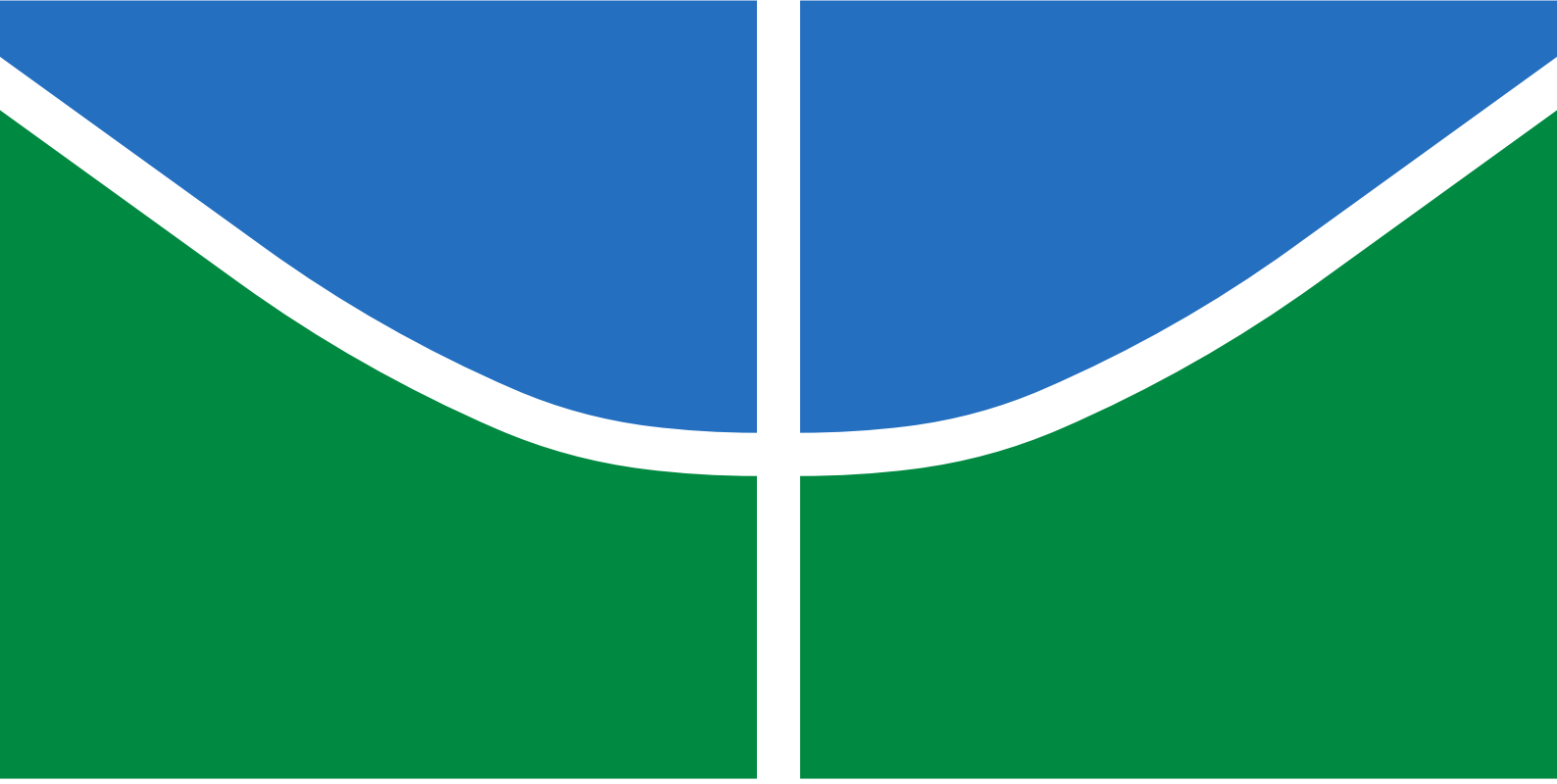


TRABALHO DE GRADUAÇÃO

RECONHECIMENTO DE ACORDES E NOTAS MUSICAIS A PARTIR DE SINAIS DE ÁUDIO EM TEMPO REAL

Daniel de Souza Ramos

Brasília, agosto de 2012

UNIVERSIDADE DE BRASÍLIA

FACULDADE DE TECNOLOGIA

UNIVERSIDADE DE BRASILIA
Faculdade de Tecnologia

TRABALHO DE GRADUAÇÃO

RECONHECIMENTO DE ACORDES E NOTAS MUSICAIS A PARTIR DE SINAIS DE ÁUDIO EM TEMPO REAL

Daniel de Souza Ramos

Relatório submetido ao Departamento de Engenharia Elétrica como requisito parcial para obtenção do grau de Engenheiro Eletricista

Banca Examinadora

Prof. Alexandre R. S. Romariz, ENE/UnB _____
Orientador

Prof. João Luiz A. Carvalho, ENE/UnB _____
Examinador interno

Prof. Adson Ferreira Rocha, FGA/UnB _____
Examinador externo

FICHA CATALOGRÁFICA

RAMOS, DANIEL DE SOUZA

Reconhecimento de acordes e notas musicais a partir de sinais de áudio em tempo real,

[Distrito Federal] 2012.

ix, 50p., 210 x 297 mm (ENE/FT/UnB, Engenheiro Eletricista, 2012).

Trabalho de Conclusão de Curso – Universidade de Brasília. Faculdade de Tecnologia. Departamento de Engenharia Elétrica

1. Reconhecimento de acordes
2. Vetores croma
3. Algoritmos genéticos
4. Tempo real

I. ENE/FT/UnB

II. Título (Série)

REFERÊNCIA BIBLIOGRÁFICA

RAMOS, D. S., (2012). *Reconhecimento de acordes e notas musicais a partir de sinais de áudio em tempo real.* Trabalho de Graduação em Engenharia Elétrica, Publicação ENE 01/2012, Departamento de Engenharia Elétrica, Universidade de Brasília, Brasília, DF, 50p.

CESSÃO DE DIREITOS

AUTOR: Daniel de Souza Ramos

TÍTULO: Reconhecimento de acordes e notas musicais a partir de sinais de áudio em tempo real.

GRAU: Engenheiro Eletricista ANO: 2012

Daniel de Souza Ramos

Rua Expedicionário Abel Mendanha, Quadra 08, Lote 12 - Jardim Cristina.

76630-000 Itaberaí – GO – Brasil.

Dedicatória

Aos meus pais, Celso e Rosali, e à minha irmã, Solange.

Daniel de Souza Ramos

RESUMO

Este trabalho apresenta o desenvolvimento de um sistema de reconhecimento de acordes e notas musicais, voltado para uso em tempo real. A estrutura básica do projeto envolve a construção de vetores croma e classificação por comparação com modelos. Nesse contexto, implementa-se um algoritmo genético para a construção de modelos otimizados. Uma plataforma de testes é desenvolvida e utilizada para analisar a performance em diversas condições e, ao final, é criada uma pequena aplicação, de modo a verificar a aplicabilidade do sistema.

Palavras-chave: reconhecimento de acordes, tempo real, vetores croma, algoritmos genéticos.

ABSTRACT

This work describes the development of a system for chord and musical note recognition, aimed at real-time use. The basic structure of the project involves building chroma vectors and classifying them by comparison with models. In this context, a genetic algorithm is implemented for building optimized models. A test platform is developed and used to analyse the performance under several conditions and, finally, a small application is designed in order to verify the system's applicability.

Keywords: chord recognition, real-time, chroma vectors, genetic algorithms.

SUMÁRIO

LISTA DE FIGURAS

LISTA DE TABELAS

Capítulo 1

Introdução

1.1 Definição do problema

O desenvolvimento de um sistema para reconhecimento de características musicais é um desafio interessante. Como em muitos problemas de processamento de sinais, o primeiro passo é a transformação em frequência do sinal. A partir desse ponto, no entanto, os métodos de análise utilizados em trabalhos da área divergem bastante. Além da análise do espectro, uma etapa de classificação faz-se necessária. Um modo de se realizar a classificação é a comparação com modelos previamente definidos, o que implica na inserção de conhecimento musical no sistema. Ela pode ser feita, também, através de estratégias de inteligência artificial ou, ainda, por uma combinação dos dois métodos.

Algumas peculiaridades acrescentam dificuldade ao projeto. As notas (e, por extensão, os acordes) repetem-se a cada oitava da escala musical, mas precisam ser classificadas da mesma maneira, independentemente da sua localização. Uma solução foi dada por Fujishima [1], em trabalho publicado em 1999. Nesse trabalho ele fez uso de vetores croma, que compõem uma técnica para a análise de sinais independentemente da oitava. Essa técnica serviu de base para vários trabalhos posteriores, como os de Stark e Plumbey [2] e de Oudre, Grenier e Févotte [3].

Outro problema é a diferença encontrada no comportamento dos harmônicos em diversos instrumentos. Para que se consiga um sistema que se comporte bem diante de diferentes fontes, é preciso lançar mão de estratégias que garantam robustez em relação às variações tímbricas do som.

Há, ainda, a possibilidade de existência de ruído e de sons percussivos, que dificultam a realização de uma classificação correta. Estas são ocorrências comuns na maioria das situações de uso prático. É, portanto, importante que sons indesejados não prejudiquem a saída.

1.2 Objetivos do projeto

O projeto tem como objetivo principal o desenvolvimento de um sistema de reconhecimento de acordes e notas musicais que utilize como entrada tanto sons naturais como sintetizados. Tal sistema deve ser suficientemente eficiente, em termos de tempo de processamento, para que possa

Figura 1.1: Afinador digital Qwik Tune®

ser utilizado em aplicações de tempo real. Deseja-se testar diversas configurações possíveis, de maneira que se possam comparar resultados obtidos com mais de uma estratégia e com várias escolhas de parâmetros.

