SOL.....	24
FIGURA 14 - ATRIBUTOS DAS FIGURAS MUSICAIS.....	24
FIGURA 15 - EXEMPLOS DE FÓRMULAS DE COMPASSO.....	25
FIGURA 16 - EXECUÇÃO DO SOFTWARE NO TERMINAL DO UBUNTU.....	28

## SUMÁRIO

<b>1 INTRODUÇÃO</b>	<b>11</b>
<b>2 FUNDAMENTAÇÃO</b>	<b>13</b>
2.1 PROCESSAMENTO DE SINAIS	13
2.1.1 Definição de sinal	13
2.1.2 Sinais analógicos e digitais	14
2.1.3 Sinais de tempo contínuo e tempo discreto	15
2.1.4 Sinais periódicos e aperiódicos	16
2.1.5 Amostragem	16
2.1.6 Teorema da amostragem	18
2.2 ANÁLISE DE FOURIER	19
2.2.1 Série de Fourier	20
2.2.2 Transformada de Fourier	20
2.3 TEORIA MUSICAL	22
2.3.1 Notas e frequências	23
2.3.2 Claves	24
2.3.3 Figuras e seus valores	25
2.3.4 Compassos	26
<b>3 DESENVOLVIMENTO</b>	<b>27</b>
3.1 PROCESSAMENTO DIGITAL DE SINAIS	27
3.1.1 Pacote SciPy	28
3.1.2 Processamento digital de sinais em Python 3	28
3.1.3 Armazenamento de frequências em arquivo CSV	30
3.2 RECONHECIMENTO DE NOTAS MUSICAIS	30
3.2.1. Leitura das frequências armazenadas em arquivo CSV	30
3.2.2. Análise de frequências obtidas	32
<b>4 RESULTADOS</b>	<b>33</b>
<b>5 CONSIDERAÇÕES FINAIS</b>	<b>34</b>
<b>6 TRABALHOS FUTUROS</b>	<b>35</b>
6.1 INTERFACE GRÁFICA	35
6.2 CONFIGURAÇÃO DO SOFTWARE	35
6.3 ESCRITA INSTANTÂNEA DE PARTITURAS	36
<b>7 REFERÊNCIAS BIBLIOGRÁFICAS</b>	<b>37</b>

## 1 INTRODUÇÃO

A partitura musical é uma forma de representar graficamente todas as partes constituintes de uma música (melodia, harmonia e ritmo). Sendo assim, ao ser interpretada, ela indica ao executante as notas a ser emitidas em cada momento, a duração de cada uma delas e possíveis alterações sonoras exigidas em determinados trechos.

Com uma partitura a execução de qualquer música é facilitada, até mesmo quando se tratam de melodias desconhecidas pelo instrumentista, auxiliando iniciantes que ainda não têm uma boa percepção musical e aprimorando a execução em conjunto, que não é muito precisa quando realizada sem ela.

Neste contexto, músicos e cientistas buscam uma forma eficiente de automatizar o processo de transcrição de sons em partituras. Apesar de existirem algumas ferramentas que apresentam tal proposta, nenhuma delas têm desempenho suficiente a ponto de se tornarem populares no ramo, ou serem usadas regularmente, como já acontece com programas de escrita e edição de partituras e afinadores digitais.

A fim de preencher esta lacuna, neste trabalho é proposto o desenvolvimento de um software automatizador de escrita de partituras musicais monofônicas, ou seja, que apresentam apenas uma melodia. Espera-se que o *software* obtenha desempenho satisfatório para a aplicação desejada, pois o intuito é torná-lo funcional para que possa ser uma ferramenta utilizada por músicos profissionais e amadores.



## **2 FUNDAMENTAÇÃO TEÓRICA**

### **2.1 PROCESSAMENTO DE SINAIS**

Os conceitos de sinais e sistemas desempenham um papel importante em diversas áreas da ciência e tecnologia, como comunicações, projeto de circuitos, sistemas de geração e distribuição de energia, controle de processos químicos, processamento de voz, etc [6]. Logo, os sistemas precisam realizar o processamento dos sinais de diversas formas, variando conforme o objetivo de cada um deles.

O processamento de sinais consiste em analisar o comportamento dos sinais e extrair ou manipular as informações contidas neles, de modo que eles se tornem mais apropriados para uma determinada aplicação, buscando obter um resultado desejado [1].

#### **2.1.1 Definição de sinal**

Os sinais são elementos básicos de nossa vida diária. A forma comum de comunicação humana, por exemplo, se desenvolve por meio do uso de sinais da fala, seja pessoalmente ou por um canal telefônico [2].

Um sinal pode ser definido como um conjunto de dados que pode ser representado por uma função de uma ou mais variáveis. Enquanto dependente de apenas uma variável, um sinal é chamado unidimensional, passando a ser multidimensional quando dependente de duas ou mais variáveis [3].

Exemplos compreensíveis presentes no cotidiano são os sinais de telefone, televisão, internet etc. Todos esses citados podem ser representados com funções de uma variável independente tempo, o que nem sempre é o caso. Além de estarem em função do tempo os sinais podem ainda estar em função do espaço ou até mesmo de uma frequência, por exemplo [3].

Existem diversas formas de classificar sinais, conforme todos os atributos dos mesmos. Neste trabalho serão abordadas apenas classificações estudadas e utilizadas durante o processo de desenvolvimento do software.

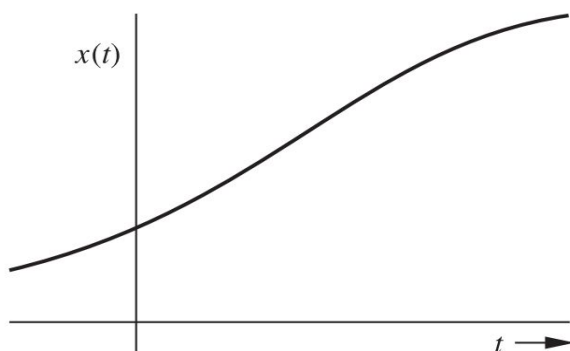
### 2.1.2 Sinais analógicos e digitais

Com o advento dos circuitos integrados e os microprocessadores, tornou-se possível a digitalização dos sinais do mundo real, que são inerentemente analógicos. Desta forma, é possível que o processamento seja realizado de maneira digital. Uma das vantagens desse processo é que o sinal digital não sofre com as variações que os sinais analógicos estão sujeitos. Além disso, com o sinal digitalizado, há a possibilidade de realizar a análise por meio de algoritmos [4].

Segundo Lathi, em seu livro Sinais e Sistemas Lineares, um sinal analógico é um sinal contínuo cuja amplitude pode assumir infinitos valores no decorrer da variável independente. Um sinal digital, por outro lado, é aquele cuja amplitude pode assumir apenas um conjunto finito de valores.

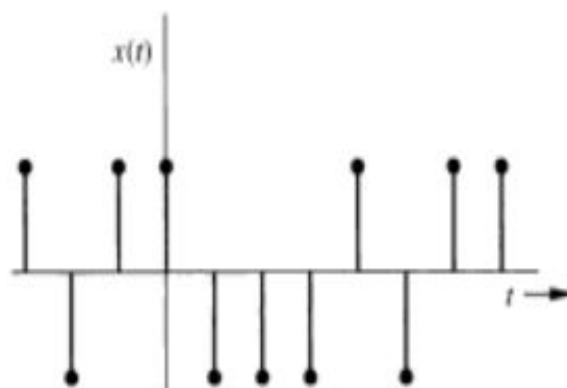
Sendo assim, para efetuar a conversão de um sinal analógico, a digitalização é feita por meio de uma quantização (arredondamento) dos valores de sua amplitude [3].

FIGURA 1 - SINAL ANALÓGICO DE TEMPO CONTÍNUO.



FONTE: Lathi (1979, p. 88) [3].

FIGURA 2 - SINAL DIGITAL DE TEMPO DISCRETO.



FONTE: Lathi (1979, p. 88) [3].

### 2.1.3 Sinais de tempo contínuo e tempo discreto

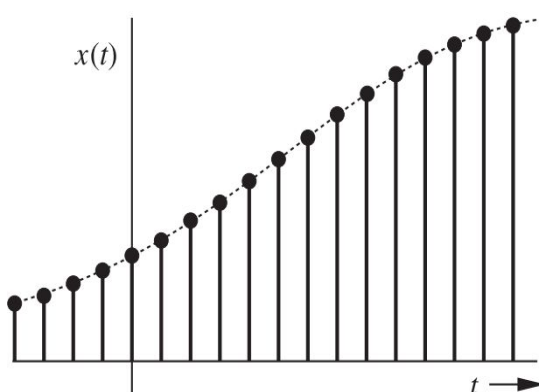
Os conceitos de sinais de tempo contínuo e de tempo discreto muitas vezes são confundidos com os conceitos de sinais analógicos e digitais [3]. Isso ocorre porque são definições de compreensão não intuitiva e que exigem uma base de conhecimento teórico em processamento de sinais.

Sinais de tempo contínuo são aqueles em que a variável independente é contínua no tempo, conseqüentemente, esses sinais são definidos em um conjunto contínuo (sem pausas) de valores. Em contrapartida, sinais de tempo discreto são definidos somente em determinados instantes, ou seja, assumem apenas um conjunto discreto de valores da variável independente [6].

Portanto, enquanto os conceitos de sinal analógico e digital são caracterizados por sua amplitude (eixo vertical), podendo assumir infinitos valores, os conceitos de sinal de tempo contínuo e tempo discreto são caracterizados ao longo de seu tempo (eixo horizontal) [3].

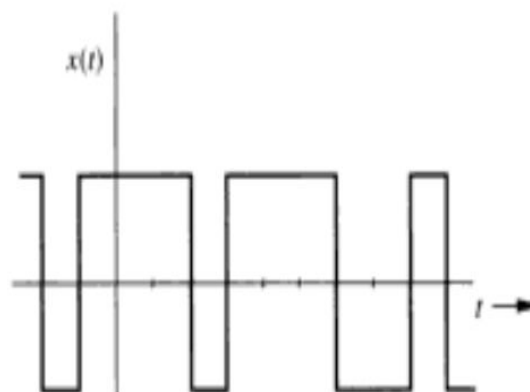
Apesar dessa divergência, um sinal analógico pode ser classificado como contínuo ou como discreto no tempo. O mesmo vale para um sinal digital [2].

FIGURA 3 - SINAL ANALÓGICO  
DISCRETO NO TEMPO



FONTE: Lathi (1979, p. 88) [3].

FIGURA 4 - SINAL DIGITAL  
CONTÍNUO NO TEMPO



FONTE: Lathi (1979, p. 88) [3].

#### 2.1.4 Sinais periódicos e aperiódicos

A periodicidade é uma propriedade que pode estar presente tanto em sinais analógicos quanto em digitais. Esta é uma característica importante a ser identificada em um sinal, pois, a partir dela serão definidas as técnicas necessárias para tratar e processar os sinais. Portanto, saber se os sinais analisados são periódicos influencia significativamente o projeto [5].

Um sinal é periódico quando, com o decorrer do tempo, ele apresenta os mesmos valores em sua amplitude a cada período de tempo  $T$  [6], ou seja, ao encontrar o intervalo onde esses valores passam a se repetir, conhece-se o comportamento deste sinal para todos os outros intervalos de tempo.

A importância de reconhecer um sinal periódico se deve ao fato do mesmo não se modificar durante o deslocamento temporal e, conseqüentemente, poder ser reconstruído com mais facilidade, isso porque ele pode ser representado através de funções periódicas [3]. Isso não significa que um sinal aperiódico não possa ser representado por meio de funções, entretanto, como o sinal se modifica durante o deslocamento temporal, a tarefa se torna mais complexa.

#### 2.1.5 Amostragem

Como visto anteriormente, para converter um sinal analógico em um sinal digital é utilizada uma quantização dos valores de sua amplitude [3]. Essa quantização ocorre juntamente com uma discretização da variável independente em função do tempo, concretizando o processo de amostragem do sinal.

Essa amostragem é realizada pois o processador é incapaz de lidar com números não inteiros (mesmo os números em ponto flutuante tem sua representação inteira), logo, o mesmo não pode convenientemente manipular um sinal analógico, cuja amplitude pode assumir infinitos valores [5].

Portanto, o processo de amostragem transforma um sinal em digital (quantizado na amplitude) e discreto no tempo (apresenta valores apenas em determinados instantes no decorrer do tempo).



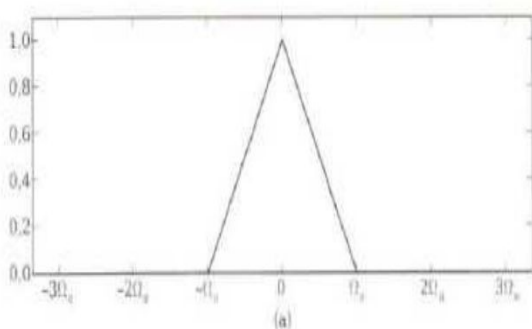
Sendo assim, valores da amplitude do sinal são obtidos e armazenados em um dispositivo de memória a cada intervalo de tempo, denominado período de amostragem, o que o torna passível de processamento [5].