O processamento de sinais para aplicações voltadas à música possui características muito próprias e possibilidades ainda inexploradas. Espera-se que o trabalho apresente conceitos que gerem novas ideias e estimulem a pesquisa na área de tecnologia musical na Universidade de Brasília, servindo de ferramenta para a construção de projetos mais elaborados.

1.3 Contextualização

Os avanços da eletrônica e da computação no último século tiveram impacto inegável no modo como produzimos e acessamos música. Meios eletrônicos foram incorporados à composição de música popular e erudita; diversas mídias de armazenamento foram desenvolvidas e se tornaram obsoletas, culminando no uso da internet como meio de distribuição; ferramentas computacionais passaram a fazer parte do dia-a-dia de apreciadores, músicos e produtores.

A facilidade de se implementarem operações relativamente complexas, como a transformada de Fourier, permitiu que surgissem usos interessantes da tecnologia de processamento digital. Um bom exemplo são os chamados "afinadores eletrônicos", como o da Figura 1.1. São aparelhos simples, geralmente de baixo custo, largamente utilizados para auxiliar a afinação de instrumentos. Esses afinadores, em essência, realizam a identificação de notas musicais.

Diferentemente do reconhecimento de notas isoladas, a identificação de acordes (conjuntos de notas soando simultaneamente) mostra-se particularmente difícil. Devido à sobreposição natural de harmônicos, as notas individuais de um acorde não são facilmente distinguíveis. Técnicas mais elaboradas de análise, muitas vezes empregando métodos de inteligência artificial, foram propostas como maneiras de se solucionar este problema.

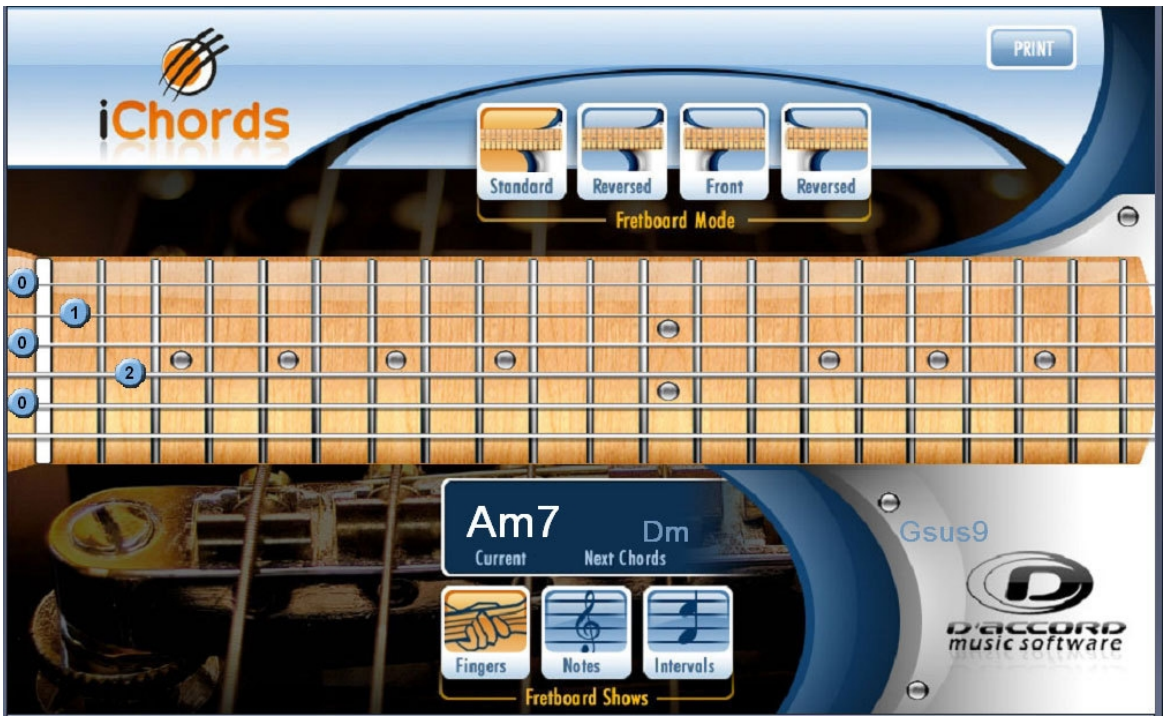


Figura 1.2: Software iChords, da empresa brasileira D'Accord

Recentemente, com estratégias de reconhecimento de acordes mais bem estabelecidas, os resultados se tornaram suficientemente bons para que surgissem aplicações comerciais. Temos o exemplo do software iChords, da empresa brasileira D'Accord, que realiza a identificação de acordes presentes em arquivos de música. O programa, cuja interface está ilustrada na Figura 1.2, se propõe a auxiliar a transcrição da música, com foco na execução em violão e guitarra. Um dicionário de acordes é usado para exibir o posicionamento dos dedos para a construção dos acordes, mas esse posicionamento não é necessariamente o mesmo que foi utilizado na gravação.

Em 2011 foi lançado, pela Ubisoft Entertainment, o jogo Rocksmith™. Este jogo é uma plataforma de aprendizado e entretenimento que se apoia no reconhecimento de notas e acordes para avaliar, em tempo real, a execução de determinada música. O jogo, visto na Figura 1.3, é voltado para a música popular e para um único instrumento, no caso a guitarra elétrica. Essas restrições, no entanto, parecem ser mais fruto de uma proposta comercial do que de possíveis impedimentos técnicos.

1.4 Apresentação do manuscrito

O capítulo 2 apresenta alguns conceitos e ferramentas necessários para o desenvolvimento do trabalho. A transformada discreta de Fourier e sua implementação mais eficiente, a transformada rápida de Fourier, são revistas. Apresenta-se um resumo de escalas e afinações musicais, seguido de uma breve explicação sobre vetores croma. Ao fim, é descrito o funcionamento geral dos algoritmos genéticos.

O capítulo 3 aborda a implementação do sistema detalhadamente. Nele a estrutura principal

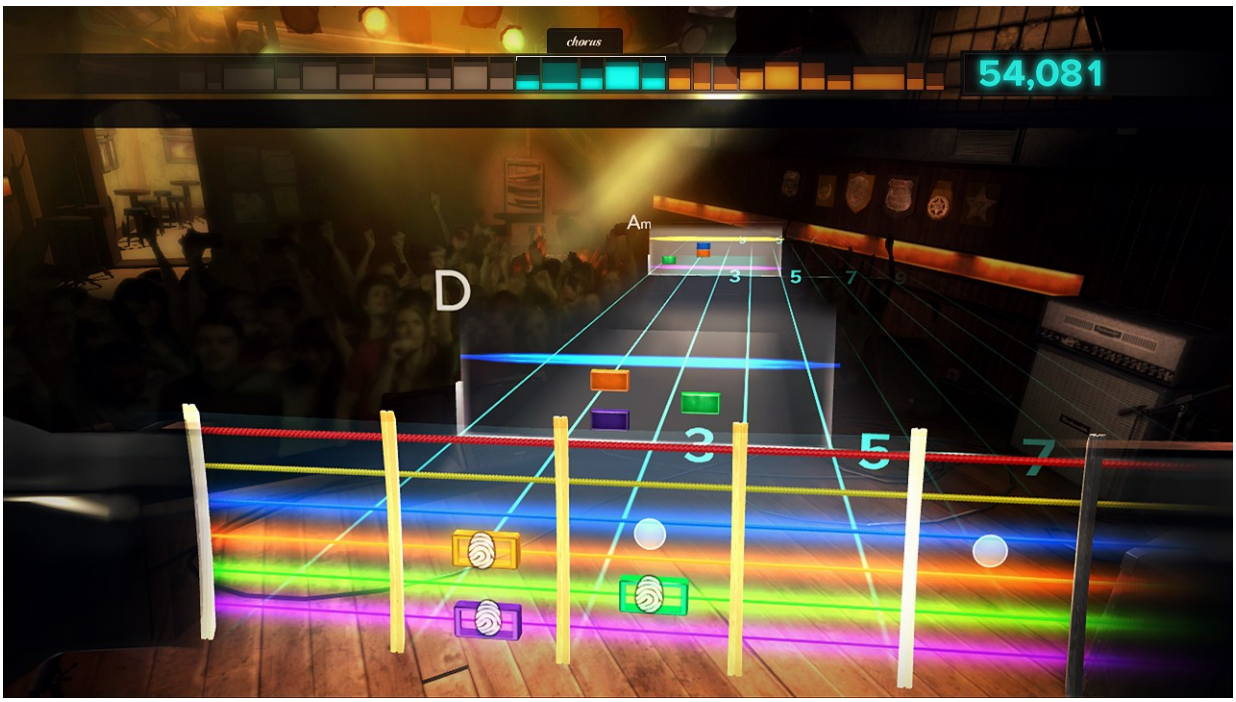


Figura 1.3: Jogo Rocksmith™, da empresa Ubisoft Entertainment

do projeto é apresentada, e as decisões tomadas são explicitadas. No capítulo 4 é apresentada a interface de testes e são discutidos os resultados obtidos. Além disso, o capítulo discorre sobre uma pequena aplicação desenvolvida e sobre algumas outras possibilidades de aplicação do projeto. Conclusões e considerações finais são feitas no capítulo 5.

Capítulo 2

Fundamentos

2.1 Transformada Discreta de Fourier

A transformação de Fourier é uma operação matemática largamente utilizada em processamento de sinais. Por meio dela, um sinal variante no tempo pode ser analisado no domínio da frequência. O estudo desta transformação, aplicada a sinais de tempo contínuo, costuma fazer parte dos currículos do ciclo básico da Engenharia Elétrica.

No entanto, a sua contrapartida em tempo discreto muitas vezes não é encontrada pelo alunos nas disciplinas obrigatórias. Dada a disseminação dos computadores digitais no campo de processamento de sinais nas últimas décadas, o uso de transformadas discretas tornou-se lugar-comum; laboratórios didáticos, nas universidades, utilizam osciloscópios digitais, e computadores pessoais podem ser utilizados para simulações e análises completas. Torna-se, assim, importante conhecer a teoria que move essas ferramentas de análise espectral.

A transformada de Fourier em tempo discreto (*discrete-time Fourier transform*, ou DTFT) é uma transformação realizada sobre uma função discreta no tempo. Supondo que a função $x(n)$ possua essas características, a sua DTFT, $X(\omega)$, é dada por:

$$X(\omega) = \sum_{n=-\infty}^{\infty} x(n)e^{-j\omega n} \tag{2.1}$$

onde ω é a frequência angular, em radianos por segundo, e n pode assumir apenas valores inteiros.

Note que a aplicação direta desta expressão em sinais físicos é impossível, pelo simples motivo de se possuir um número finito de amostras. Além disso, é desejável um resultado discreto, visto que esta é a representação interna utilizada por sistemas digitais em geral. A saída da DTFT, no entanto, é uma função contínua da frequência, e teria de ser calculada apenas em valores discretos específicos de ω.

Para contornar estes problemas e viabilizar a implementação prática da análise discreta em frequência, é utilizada a transformada discreta de Fourier (*discrete Fourier transform*, ou DFT). Considere que a função $x(n)$ seja definida da seguinte maneira:

$$x(n) = \begin{cases} 0, & n < 0 \\ y(n), & 0 \leq n \leq (N-1) \\ 0, & n \geq N \end{cases} \tag{2.2}$$

onde $y(n)$ é um sinal medido em N pontos. Além disso, tome apenas os valores discretos de ω dados por:

$$\omega = \frac{2\pi k}{N}, \tag{2.3}$$

onde $k = 0, 1, ..., N-1$.

A partir destas considerações, a equação 2.1 se torna:

$$X(k) = \sum_{n=0}^{N-1} x(n) e^{-j2\pi \frac{kn}{N}} \tag{2.4}$$

O resultado $X(k)$ é chamado de DFT do sinal de entrada. Como notado por Haykin e Veen [4], a operação realizada pela equação 2.4 gera, na verdade, uma série de Fourier em tempo discreto (*discrete-time Fourier series*, ou DTFS), não caracterizando uma transformada no sentido matemático. Porém, o termo DFT se popularizou de tal maneira que foi adotado por livros e artigos em processamento de sinais, integrando a nomenclatura e o jargão comuns da área.

2.1.1 Transformada Rápida de Fourier

Calcular a DFT exige muito esforço computacional. Utilizando a notação assintótica[1], diz-se que $X(k)$, como calculado pela DFT, é $O(k^2)$. Ou seja, para grandes valores de N, o número máximo de operações necessárias para o cálculo da DFT aumenta quadraticamente. Com o objetivo de diminuir o custo computacional da transformação discreta em frequência, diversos algoritmos alternativos foram desenvolvidos para calculá-la. Estes algoritmos são, geralmente, agrupados sob o nome de transformada rápida de Fourier (*fast Fourier transform*, ou FFT), e realizam a mesma operação da DFT, mas de maneira mais eficiente.

Uma das implementações mais utilizadas da FFT é conhecida como "algoritmo de Cooley-Tukey", em homenagem ao matemático James William Cooley e ao estatístico John Wilder Tukey, que desenvolveram o método na década de 1960. Este algoritmo requer apenas $O(k \cdot \log_2 k)$ operações, o que o torna extremamente eficiente se comparado ao cálculo direto da DFT.

A essência do algoritmo de Cooley-Tukey é o aproveitamento da simetria dos fatores da DFT. Para entender como essa estratégia funciona, vamos calcular a DFT para um número limitado de

[1] A notação assintótica, notação O ou notação Grande-O é utilizada para se determinar a eficiência de um algoritmo. Ela é especialmente útil quando um grande número de operações está envolvido. Dizer que uma função $f(n)$ é $O(g(n))$ equivale a dizer que existem duas constantes positivas c e m, tais que $|f(n)| \leq c \cdot |g(n)|$ para todo $n \geq m$. Assim, $g(n)$ pode ser utilizada como referência para o crescimento máximo da função $f(n)$.

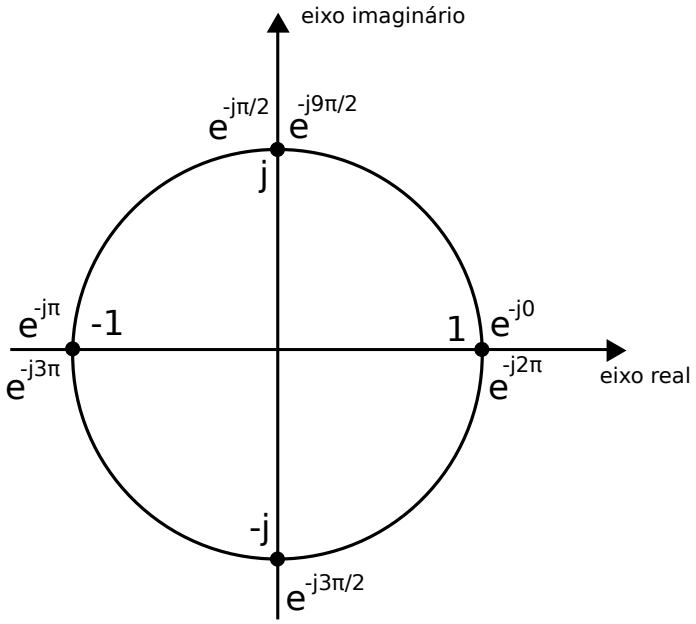


Figura 2.1: Fatores exponenciais da DFT de um sinal com quatro amostras

amostras, utilizando a equação 2.4. Para não sobrecarregar a notação, considere $X(k) = X_k$, e $x(n) = x_n$. Fazendo $N = 4$, temos:

$$\begin{cases} X_0 = x_0 + x_1 + x_2 + x_3 \\ X_1 = x_0 + x_1 e^{-j\frac{\pi}{2}} + x_2 e^{-j\pi} + x_3 e^{-j\frac{3\pi}{2}} \\ X_2 = x_0 + x_1 e^{-j\pi} + x_2 e^{-j2\pi} + x_3 e^{-j3\pi} \\ X_3 = x_0 + x_1 e^{-j\frac{3\pi}{2}} + x_2 e^{-j3\pi} + x_3 e^{-j\pi\frac{9\pi}{2}} \end{cases} \tag{2.5}$$

Lembrando da fórmula de Euler, segundo a qual $e^{ja} = \cos(a) + j\sin(a)$, podemos localizar todos os fatores exponenciais da equação 2.5 no círculo de raio unitário da Figura 2.1. Perceba que muitos dos fatores são, na verdade, iguais, podendo ser reduzidos aos valores 1, -1, j ou $-j$. A partir desta consideração, vamos simplificar as expressões 2.5, resultanto em:

$$\begin{cases} X_0 = x_0 + x_1 + x_2 + x_3 \\ X_1 = x_0 - jx_1 - x_2 + jx_3 \\ X_2 = x_0 - x_1 + x_2 - x_3 \\ X_3 = x_0 + jx_1 - x_2 - jx_3 \end{cases} \tag{2.6}$$

Estas equações podem ser rearranjadas da seguinte maneira:

$$\begin{cases} X_0 = (x_0 + x_2) + (x_1 + x_3) \\ X_1 = (x_0 - x_2) - j(x_1 - x_3) \\ X_2 = (x_0 + x_2) - (x_1 + x_3) \\ X_3 = (x_0 - x_2) + j(x_1 - x_3) \end{cases} \tag{2.7}$$

Observe que os termos entre parênteses se repetem. É justamente este fato que permite a redução do número de operações. Finalmente, para resolver estas equações, fazemos:

$$\begin{cases} x_0 + x_2 = a \\ x_0 - x_2 = b \\ x_1 + x_3 = c \\ x_1 - x_3 = d \end{cases} \tag{2.8}$$

E então, com os resultados parciais a, b, c e d, calculamos:

$$\begin{cases} X_0 = a + c \\ X_1 = b - jd \\ X_2 = a - c \\ X_3 = b + jd \end{cases} \tag{2.9}$$

E obtêm-se, assim, os resultados desejados.

Podemos, agora, comparar o custo computacional do uso do algoritmo de Cooley-Tukey com o custo do cálculo direto da DFT. Considere uma operação do tipo:

$$z = p + q \cdot r \tag{2.10}$$

Esta operação é facilmente realizada por processadores digitais, sendo geralmente implementada como uma instrução básica do processador. Ela é conhecida, na literatura, como multiply-accumulate (ou simplesmente MAC), e será utilizada como medida de eficiência. No algoritmo original, representado pela equação 2.4, deveriam-se realizar quatro operações MAC para cada um dos quatro pontos da DFT, ou seja, seriam necessárias $N^2 = 16$ operações para o cálculo dos resultados finais. Com o aproveitamento das simetrias, foram necessárias apenas as operações das equações 2.8 e 2.9, num total de 8 (ou $N \log_2 N$) operações MAC. Este é exatamente o ganho de eficiência buscado.

A estratégia de reaproveitamento de termos utilizada pelo algoritmo de Cooley-Tukey pode ser estendida para qualquer número de amostras que seja uma potência de dois. Isto é feito dividindo-se o grupo de amostras original em grupos menores, recursivamente, até que se tenha um conjunto de grupos de quatro amostras cada. Essa operação é exemplificada na Figura 2.2, para o caso em que $N = 16$. O algoritmo é aplicado, da maneira vista, em cada um dos grupos de quatro amostras. Os resultados produzirão coeficientes simétricos, que poderão ser reaproveitados durante o cálculo dos grupos de oito amostras, e assim sucessivamente, mantendo a eficiência do algoritmo em $O(k \cdot \log_2 k)$.