Com isso, durante o processamento de um sinal é feita a amostragem por um conversor analógico-digital (A/D), que irá convertê-los em sinais de tempo discreto para serem manipulados e processados. Durante essa etapa de processamento, o sinal passa ainda por uma filtragem digital, que é programada diretamente no processador a fim de eliminar distorções nos sinais.

Posteriormente, esses sinais serão convertidos novamente, por um conversor digital-analógico (D/A), em um sinal de tempo contínuo, buscando retornar algo muito parecido com o que entrou inicialmente no sistema [6].

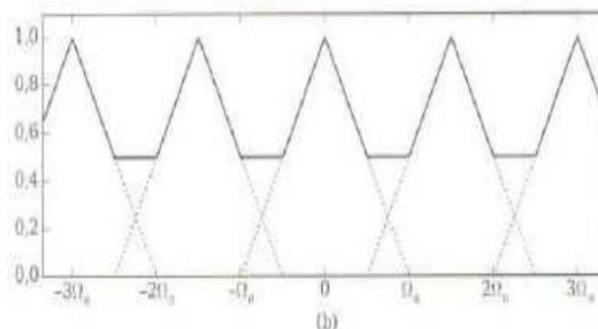
Caso o sinal que estiver sendo amostrado não for limitado e ele contiver energia acima de uma determinada frequência, ocorrerá uma superposição entre os espectros replicados, ou seja, em certos instantes a amplitude do sinal receberá mais de um valor, causando distorções na amostragem, que também são chamadas de *aliasing* [5].

FIGURA 5 - SINAL A SER AMOSTRADO.



FONTE: Haykin(1979, p. 88) [3].

FIGURA 6 - SUPERPOSIÇÃO ENCONTRADA EM SINAL AMOSTRADO.



FONTE: Lathi (1979, p. 88) [3].

### 2.1.6 Teorema da amostragem

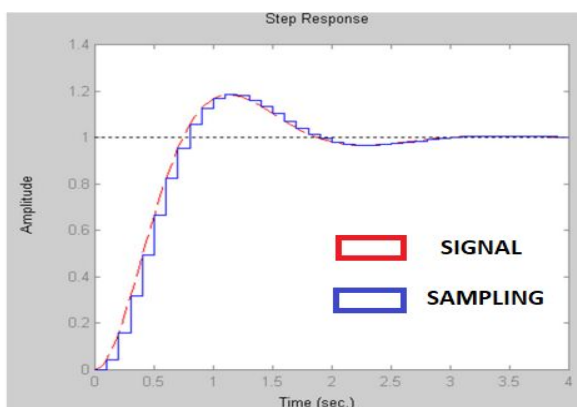
As distorções que ocorrem durante o processo de amostragem podem ser evitadas se o sinal a ser amostrado for limitado em frequência [5]. Essa condição é conhecida como taxa de amostragem, proposta pelo Teorema de Nyquist (mais conhecido como Teorema da Amostragem), que é visto por Lathi como a ponte entre os mundos de tempo contínuo e tempo discreto [3].

Segundo esse teorema, com a frequência de um sinal a ser amostrado sendo limitada em um determinado valor, e a frequência de amostragem sendo colocada como o dobro desse limite, é possível reconstruir totalmente esse sinal sem nenhuma distorção [6].

No entanto, o Teorema da Amostragem utiliza-se de uma idealização, visto que não é possível limitar perfeitamente um sinal em uma frequência desejada. Logo, por mais aproximado que os valores estejam da condição proposta, eles ainda não a satisfazem, portanto, sempre haverá uma pequena distorção nos sinais.

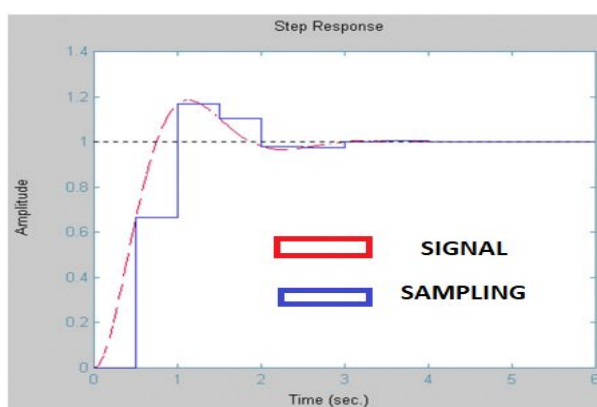
Apesar disso, quando se trata de valores reais, é obtida uma aproximação suficientemente boa em relação à situação proposta pela taxa de amostragem de Nyquist. Com isso, a quantidade de *aliasing* encontrada no sinal amostrado vai depender das frequências utilizadas durante o processo [5].

7 - AMOSTRAGEM COM FREQUÊNCIA ADEQUADA.



FONTE: Weeks (2007) [7].

FIGURA 8 - AMOSTRAGEM COM FREQUÊNCIA BAIXA.



FONTE: Weeks (2007) [7].

Contudo, ainda que ocorram eventuais distorções no resultado, os conceitos propostos pelo teorema da amostragem são vastamente utilizados atualmente durante o processo de desenvolvimento de sistemas. Isso ocorre porque muitas das vezes quando um sinal é amostrado em uma frequência próxima à ideal, as distorções encontradas não são perceptíveis aos sentidos humanos.

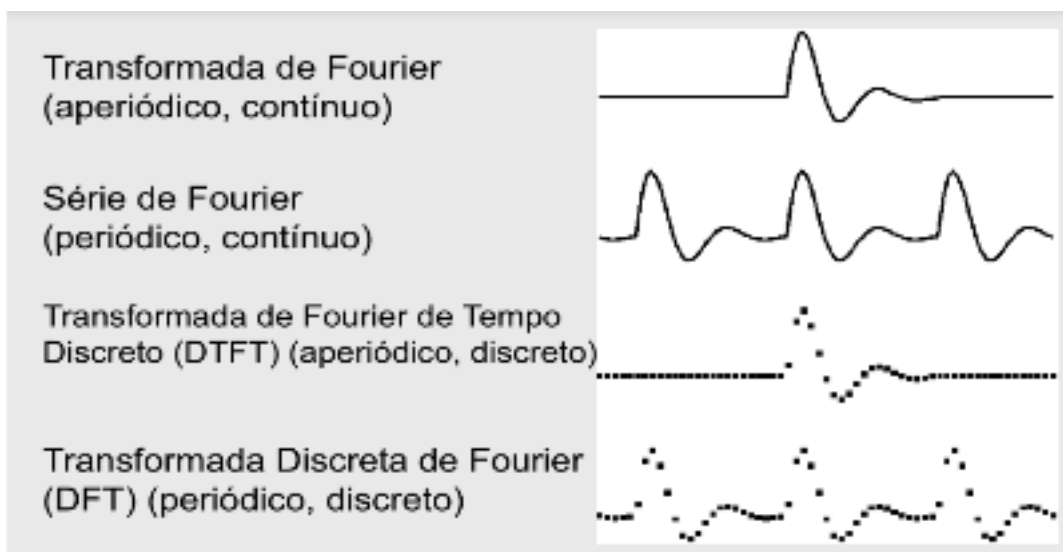
## 2.2 ANÁLISE DE FOURIER

Jean Baptiste Joseph Fourier (1768 - 1830) foi um famoso físico e matemático que contribuiu com diversas áreas de atuação apresentando cálculos teorias [8]. Os estudos, séries e transformadas que levam seu nome foram, e ainda são, amplamente utilizados, inclusive durante o processamento digital de sinais.

O estudo de sinais e sistemas utilizando representações senoidais é denominado Análise de Fourier em homenagem a Joseph Fourier. Os métodos de Fourier são utilizados em diversos ramos da engenharia e da ciência [2]. O desenvolvimento da análise de Fourier envolve diversos estudiosos, além de Joseph Fourier, e a investigação de fenômenos físicos que contribuíram para chegar aos conhecimentos de aplicação encontrados atualmente [6].

Existem diversos cálculos propostos para a representação de sinais, como a transformada de Laplace, porém, a Análise de Fourier é o método mais recorrido durante esse processo, afinal, por meio dela é possível representar sinais de todas as classificações descritas anteriormente [9].

FIGURA 9 - "FAMÍLIA FOURIER"



FONTE: Professor Doutor Carlos Alberto Ynoguti [9]

### 2.2.1 Série de Fourier

Por volta de 1807, Joseph Fourier completou uma obra onde ele concluiu que séries senoidais harmonicamente relacionadas eram úteis na representação da distribuição de temperatura em um corpo. A partir da observação dessa obra ele passou a afirmar que qualquer sinal periódico poderia ser representado a partir de séries senoidais [6].

Entretanto, as afirmações e os argumentos matemáticos dados por Fourier eram imprecisos para comprovar a teoria. Ainda assim, foram importantes pois estimularam Dirichlet, que em 1829 forneceu condições precisas sob as quais um sinal periódico pode ser representado pela atualmente chamada Série de Fourier (FS - Fourier Serie) [6].

Apesar de não ser efetivada por Fourier, a série recebeu esse nome devido a sua influência e a continuidade dada por ele aos estudos na área de processamento de sinais, afinal, além da Série de Fourier existem as outras ferramentas compositoras da Análise de Fourier.

### 2.2.2 Transformada de Fourier

Após a efetivação da Série de Fourier, sinais periódicos passaram a ser representados com séries harmonicamente relacionadas. No entanto, existiam os

sinais que não poderiam ser representados através desse método, os sinais aperiódicos (que não possuem período fundamental) [6].

A partir disso foi desenvolvida uma das contribuições mais importantes de Fourier. Vista como a extensão da Série de Fourier, a Transformada de Fourier (FT - Fourier Transform) analisa o domínio da frequência de um sinal, tornando possível a representação de sinais aperiódicos [5].

A Transformada de Fourier torna uma função temporal em frequências presentes nela originalmente, utilizando uma superposição de senóides complexas. Realizando essa decomposição de senóides componentes em um sinal, permite-se modificar o domínio de representação do mesmo, passando do domínio do tempo para o domínio da frequência [3].

A partir disso, observa-se a importância da Transformada de Fourier quando se trata de instrumentos musicais, afinal, as notas musicais são sinais analógicos contínuos no tempo. Sendo assim, para que sons e notas musicais sejam processados digitalmente se faz necessária a realização da Transformada de Fourier.

Além da Transformada de Fourier que trabalha com sinais de tempo contínuo, existem ainda suas variações que tratam sinais de tempo discreto, como é o caso da Transformada Discreta de Fourier (DFT - Discrete Fourier Transform) que trabalha com sinais de tempo discreto e periódicos e da Transformada de Fourier de Tempo Discreto (DTFT - Discrete-Time Fourier Transform) que trabalha com sinais discretos no tempo e aperiódicos.

## 2.3 TEORIA MUSICAL

A teoria musical é um conjunto de estudos ou regras que analisam e classificam os elementos da música [10]. Estudar teoria musical é, muita vezes, considerado pelos músicos, como o processo de entendimento do funcionamento das partituras.

A partitura é um elemento muito importante utilizado para compor, entender e executar uma música. Ela em si, constitui-se de uma notação impressa ou manuscrita com todos os elementos presentes na música tocada.

Muitas das músicas famosas são escritas em partituras, facilitando, de certa forma, a popularização delas entre os músicos. Um exemplo disso são as músicas clássicas que ainda hoje são executadas apenas com a observação da partitura das mesmas.

Em uma escrita de partitura são encontrados vários elementos, em conjunto, permitindo ela cumprir seu objetivo de representar graficamente uma música entoada, fazendo com que ela possa ser executada perfeitamente, diminuindo a probabilidade de erros durante a execução.

O pentagrama, também chamado de pauta, é o primeiro desses elementos que compõem a partitura, podendo ser rapidamente notado ao observá-la. Trata-se de um conjunto de 5 linhas em que toda a representação gráfica das informações da música será escrita.

Além das linhas do pentagrama, existem ainda linhas suplementares que podem ser inferiores ou superiores a ele. É importante destacar que quanto mais abaixo no pentagrama, ou fora dele, mais grave (baixa frequência) a nota será, sendo assim, quanto mais se sobe no pentagrama, ou acima dele, mais aguda (alta frequência) será a nota representada [11].