É importante ressaltar que o método de Cooley-Tukey exige que o número de amostras seja potência de dois. Esta não é, no entanto, uma restrição grave, pois é comum que se possa escolher a quantidade de amostras de modo a satisfazer a condição. Uma outra solução é realizar *zero-padding*, que consiste na adição de amostras de valor zero para que se atinja o número de amostras

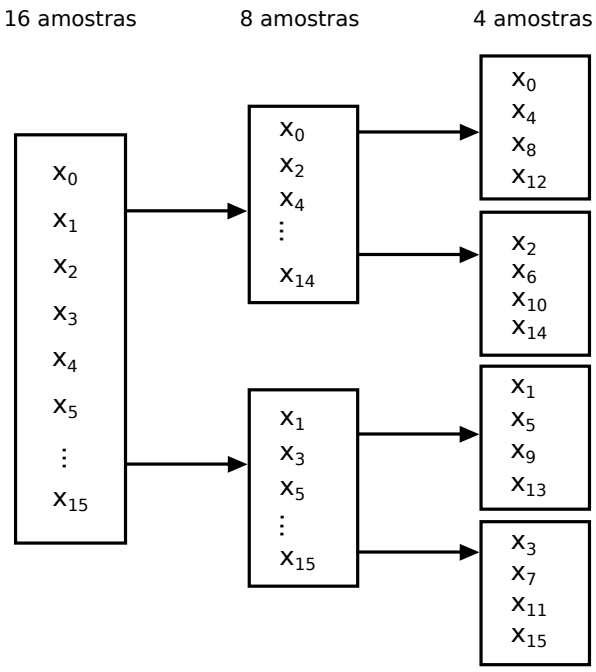


Figura 2.2: Reagrupamento de amostras no algoritmo de Cooley-Tukey

desejado.

Outros algoritmos para o cálculo da FFT existem, muitos dos quais são variantes do método visto. Alguns deles eliminam, por exemplo, a restrição ao número de amostras. Pouco progresso, no entanto, foi feito no sentido de redução de custo computacional, e o algoritmo de Cooley-Tukey continua sendo uma solução importante.

2.2 Escalas musicais e afinação

As notas musicais são constituídas por componentes de frequência relacionados harmonicamente. Como exemplo, o espectro de uma nota Lá produzida por um violão de cordas de nylon está representado na Figura 2.3. As intensidades relativas desses componentes dependem do tipo de instrumento, da técnica utilizada, da região da escala, entre outros fatores.

Na música ocidental, as notas costumam ser associadas aos seus primeiros harmônicos, ou harmônicos fundamentais. Assim, diz-se que o Lá central do piano possui frequência de 440 Hz, quando se quer dizer que esta é a frequência de seu harmônico fundamental. Voltando à Figura 2.3, vê-se que o pico em 110 Hz corresponde ao primeiro harmônico da nota representada, e os diversos picos subsequentes correspondem aos demais harmônicos. Diria-se, assim, que esta é uma nota Lá em 110Hz.

Representando a frequência do harmônico fundamental de uma nota qualquer por f_0, chamamos de "oitava" o intervalo entre esta e outra nota cujo harmônico fundamental f_{0_2} possua o dobro da frequência de f_0. As escalas musicais ocidentais são divisões do intervalo de oitava, que é considerado o limite a partir do qual a escala se repete. Temos, como exemplo, a escala diatônica maior, que divide a oitava em sete notas, representada na Tabela 2.1; e a escala cromática, que

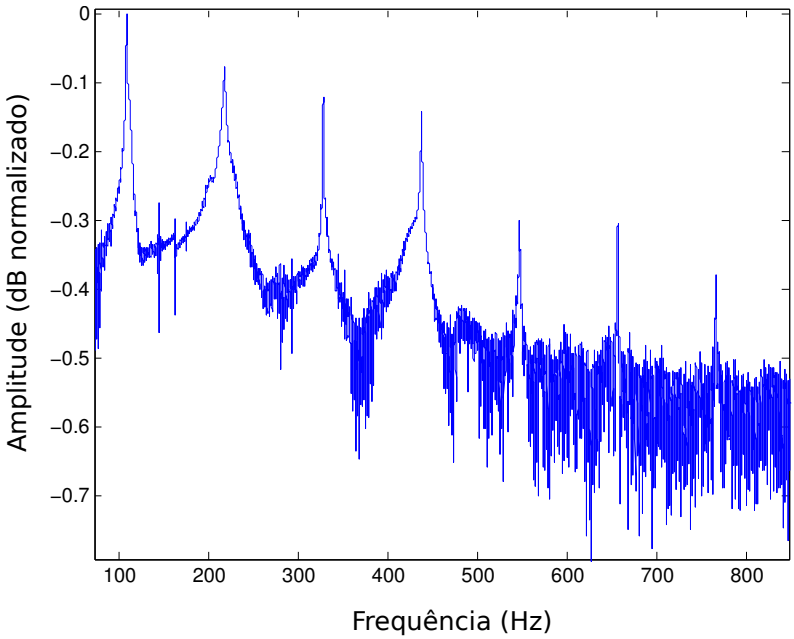


Figura 2.3: Espectro de uma nota Lá tocada em violão

Nota	**Intervalo em relação ao Dó**
Dó	-
Ré	Segunda maior
Mi	Terça maior
Fá	Quarta
Sol	Quinta
Lá	Sexta maior
Si	Sétima maior

Tabela 2.1: Escala diatônica de Dó maior

divide a oitava em doze notas, representada na Tabela 2.2. Nota-se que a escala diatônica está contida na escala cromática.

2.2.1 Sistemas de afinação

Podem-se dividir as escalas de acordo com várias regras. A chamada "afinação justa" utiliza razões entre números inteiros para implementar a escala diatônica, conforme a Tabela 2.3. Estas razões se baseiam nos intervalos harmônicos que ocorrem naturalmente, e por isto essa afinação também é conhecida como "afinação natural".

Um instrumento em afinação justa possui algumas inconveniências. Em especial, ele só pode ser tocado no tom para o qual foi afinado, pois as relações entre as notas se alteram em outros tons. Para resolver esse problema, utiliza-se o chamado "temperamento" da afinação. O temperamento consiste na realização de pequenas alterações na afinação justa, de maneira que a execução em todos os tons seja possível com a mesma afinação.

Nota	Intervalo em relação ao Dó
Dó	-
Ré♭	Segunda menor
Ré	Segunda maior
Mi♭	Terça menor
Mi	Terça maior
Fá	Quarta
Sol♭	Quinta diminuta
Sol	Quinta
Lá♭	Sexta menor
Lá	Sexta maior
Si♭	Sétima menor
Si	Sétima maior

Tabela 2.2: Escala cromática

Nota	Razão em relação ao Dó
Dó	$1/1$
Ré	$9/8$
Mi	$5/4$
Fá	$4/3$
Sol	$3/2$
Lá	$5/3$
Si	$15/8$

Tabela 2.3: Afinação justa da escala diatônica de Dó maior

Intervalo	Frequência
-	f_0
Segunda menor	$1,059 f_0$
Segunda maior	$1,122 f_0$
Terça menor	$1,189 f_0$
Terça maior	$1,260 f_0$
Quarta	$1,335 f_0$
Quinta diminuta	$1,414 f_0$
Quinta	$1,498 f_0$
Sexta menor	$1,587 f_0$
Sexta maior	$1,682 f_0$
Sétima menor	$1,782 f_0$
Sétima maior	$1,888 f_0$

Tabela 2.4: Valores aproximados para a afinação em temperamento igual da escala cromática

Atualmente, o temperamento mais utilizado é o temperamento igual. Nele, a oitava é dividida em doze partes iguais, equivalentes aos graus da escala cromática da tabela 2.2. Essas divisões são chamadas "semitons", e suas frequências são dadas por:

$$f_n = f_0 \cdot 2^{\frac{n}{12}} \tag{2.11}$$

onde f_0 é a frequência fundamental de uma nota de referência e n é a distância inteira, positiva ou negativa, a essa nota. A nota de referência normalmente utilizada é o Lá em 440Hz. A Tabela 2.4 exemplifica os intervalos aproximados gerados pelo temperamento igual.

As notas do temperamento igual são boas aproximações para a afinação justa em todas as tonalidades. Assim, o mesmo instrumento pode ser utilizado para tocar em diversos tons, sem a necessidade de se alterar a afinação.

2.3 Vetores croma

A partir do trabalho de Fujishima [1], tornou-se comum a utilização de vetores croma, também chamados de *pitch class profiles* (PCP), como ferramentas para a classificação de acordes.

Um vetor croma é uma representação do espectro de um sinal. Consiste em um vetor de doze posições, cada uma relacionada a uma nota da escala cromática. Para construir o vetor, primeiramente se relacionam as frequências de todo o espectro às suas notas correspondentes na escala cromática em temperamento igual, independentemente da oitava. A energia presente em regiões que correspondem à mesma nota é, então, somada e o resultado constituirá um componente do vetor. Isto é realizado para as doze notas da escala. O vetor croma assim formado é, geralmente, normalizado após as operações.

Figura 2.4: Inseto *Aquarius remigis*, da família *Gerridae*

É importante notar que a diferenciação entre oitavas não é possível com os vetores croma. A informação de posicionamento absoluto no espectro é perdida durante a montagem do vetor. Isto permite que acordes tocados em qualquer região da escala sejam analisados da mesma maneira, mas tem o inconveniente de impedir a diferenciação de acordes com notas em posições invertidas, por exemplo.