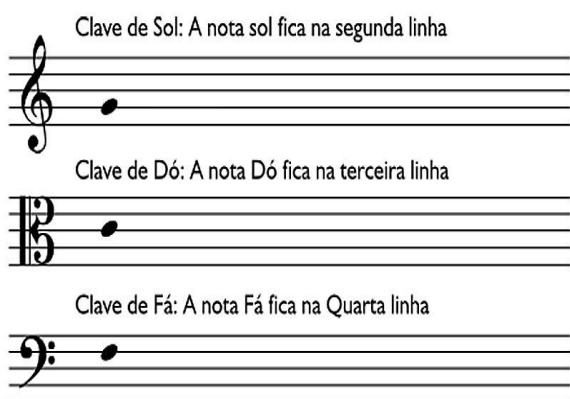


### 2.3.2 Claves

Cada linha e espaço do pentagrama representa uma nota musical que pode variar sua posição conforme a clave definida para aquela partitura. Portanto, a clave é um sinal colocado no início de cada pauta que pega uma nota como referência para definir todas as outras [11]. Obviamente, o nome da clave é dado conforme a nota tomada por referência.

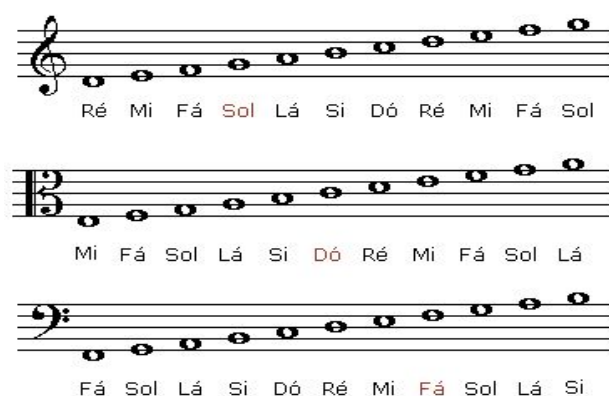
Atualmente usam-se 3 tipos de claves: de Sol, de Fá e de Dó [11]. Portanto, em cada uma delas, as notas musicais serão posicionadas de diferente modo como é possível notar na figura 12.

FIGURA 11 - CLAVES UTILIZADAS PARA ESCRITA ATUALMENTE.



FONTE: Blogs Canção Nova [13].

FIGURA 12 - POSICIONAMENTO DAS NOTAS EM CADA CLAVE.



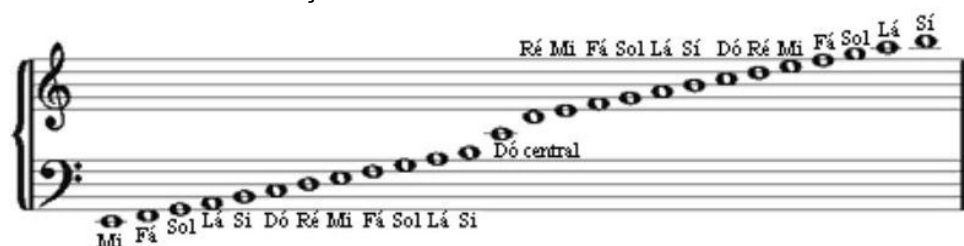
FONTE: Coral Accordis [14].

A clave de Dó é pouco utilizada atualmente, um dos poucos instrumentos que a utilizam é a Viola (instrumento de cordas). No entanto, quando se tratam de instrumentos de sopro, nenhum utiliza essa clave por completo, ela é encontrada apenas em alguns trechos isolados de partituras de trombone (instrumento de sopro). Sendo assim, a grande maioria das partituras musicais se encontram na clave de Fá e na clave de Sol.

Geralmente a clave de Fá é utilizada para instrumentos que emitem notas mais graves, enquanto a clave de Sol é utilizada para instrumentos que emitem notas mais agudas. Com isso, observa-se que a clave de sol é uma complemento da clave de Fá.



FIGURA 13 - TRANSIÇÃO DA CLAVE DE FÁ PARA A CLAVE DE SOL



FONTE: Música e Adoração [15];








### 2.3.3 Figuras e seus valores

Além da altura e, consequentemente a frequência, uma partitura indica também a duração de tempo de execução de cada uma das notas que serão emitidas e as pausas existentes entre elas, possibilitando assim a completa representação gráfica de uma melodia.

Para que isso seja possível existe um conjunto de sinais convencionais representativos das durações. As Figuras musicais representam a duração de tempo que cada uma das notas deverão ser mantidas, enquanto isso, as Pausas representam o tempo de silêncio existente entre elas. Apesar disso, para cada Figura Musical existe uma pausa correspondente [11].

Atualmente são comumente utilizadas 7 tipos de figuras e pausas, com elas é possível representar graficamente qualquer música. Além desses, existem outros tipos que são chamados de Valores Extremos, mas que caíram em desuso e não são utilizados hodiernamente.

FIGURA 14 - ATRIBUTOS DAS FIGURAS MUSICAIS

NÚMERO	FIGURA	NOME	TEMPO
1		SEMIBREVE	4
2		MÍNIMA	2
4		SEMÍNIMA	1
8		COLCHEIA	$\frac{1}{2}$
16		SEMICOLCHEIA	$\frac{1}{4}$
32		FUSA	$\frac{1}{8}$
64		SEMIFUSA	$\frac{1}{16}$

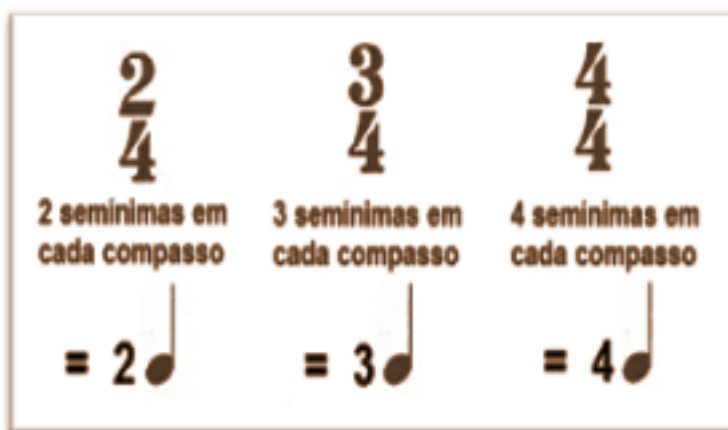
FONTE: Bohumil Med (1996) [11].

### 2.3.4 Compassos

Os compassos são elementos encontrados no pentagrama separados por um segmento vertical, chamado barra de compasso ou travessão. Dentro de cada um deles existe um limite de tempos a ser atingido [11].

Eles são determinantes de tempo da música que dividem-na em séries regulares, ditando o tempo em que as notas são emitidas e constituindo portanto o ritmo da música. Para isso, cada figura musical tem um valor empregado e cada compasso recebe uma determinada quantidade de notas, que vai variar conforme a fórmula de compasso empregada na partitura.

FIGURA 15 - EXEMPLOS DE FÓRMULAS DE COMPASSO



FONTE: Teoria Descomplicada [16]

### 3 DESENVOLVIMENTO

Ao ser iniciada a etapa de desenvolvimento, foram observados e listados os obstáculos relacionados ao objetivo do software em extrair e manipular sinais encontrados nos sons emitidos por um instrumento musical.

Inicialmente a proposta era realizar o processamento de sinais diretamente do som que era capturado pelo microfone do computador, todavia, devido às dificuldades encontradas em tal procedimento, e o interesse em fazer com que não necessariamente a gravação fosse feita em um computador, o software foi desenvolvido para analisar sons armazenados em um arquivo de áudio.

Com o que foi estudado, é possível notar que os sons emitidos por um instrumento musical são sinais analógicos que precisam ser digitalizados antes de armazenados. No entanto, essa tarefa não foi desempenhada pelo software desenvolvido, afinal, ele reconhece, extrai e manipula as informações de um arquivo já digitalizado.

Ainda assim, o áudio que será utilizado necessita de um processamento digital, afinal, a partir do sinal contido nele, outras informações serão obtidas como, por exemplo, a frequência do som emitido pelo instrumento e a nota musical que essa frequência representa.

#### 3.1 PROCESSAMENTO DIGITAL DE SINAIS

Durante essa etapa do desenvolvimento foi utilizada a linguagem de programação Python, principalmente devido a quantidade de pacotes relacionados a processamento de sinais existentes, e as facilidades oferecidas pela linguagem, que cada vez mais tem se destacado em tarefas desta natureza.

Por isso, a implementação dos processos descritos na fundamentação teórica foram facilitados, afinal, é possível encontrar pacotes com classes já prontas que os realizam, sendo necessários apenas alguns ajustes conforme a necessidade do desenvolvedor.

### 3.1.1 Pacote SciPy

O pacote mais utilizado e que mais se destaca é o SciPy, que foi o principal motivo pela escolha da linguagem. Trata-se de um pacote exclusivamente desenvolvida para Python que implementa diversas técnicas úteis na computação científica [17]. Ele permitiu e facilitou o desenvolvimento de boa parte dos processos realizados pelo software.

A primeira das funções é o reconhecimento do áudio inserido como um arquivo WAV (formato de arquivo de áudio comumente utilizado em computadores), que a princípio é o único formato aceito pelo software, pelo fato de ser um formato que oferece uma boa qualidade de áudio e o reproduz da melhor maneira possível, afinal é um formato que não permite a compressão do arquivo.

Além do reconhecimento de formato, o SciPy vai ficar responsável por atribuir informações do áudio, como o nome e a duração do arquivo, e também todas as informações necessárias para que o processamento do mesmo seja possível, como o número total de amostras realizadas, a frequência de amostragem, o período de amostragem, o armazenamento do sinal amostrado e dos valores obtidos da amplitude do mesmo em certos instantes de tempo conforme o período de amostragem.

Não obstante a todas essas funções oferecidas, também é possível realizar a transformada de Fourier com o que é oferecido pelo pacote, o que torna seu uso ainda mais indispensável durante o processo de desenvolvimento, sem o mesmo não só essa como todas as demais tarefas desempenhadas por ele seriam muito mais trabalhosas e complexas de serem implementadas.

### 3.1.2 Processamento digital de sinais em Python 3

Toda a parte de processamento digital existente no software, foi dividida em 4 arquivos Python. O primeiro deles (Audio.py) contém a classe que, utilizando o pacote SciPy, vai reconhecer o áudio como arquivo WAV, coletar as informações dele e realizar a amostragem do sinal obtido.

Já o segundo arquivo (FrequencyAnalysis.py) contém as classes que realizam a FFT (Fast Fourier Transform) e define a maior frequência analisada no sinal, que será comparada e classificada conforme a frequência das notas musicais e suas respectivas oitavas.

O terceiro arquivo (Main.py) é o responsável por carregar o áudio inserido e implementá-lo em todas as classes criadas. Além disso, ele armazenará as frequências obtidas em uma lista que será enviada para o último arquivo (WriteFrequency.py) responsável por armazenar os valores que serão utilizados na próxima etapa do desenvolvimento.

Para que a Main seja executada no terminal do Ubuntu (distribuição linux utilizada durante o desenvolvimento) ou em outras distribuições semelhantes, basta executá-la com Python 3 passando como parâmetros o caminho do áudio que será inserido para análise e o valor da janela de recorte que será utilizada.

FIGURA 16 - EXECUÇÃO DO SOFTWARE NO TERMINAL DO UBUNTU

```

gabriel@gabriel-VirtualBox: ~/Desktop/TCC
File Edit View Search Terminal Help
gabriel@gabriel-VirtualBox:~/Desktop/TCC$ python3 Main.py cello.wav 1
311.0 Hz
311.0 Hz
311.0 Hz
gabriel@gabriel-VirtualBox:~/Desktop/TCC$ javac Main.java
gabriel@gabriel-VirtualBox:~/Desktop/TCC$ java Main
Ré#3
Ré#3
Ré#3
gabriel@gabriel-VirtualBox:~/Desktop/TCC$

```

Annotations in the image:

- EXECUÇÃO DOS ARQUIVOS PYTHON**: Points to the command `python3 Main.py cello.wav 1`.
- FREQUÊNCIAS OBTIDAS A CADA SEGUNDO**: Points to the output `311.0 Hz`.
- COMPILAÇÃO DA MAIN EM JAVA**: Points to the command `javac Main.java`.
- NOTAS MUSICAIS ENCONTRADAS**: Points to the output `Ré#3`.
- EXECUÇÃO DA MAIN**: Points to the command `java Main`.

FONTE: Própria (2019).

### 3.1.3 Armazenamento de frequências em arquivo CSV

Comma Separated Values (CSV) é um formato de arquivo geralmente utilizado em trocas de dados entre planilhas. Cada linha em um arquivo CSV representa uma linha em uma planilha e os valores que são delimitados, normalmente entre vírgulas, representam uma célula [18].

Como dito anteriormente, o arquivo WriteFrequency.py recebe uma lista com todas as frequências do sinal obtidas durante a análise do áudio. Após receber essa lista, a classe criará um arquivo no formato CSV para armazenar todos os valores, a fim de que eles possam, posteriormente, ser lidos por outra linguagem de programação.

Portanto, a finalidade de utilização do formato CSV não é inserir as frequências do sinal em uma planilha, por mais que isso seja uma consequência de sua utilização, mas sim permitir que os valores obtidos durante o processamento em Python sejam lidos e tratados por outra classe em Java.