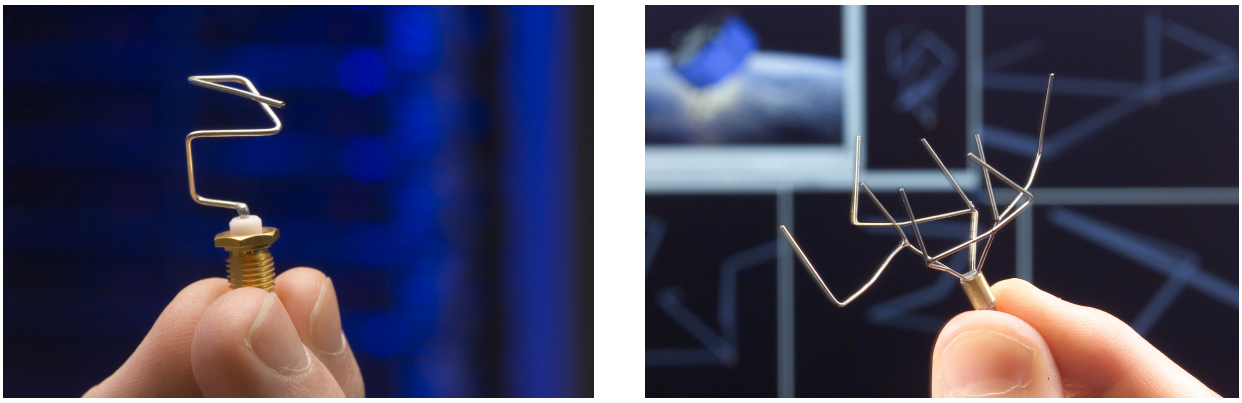
2.4 Algoritmos genéticos

2.4.1 Introdução

A inspiração para o desenvolvimento dos algoritmos genéticos é o processo de seleção natural dos seres vivos. Este processo é capaz de, ao longo do tempo, gerar organismos extremamente bem adaptados ao meio em que se encontram. Diversas adaptações complexas e interessantes ocorrem de maneira natural. Pode-se citar, por exemplo, o caso dos insetos da família *Gerridae*, que vivem sobre a água utilizando o efeito da tensão superficial. Na Figura 2.4 podemos ver um representante desta família. O tamanho de suas patas e a distribuição do peso corporal sofreram repetida seleção de maneira que, em sua configuração atual, o inseto pode tanto se deixar levar pelo movimento da água quanto caminhar sobre ela com grande velocidade.

Diversos outros seres altamente adaptados podem ser encontrados na natureza. Um dos exemplos mais interessantes talvez seja o próprio *Homo sapiens*, cuja evolução lhe permitiu realizar tarefas altamente complexas, do desenvolvimento de ferramentas à representação simbólica e alteração sistemática do meio.

Em problemas de otimização, uma solução analítica pode, muitas vezes, não existir, ou demandar tamanho esforço computacional que a torne impraticável. Estratégias que permitam encontrar



(a) (b)

Figura 2.5: Antenas desenvolvidas pela NASA com o uso de algoritmos genéticos

boas soluções, mesmo que não ótimas, fazem-se necessárias para resolver esse tipo de problema. Com base no processo de evolução dos seres vivos, uma classe de métodos sob o nome geral de "algoritmos genéticos" foi desenvolvida para solucionar este tipo de problema.

Os algoritmos genéticos procuram reproduzir, de certa maneira, os mecanismos da seleção natural. É compreensível, portanto, que grande parte da terminologia utilizada derive da biologia. Assim, uma solução específica é chamada de "indivíduo", enquanto um conjunto de soluções é chamado de "população"; os indivíduos são constituídos de componentes denominados "genes", que podem sofrer alterações chamadas de "mutações"; cada ciclo do algoritmo, ainda, é denomidado "geração".

Um caso célebre de uso de algoritmos genéticos foi o desenvolvimento, pela NASA, de um conjunto de antenas, postas em órbita terrestre em 2006. As antenas, duas das quais se encontram na Figura 2.5, foram desenvolvidas a partir de algoritmos genéticos, que otimizaram o número, o tamanho e o posicionamento dos ramos que compõem a estrutura.

2.4.2 Funcionamento

Dois processos são essenciais para o funcionamento de um algoritmo genético. Eles são: variação e seleção dos mais bem adaptados. A variação garante que novos indivíduos serão formados a cada geração, enquanto o processo de seleção privilegia as melhores soluções em detrimento das piores, segundo algum critério. O diagrama de fluxo da Figura 2.6 descreve, de maneira geral, o funcionamento de um algoritmo genético.

Os indivíduos e seus genes podem ser definidos de várias maneiras, dependendo do problema em questão. Uma vez definidos, uma população inicial deve ser criada. Os indivíduos desta população podem ser gerados tanto de maneira aleatória como a partir de conhecimento prévio.

As soluções individuais são, então, testadas. Os resultados dos testes permitem atribuir, a cada indivíduo, uma medida de adequação. A adequação é, normalmente, utilizada como critério de parada, mas outros critérios podem ser incluídos. Caso todos os critérios desejados sejam atendidos, o algoritmo é interrompido e a solução final será dada pelos indivíduos mais bem-sucedidos. Caso

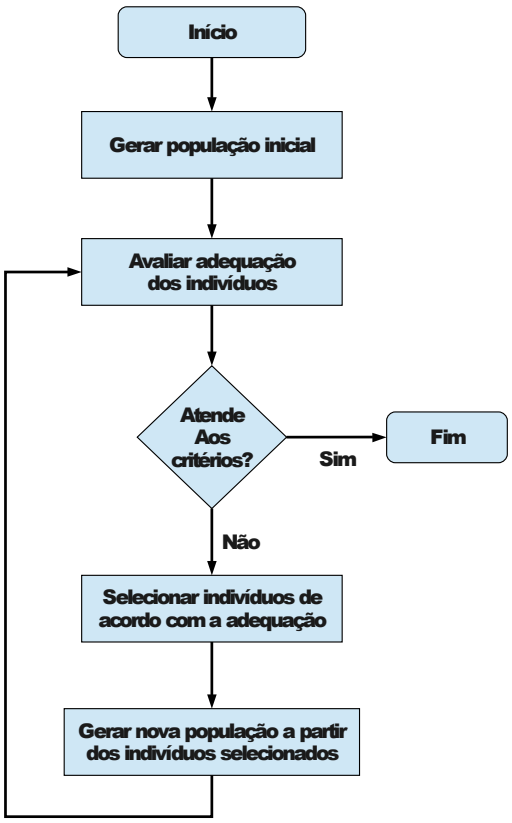


Figura 2.6: Diagrama de fluxo de um algoritmo genético

contrário, o algoritmo segue para a próxima fase.

Não atendidos os critérios de parada, uma nova população deve ser gerada. Os indivíduos com maior adequação são privilegiados, por qualquer método, para servirem de modelo para a geração seguinte, de maneira que todos ou a maior parte dos indivíduos da nova população sejam seus descendentes.

Os indivíduos selecionados podem passar por dois processos para dar origem à população seguinte. Estes processos são o *crossover* e a mutação. O *crossover* consiste na troca de genes entre indivíduos, de maneira a gerar um descendente com características de ambos. A mutação é a alteração aleatória de um ou mais genes, criando um indivíduo completamente novo. É comum utilizarem-se ambos os processos. No entanto, de acordo com o caso, é possível descartar o uso de *crossover*, contando apenas com as mutações para a diferenciação dos indivíduos. A nova população gerada é, então, submetida novamente aos testes, reiniciando o ciclo.

Os algoritmos genéticos tendem a gerar soluções ótimas. Apesar de a solução ser atraída por mínimos locais, a aleatoriedade das mutações e a combinação de soluções realizada pelos *crossovers* são capazes de superar esses mínimos sem a dificuldade que outros algoritmos (como os baseados em gradiente) têm. A principal desvantagem dos algoritmos genéticos é o alto custo computacional, pois um grande número de possíveis soluções é gerado e testado a cada ciclo. No entanto, para aplicações em que a velocidade de execução não é uma prioridade, como na otimização prévia de parâmetros, os algoritmos genéticos são uma boa escolha.

Capítulo 3

Implementação

3.1 Visão geral

No sistema desenvolvido, da obtenção do áudio aos resultados finais, é percorrida uma série de etapas. O diagrama de blocos da Figura 3.1 mostra o processo de modo simplificado. Estes blocos funcionais podem ser implementados de diversas maneiras, cada qual com suas próprias subdivisões e peculiaridades.

Os algoritmos de classificação de acordes encontrados na literatura normalmente seguem essa estrutura, com exceção da etapa de otimização de parâmetros. Apesar disso, não há consenso quanto à maneira de se construir os vetores, tampouco quanto à técnica utilizada para classificação.

A seguir, veremos uma descrição dos pormenores de cada parte do sistema.

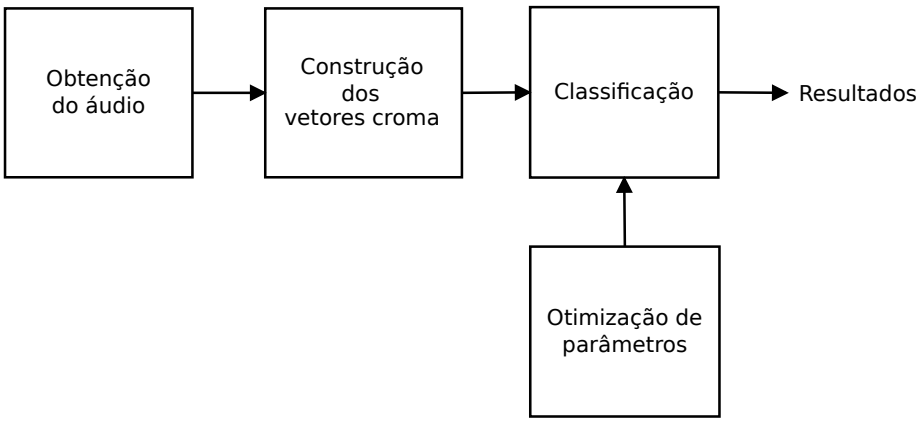


Figura 3.1: Diagrama simplificado do sistema

3.2 Obtenção do áudio

3.2.1 Notas e acordes

Foram consideradas 144 possibilidades diferentes de entrada, sendo 12 notas e 132 acordes, que, a princípio, podem ser construídos em qualquer oitava. A lista completa é composta por:

- As doze notas da escala cromática;
- As doze variações das seguintes tríades:
  - Maiores
  - Menores (m)
  - Suspensas com quarta (sus4)
  - Aumentadas (aug)
  - Diminutas (dim)
- A doze variações das seguintes tétrades:
  - Maiores com sétima (7)
  - Menores com sétima (m7)
  - Maiores com sétima maior (7M)
  - Menores com sétima maior (m7M)
  - Maiores com nona (9)
  - Menores com nona (m9)

A Figura 3.2 ilustra a construção de cada item da lista no piano, utilizando como exemplo as variações em Dó. Outras doze variações são possíveis, uma para cada nota da escala cromática.