## 3.2 RECONHECIMENTO DE NOTAS MUSICAIS

Com os valores das frequências encontradas armazenados em um arquivo CSV, encerra-se a etapa de processamento do sinal, afinal, já foram extraídas e encontradas todas as informações necessárias para o funcionamento do programa. Com isso, o próximo passo a ser dado é um processo de reconhecimento de notas musicais conforme cada uma das frequências obtidas.

Antes disso, porém, é necessário lembrar que todo o processamento foi realizado em Python e essa próxima etapa foi desenvolvida em Java. Para a comunicação entre os módulos desenvolvidos em Python e os módulos desenvolvidos em Java foram utilizados arquivos CSV contendo as informações necessárias para identificação das notas musicais.

### 3.2.1. Leitura das frequências armazenadas em arquivo CSV

Sabendo que um arquivo em formato CSV é um arquivo separado por vírgulas, a extração de informações em qualquer linguagem de programação

torna-se uma tarefa trivial, principalmente considerando que em Java existem pacotes que oferecem classes que facilitam o procedimento, como as classes utilizadas: `BufferedReader` e `FileReader`.

Entretanto, o que uma dificuldade dessa etapa de leitura foram os conceitos lógicos envolvidos durante o processo. Como ao extrair as frequências do arquivo CSV os valores estarão como um conjunto de objetos da classe `String` (a forma mais simples de lidar com cadeia de caracteres em Java), eles serão armazenados em uma lista de arrays de `Strings`, ou seja, uma lista que contém em cada posição uma cadeia de caracteres, que na verdade são os números que compõem os valores de cada frequência obtida.

Como os valores de frequência são números reais que precisam ser utilizados durante a execução do software, não é viável que eles fiquem como objetos da classe `String`. Por isso, é realizada uma conversão para `double` (tipo de dado normalmente utilizado para números reais em Java) de cada uma das posições, e o resultado da conversão vai sendo salvo em uma nova lista, agora do tipo `double`. Sendo assim, a lista de arrays de `Strings` passa a ser uma lista de variáveis do tipo `double`, podendo agora utilizar os valores como números.

### 3.2.2. Análise de frequências obtidas

Após os valores de frequência estarem definidos como números reais, eles podem ser comparados e classificados conforme as frequências das notas musicais. Todavia, para o propósito deste software a afinação foi completamente desconsiderada, sendo assim, foram estabelecidos limites de frequência entre as notas.

Para definir esses limites de frequência das notas, e distinguir quais delas foram emitidas, calcula-se a diferença de frequência entre elas que é, então, dividida pela metade, separando o espaçamento igualmente para ambas.

Segue como exemplo um cálculo de alguns limites das seguintes notas musicais que se encontram na quinta oitava: Fá # (Fá sustenido), ou Sol b (sol bemol), que tem frequência de 370 Hz (Hertz) e Sol que tem frequência de 392 Hz. A diferença entre elas é de 22 Hz, dividindo igualmente este espaçamento ficam 11 Hz para cada uma das notas.

Sendo assim, considerando apenas este espaçamento (370 - 392 Hz), de 370 Hz (frequência exata da nota Fá #) até 381 Hz (limite máximo da nota Fá #) a frequência seria definida como um Fá # (ou Sol b) e de 381 Hz (limite mínimo da nota Sol) até 392 Hz (frequência exata da nota Sol) a frequência seria definida como um Sol. Deste mesmo modo, com a exceção que normalmente os cálculos apresentarão valores muito fracionados, podem ser definidos e utilizados os limites mínimos e máximos de todas as notas musicais em cada uma das oitavas.



## 4 RESULTADOS

A princípio o projeto apresentado foi visto como complexo e muitas vezes sua realização foi desacreditada. Apesar disso, com as orientações recebidas e os estudos realizados o mesmo apresentou resultados satisfatórios e que incentivam a continuação do desenvolvimento.

Até o momento da finalização deste trabalho escrito o software não era capaz de escrever partituras, o que é seu principal objetivo. No entanto, foram dados os principais passos para que essa escrita seja realizada, afinal, o software já é capaz de realizar o processamento de um áudio (inclusive realizar a transformada de Fourier), descobrir a frequência que está sendo emitida e classificar essa frequência em notas musicais, detalhando inclusive a oitava que ela pertence.

Uma vez que é possível descobrir as notas musicais que estão sendo emitidas em um áudio, basta fazer uma integração com um editor de partituras, como o Encore ou o Sibelius 7 (os mais utilizados) para que a partitura seja escrita. Apesar disso não ser uma tarefa complexa, exige conhecimentos e ferramentas não convenientes a este trabalho escrito, devido o seu prazo de entrega.

Portanto, até o momento não foi possível alcançar os resultados esperados. No entanto, a partir do momento que a integração com outro programa (editor de partituras) for realizada, ele passará a cumprir sua função principal de escrever a partitura e atingirá seu objetivo.

## 5 CONSIDERAÇÕES FINAIS

Desenvolver um Software escritor de partituras musicais monofônicas a princípio parece uma tarefa improvável, no entanto, apesar de ser complexa e demorada, é possível de ser realizada, ainda que inicialmente não tenha uma eficiência considerável.

Até o presente estágio de desenvolvimento, o software é capaz de processar os sinais existentes em um arquivo de áudio no formato WAV, realizar a amostragem desses sinais através de técnicas como a Transformada de Fourier e, após reconhecer as frequências presentes no arquivo, consegue definir qual é a nota musical e a oitava em que se encontra, o que torna possível também definir sua posição na partitura conforme a clave definida.

Ainda assim, para que o projeto seja efetivado será necessária a realização de uma integração com um programa editor de partituras musicais. Com isso, como projetos futuros, o desenvolvimento do software terá continuidade a fim de que o mesmo seja completado e aprimorado, conforme a análise de novas técnicas e ferramentas. Além disso, é previsto que o projeto seja inscrito e possivelmente apresentado em feiras de tecnologia e a professores especialistas na área, buscando sugestões de como melhorá-lo e encontrar possíveis investidores.

## 6 TRABALHOS FUTUROS

Entre os trabalhos futuros a serem realizados, estão previstos o desenvolvimento de uma interface gráfica, a possibilidade de configuração da ferramenta pelos usuários e a escrita instantânea de partituras.

### 6.1 INTERFACE GRÁFICA

Para que o usuário visualize e interaja com uma programação, é necessário o desenvolvimento de uma interface gráfica. São poucos os softwares atuais que não utilizam uma interface gráfica, e na maioria das vezes os que não possuem desempenham uma tarefa raramente buscada por um usuário comum (que não é acostumado a utilizar comandos e executar aplicações em terminais).

Saber desenvolver uma interface de forma que ela fique intuitiva e de fácil entendimento é importante para que o software se torne útil e seja preferido pelos usuários. Um exemplo disso são os Sistemas Operacionais, os quais muitas vezes são preferidos por conta de uma interface gráfica chamativa, como é o caso do Windows (Sistema Operacional criado pela Microsoft).

Devido a imprevistos durante o processo não foi possível desenvolver a interface gráfica do software, que foi inserida e será realizada em trabalhos futuros do projeto. Isso será feito conforme as ferramentas de desenvolvimento de interfaces estudadas e o projeto gráfico que ainda será definido conforme as funções desenvolvidas.

A ideia de interface gráfica buscada para o Software é algo similar ao encontrado em editores de partitura, onde todos os elementos da partitura poderão ser manipulados pelos usuários a fim de que ele funcione conforme as necessidades de cada um deles.

### 6.2 CONFIGURAÇÃO DO SOFTWARE

Para que a escrita seja feita de modo efetivo, alguns fatores devem ser levados em consideração, por exemplo, conforme a fórmula de compasso definida a

partitura será escrita de uma forma diferente, e ainda, segundo o andamento musical (grau de velocidade do compasso) ela poderá ser executada mais rapidamente sem alterar sua forma de escrita.

Deste modo, além de inserir o áudio o usuário deve também ter a possibilidade de fazer ajustes conforme a música que desejar escrever. Dentre esses ajustes se faz necessária a existência de configurações como: instrumento musical que está sendo executado, clave a ser definida na partitura, fórmula de compasso, grau de velocidade do compasso e tonalidade da música.

### 6.3 ESCRITA INSTANTÂNEA DE PARTITURAS

A escrita instantânea de partituras conforme as músicas são executadas de início pode parecer uma função desnecessária mas que pode ser muito útil para a realização de outras funcionalidades planejadas.

A vantagem inicial de ter o software funcionando a todo momento é também a possibilidade de verificar se o mesmo está desempenhando sua função corretamente. Dessa forma o usuário poderia verificar durante a execução da música possíveis erros de escrita e corrigi-los, ou ainda, poderia verificar uma escolha errada na configuração realizada por ele mesmo e refazê-la.

Pensando ainda em um diferencial do software, caso ele trabalhasse instantaneamente seria possível também o desenvolvimento de um “modo acompanhamento”, que no caso analisaria as notas de uma partitura já escrita e verificaria se a execução que está sendo realizada no momento está conforme a música desejada. Esse modo visaria principalmente músicos iniciantes por proporcionar um aprendizado intuitivo e pela indicação de erros que muitas vezes passam despercebidos pelos mesmos.

## 7 REFERÊNCIAS BIBLIOGRÁFICAS

- [1] Paulo S. R. Diniz, Eduardo A. B. da Silva, Sergio L. Netto. “Processamento digital de sinais”, Editora Bookman, 2014.
- [2] Simon Haykin, Barry Van Veen. “Sistemas de Comunicação”, Editora Bookman, 2001.
- [3] Lathi, B.P., “Sinais e Sistemas Lineares”, Editora Guanabara, 1979.
- [4] Henrique Frank Werner Puhlmann. “Processamento Digital de Sinais - DSP - Parte 1”, 2014. Disponível na internet em:  
<<https://www.embarcados.com.br/introducao-ao-processamento-digital-de-sinais-dsp-parte-1>>. Acesso em: 08 de fevereiro de 2019.
- [5] José Alexandre Nalon. “Introdução ao processamento digital de sinais”, Editora LTC (Livros Técnicos e Científicos), 2009.
- [6] Allan V. Oppenheim, Alan S. Willsky com S. Hamid Nawab. “Sinais e sistemas 2. ed.”, Editora Pearson, 1983.
- [7] Michael Weeks. “Digital Signal Processing Using MatLab and Wavelets”, Editora Infinity Science Press, 2007.
- [8] Virtuous Tecnologia da Informação, 1998-2019. “Jean Baptiste Joseph Fourier”, Só Matemática. Disponível na Internet em:  
<<https://www.somatematica.com.br/biograf/fourier.php>>. Acesso em 03/03/2019.
- [9] Prof. Dr. Carlos Alberto Ynoguti. “Processamento Digital de Sinais - Transformada Discreta de Fourier”, INATEL (Instituto Nacional de Telecomunicações). Disponível na Internet em:  
<<https://www.inatel.br/docentes/ynoguti/downloads/dsp-s886637-1>>. Acesso em 04/03/2019.
- [10] Michael Pilhofer e Holly Day. “Teoria Musical Para Leigos”, Editora Alta Books, 2013.
- [11] Bohumil Med. “Teoria da Música 4. ed.”, Editora Revista e Ampliada, 1996.

[12] Ponto Ciência. Disponível na internet em:

<<http://pontociencia.org.br/galeria/?content%2FFisica%2FOptica%2FFrequencias+de+cordas+do+Piano.jpg>>

[13] Blogs Canção Nova. “Como podemos escrever as notas musicais”. Disponível na internet em:

<<https://blog.cancaonova.com/musicadedeus/como-podemos-escrever-as-notas-musicais>>. Acesso em 07/03/2019.

[14] Coral Accordis. “Aprendendo a ler partitura”. Disponível na internet em:

<<http://coralaccordis.blogspot.com/2015/07/aprendendo-ler-partitura-musical-parte-2.html>>. Acesso em 05/03/2019.

[15] Música Sacra e Adoração. “Pautas, Notas e Claves. Disponível na internet em:

<<https://musicaeadoracao.com.br/26064/pauta-notas-e-claves>>. Acesso em 23/03/2019.

[16] Teoria Musical Descomplicada. “Compassos Simples”, Teoria Descomplicada.

Disponível na internet em:

<<http://teoriadescomplicada.blogspot.com/p/9-compassos-simples-e-compostos.html>>.

Acesso em 14/03/2019

[17] PyScience-Brasil. “SciPy”, Wikidot. Disponível na internet em:

<<http://pyscience-brasil.wikidot.com/module:scipy>>. Acesso em 05/03/2019.

[18] Dominic John Repici. “The Comma Separated Value (CSV) File Format”,

Creativyst, Inc. 2002-2010. Disponível na Internet em:

<<http://creativyst.com/Doc/Articles/CSV/CSV01.htm>>. Acesso em 05/03/2019.