Ressalta-se que esta quantidade de acordes supera a encontrada em muitos dos trabalhos sobre o tema. Como exemplo, temos Lee [5] e Bello e Pickens [6], que utilizaram apenas tríades maiores e menores. Um número maior de acordes corresponde de maneira mais fiel ao que é encontrado na prática. A teoria musical nos presenteia com uma quantidade enorme de possibilidades harmônicas. No entanto, alguma limitação é necessária, por motivos práticos. O grande motivo limitador foi o tempo disponível, posto que foram necessárias gravações e testes para todos os acordes da lista. Outro possível limitador é o esforço computacional, que cresce com o número de acordes. Porém, o processamento foi realizado de maneira bastante rápida e não pareceu ser um problema. Estima-se que um número maior de possibilidades de entrada seria suportado sem prejuízos para o resultado.

Uma característica a ser observada é a presença de notas simples na lista de entradas. Isto é incomum em trabalhos do gênero e foi utilizado para explorar a versatilidade da classificação por vetores croma. Notas e acordes são tratados de maneira equivalente no processo, sem nenhuma distinção.

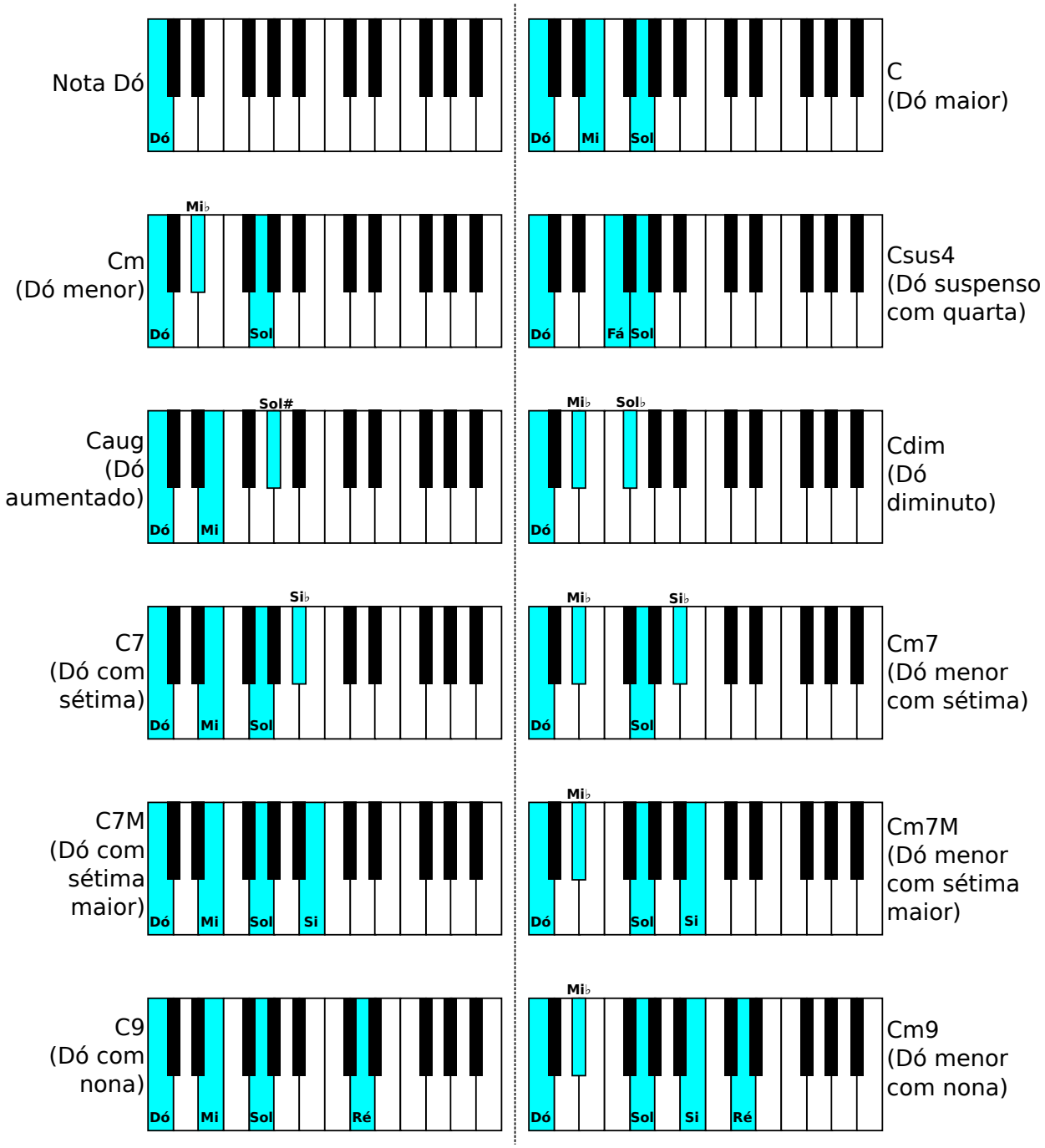


Figura 3.2: Formação das possíveis entradas, utilizando as variações em Dó como exemplo

3.2.2 Sinais de entrada

Dois tipos de entrada foram utilizados: áudio a partir de microfone, para análise em tempo real, e áudio a partir de arquivos previamente gravados.

3.2.2.1 Arquivos de áudio como entrada

Os arquivos de entrada devem estar em formato WAVE. A frequência de amostragem deve ser pelo menos o dobro da máxima frequência a ser utilizada para a montagem dos vetores croma, de modo a satisfazer o teorema da amostragem.

Para uso em testes e em otimização, foram gravados 396 arquivos de áudio. Sua frequência de amostragem foi de 44100Hz, com 16 bits por amostra. A duração das gravações foi variável, em geral entre 1 e 5 segundos.

Os arquivos gravados podem ser divididos em:

- 264 gravações de sons naturais, sendo:
  - 252 com violão de cordas de nylon
  - 12 com voz
- 132 gravações de sons sintetizados, com instrumentos escolhidos entre os pré-definidos do programa ZynAddSubFx, um sintetizador para a plataforma Linux. Foram 12 gravações em cada um dos instrumentos:
  - Acordeão
  - Clarinete
  - Cordas
  - Fagote
  - Órgão
  - Órgão Hammond
  - Piano 1
  - Piano 2
  - Trompete
  - Violão Nylon
  - Xilofone

Foram 33 gravações (22 de sons naturais e 11 de sons sintetizados) para cada uma das doze categorias (maior, menor, 7, m7, 7M, m7M, 9, m9, sus4, aug, dim e notas simples). Cada instrumento gerou pelo menos um arquivo de cada categoria. Todas as gravações foram feitas com o auxílio do software de gravação e edição Audacity.

Procurou-se utilizar uma grande variedade de instrumentos sintetizados, de forma a não enviesar os resultados. A obtenção de sons naturais diversificados, no entanto, impôs maior dificuldade. Apenas um instrumento, o violão, estava disponível. Tentou-se compensar a falta de variedade executando-se o violão de diversas maneiras durante as gravações. Os acordes variaram entre as regiões do braço do instrumento e foi utilizada mais de uma técnica de execução com a mão direita, como dedilhado, palhetada, entre outras. Ainda como tentativa de amenizar a pouca diversidade, foram utilizados também sons vocais. Neste caso, várias notas foram gravadas com uma só voz e então sobrepostas para formar acordes.

Um microfone comum foi utilizado para gravar os sons naturais, o que introduziu tanto ruído ambiente como ruído do sistema de gravação. Como nosso objetivo é testar o funcionamento do sistema em situações reais, a presença de ruído é desejável. No entanto, podem ter havido prejuízos para o algoritmo de otimização, já que um ruído semelhante presente em muitas gravações poderia induzir ao overfitting.

Os instrumentos sintetizados foram gravados, a partir do sintetizador, diretamente no Audacity, através da interface de áudio Jack (Jack Audio Connection Kit). Neste caso não há ruído ambiente nem de gravação, pois os programas foram conectados diretamente.

3.2.2.2 Sinal de microfone como entrada

A captura de áudio em tempo real foi realizada, em ambiente Linux, através da interface Jack. A conexão à interface se dá por meio de código escrito em linguagem C incorporado ao Matlab[1].

Um buffer da entrada é mantido. Este buffer, no entanto, não é utilizado diretamente para a transformação em frequência. Um vetor com tamanho definido é atualizado com os dados do buffer a cada ciclo de reconhecimento. O buffer, portanto, deve ser capaz de conter pelo menos a mesma quantidade de dados que é acrescentada ao vetor em cada ciclo.

A atualização do vetor pode ser feita de duas maneiras. A primeira consiste em se excluir um trecho antigo do vetor e acrescentar-se um trecho novo a cada reconhecimento realizado, conforme ilustrado na Figura 3.3. Essa forma foi utilizada durante o desenvolvimento do sistema e em todos os testes.

A segunda solução exige conhecimento do andamento da música que está sendo executada. Apesar de, em princípio, não existir um andamento fixo quando se realizam testes livres, na aplicação descrita na seção 4.3.1 essa informação é muito facilmente adquirida. Neste caso o vetor utilizado pode ser um segundo buffer, maior, ao qual o buffer de entrada é acrescentado pouco a pouco. O vetor, assim, é analisado a cada passo, enquanto aumenta de tamanho. Quando o conteúdo armazenado corresponde ao tamanho de uma divisão definida do compasso musical, vetor e buffer de entrada são esvaziados e começa-se novamente. A Figura 3.4 ilustra o processo. A vantagem dessa abordagem é possibilidade de se ter maior quantidade de dados para análise, o que permite gerar resultados mais precisos conforme o vetor cresce em tamanho.

[1] Foi utilizado o código do projeto JacktLab, disponibilizado gratuitamente em: http://web.mit.edu/mmt/www/matlabjack

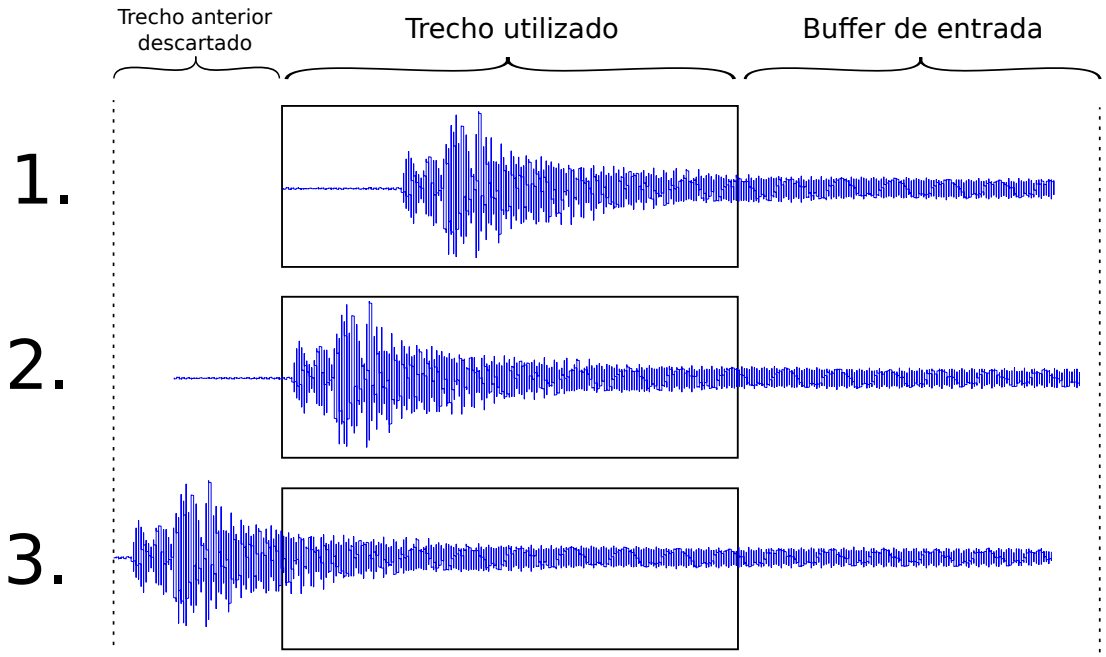


Figura 3.3: Entrada de áudio

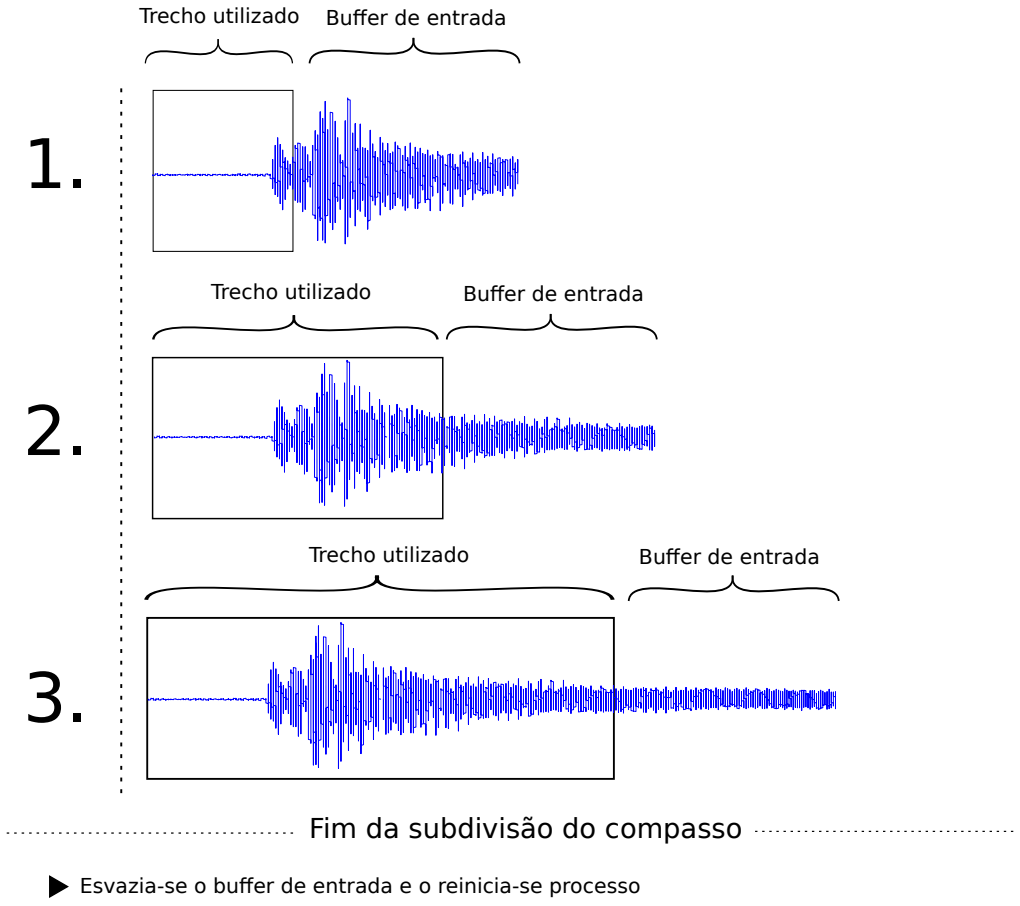


Figura 3.4: Solução alternativa para entrada de áudio

3.3 Construção dos vetores croma

3.3.1 Transformação em frequência

Inicialmente, o sinal de áudio no tempo deve ser levado para o domínio da frequência. Essa operação foi realizada através da transformada rápida de Fourier.

3.3.1.1 Representação em dB

O módulo da transformada e o seu valor em dB nos dão informações semelhantes. No entanto, os poucos picos maiores são muito valorizados quando se utiliza o módulo. A visualização do conteúdo harmônico melhora significativamente quando os valores estão em decibéis. Optou-se, assim, por trabalhar com a transformada em dB, que é obtida através da equação:

$$|\boldsymbol{F}|_{dB} = 20 \cdot \log_{10}(\frac{|\boldsymbol{F}|}{p_r}) \tag{3.1}$$

onde $\boldsymbol{F}$ são os valores da transformada de Fourier e p_r é um valor de referência qualquer, na mesma dimensão de $|\boldsymbol{F}|$. Por simplicidade, escolhemos $p_r = 1$.

3.3.1.2 Filtragem com base na percepção humana

Sabe-se que nossa audição favorece certas frequências em detrimento de outras. É razoável supor que os instrumentos musicais sejam feitos de maneira a compensar a menor sensibilidade do nosso sistema auditivo a certas regiões do espectro. A análise espectral direta dos sons indicaria, então, intensidades discordantes das percebidas. Notas muito graves ou muito agudas precisam de maior intensidade para soarem com a mesma intensidade aparente das notas centrais. Assim, um instrumento precisa beneficiar essas faixas de frequência para que eles pareçam estar em equilíbrio com as demais.

Com essa premissa, desenvolveu-se um filtro que se assemelha à filtragem natural realizada pela audição humana. A Figura 3.5 exibe os pontos da resposta desejada do filtro. Devido à dificuldade de se conseguir dados precisos[2], utilizaram-se valores aproximados, baseados nas curvas geradas por Suzuki e Takeshima [8].

Através do algoritmo RLS (*recursive least squares*), foi gerado um filtro que melhor atendesse aos valores desejados. Esse algoritmo, encapsulado em um comando padrão em Matlab, tenta aproximar a resposta desejada de um filtro através de estratégias de minimização de erro. No entanto, como se pode ver na Figura 3.6(a), a região de baixas frequências não correspondeu ao esperado, aproximando assintoticamente um valor fixo. Um filtro de Butterworth, passa-altas, foi criado para complementar o anterior. Por tentativa e erro chegou-se a uma ordem e a uma frequência de corte que melhor aproximassem a curva esperada. Assim, decidiu-se por uma frequência

[2]Os valores das curvas de intensidade percebida foram padronizados pela ISO [7]. O documento não é disponibilizado gratuitamente, o que impediu o acesso aos dados.

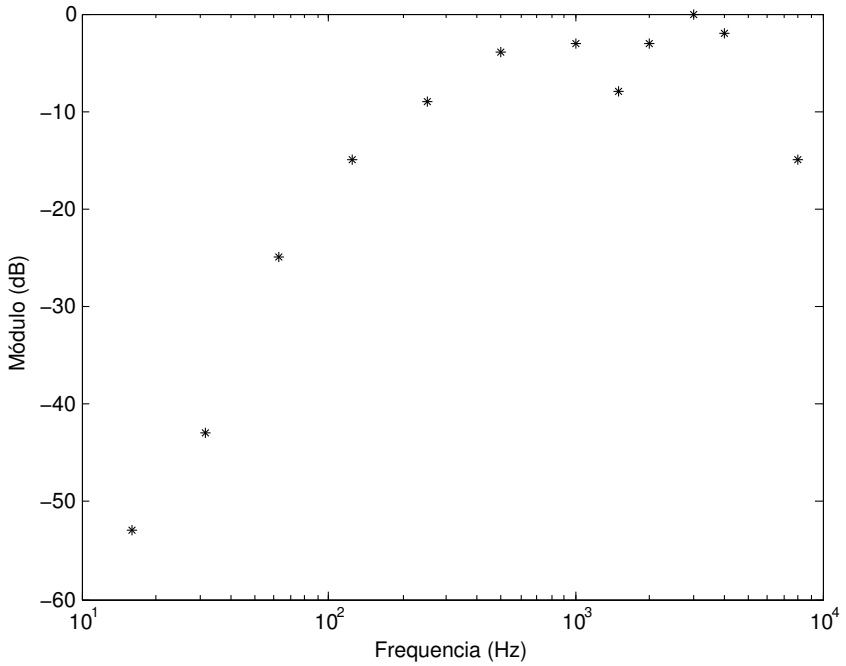


Figura 3.5: Resposta desejada do filtro

de corte de 200Hz e um filtro de segunda ordem. A resposta final, unindo-se os dois filtros, está representada na Figura 3.6(b).

Diante da imprecisão dos dados de referência, considerou-se importante testar os resultados com e sem filtragem. Estes testes revelaram que o uso do filtro não melhorou significativamente os resultados. Assim, a etapa de filtragem foi, ao final, abandonada. Mais detalhes serão vistos no capítulo 4.

3.3.2 Frequências de referência para a construção dos vetores croma

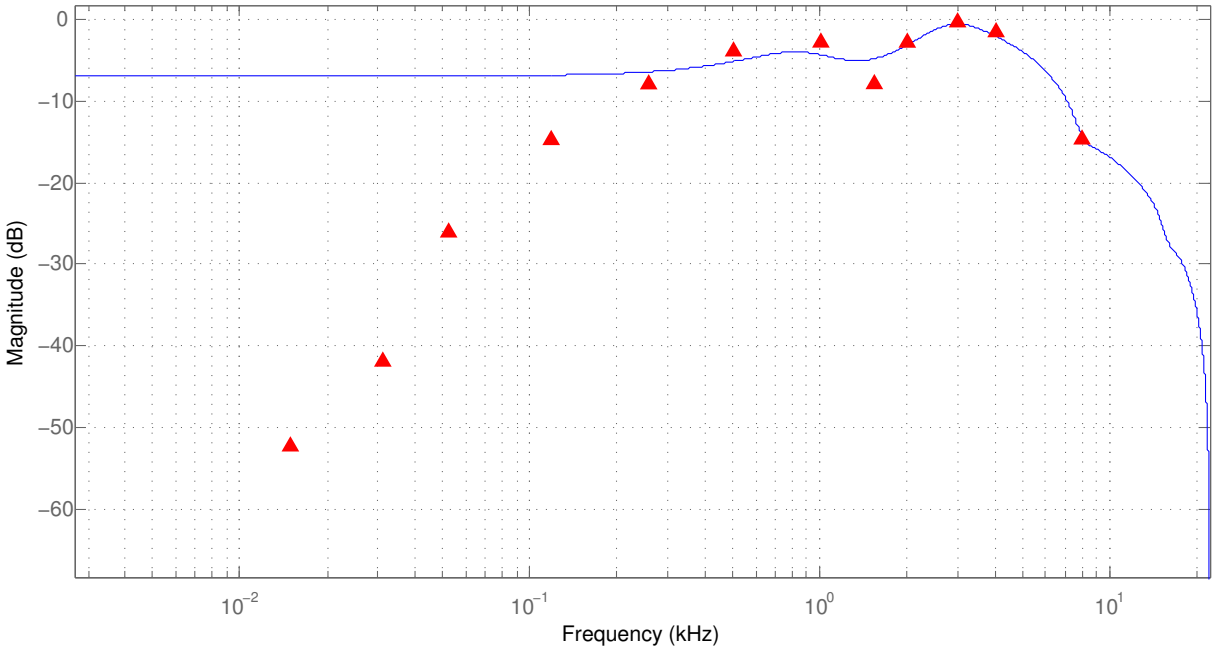
Para gerar os vetores croma são necessários valores de referência para cada nota da escala cromática. O valores utilizados foram os da afinação temperada, gerados através da equação 3.2. Esta equação relaciona a frequência de uma nota, f_n, à sua posição na escala, n, a partir de uma referência inicial, f_0.

$$f_n = f_0 \cdot 2^{\frac{n}{12}} \tag{3.2}$$

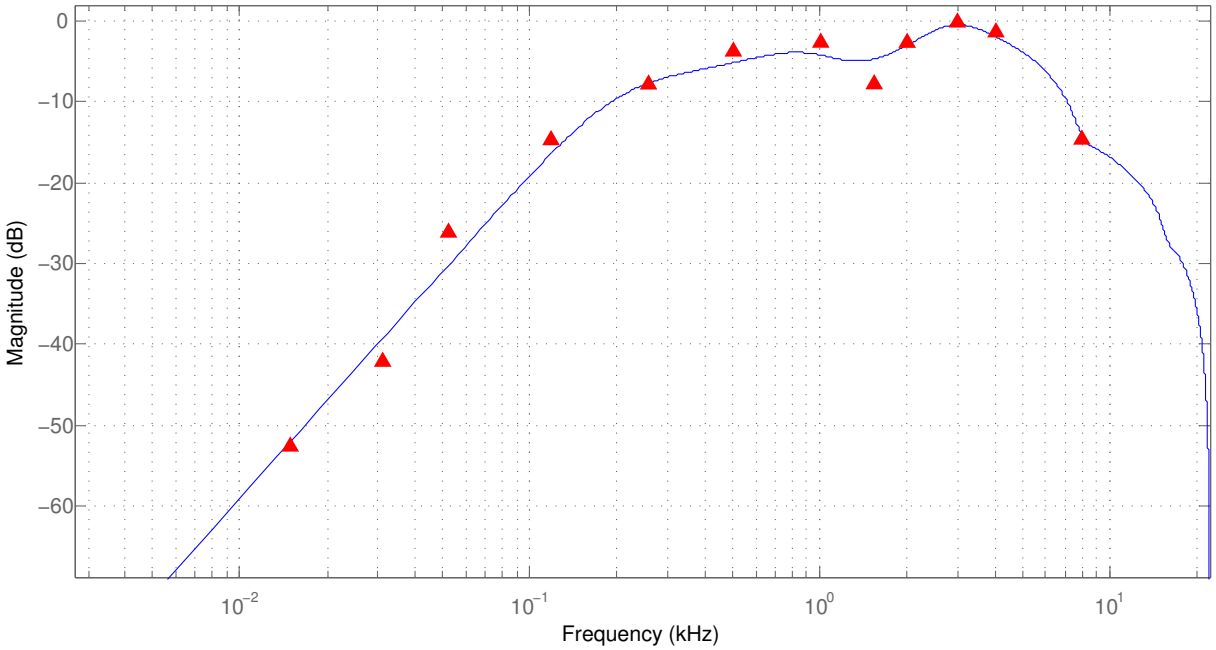
A posição n deve ser entendida como a distância da nota de referência à nota desejada, podendo, inclusive, ser um valor negativo. Partindo da consideração de que temos uma nota Lá em 440Hz, a nota Lá# seguinte, por exemplo, estará à distância 1. Assim, fazendo $f_0 = 440Hz$ e $n = 1$, temos a frequência da nota Lá#:

$$f_1 = 466,16Hz$$

Desta maneira, os demais valores foram calculados em toda a faixa audível (20Hz a 20KHz). A Tabela 3.1 mostra os resultados, que aqui foram aproximados por limitações de espaço. Nesta tabela, cada coluna corresponde a uma nota da escala cromática, enquanto as linhas correspondem



(a) Resposta após a primeira aproximação (triângulos indicam a resposta desejada)



(b) Resposta final (triângulos indicam a resposta desejada)

Figura 3.6: Respostas obtidas para o filtro

Dó	**Dó#**	**Ré**	**Ré#**	**Mi**	**Fá**	**Fá#**	**Sol**	**Sol#**	**Lá**	**Lá#**	**Si**
				20,60	21,83	23,12	24,50	25,96	27,50	29,14	30,87
32,70	34,65	36,71	38,89	41,20	43,65	46,25	49,00	51,91	55,00	58,27	61,74
65,41	69,30	73,42	77,78	82,41	87,31	92,50	98,00	103,8	110,0	116,5	123,5
130,8	138,6	146,8	155,7	164,8	174,6	185,0	196,0	207,7	220,0	233,1	246,9
261,6	277,2	293,7	311,1	329,6	349,2	370,0	392,0	415,3	440,0	466,2	493,9
523,3	554,4	587,3	622,3	659,3	698,5	740,0	784,0	830,6	880,0	932,3	987,8
1047	1109	1175	1245	1319	1397	1480	1568	1661	1760	1865	1976
2093	2217	2349	2489	2637	2794	2960	3136	3322	3520	3729	3951
4186	4435	4699	4978	5274	5588	5920	6272	6645	7040	7459	7902
8372	8870	9397	9956	10548	11175	11840	12544	13290	14080	14917	15804
16744	17740	18795	19912								

Tabela 3.1: Frequências de referência em todo o espectro audível. Os valores são aproximados e estão dados em Hz.

a diferentes oitavas. A primeira linha é a oitava 0, a segunda é a oitava 1, e assim por diante, de maneira que iniciamos com o Mi 0 e terminamos em Ré# 10.

Utilizar toda a faixa audível é desnecessário. A maioria dos instrumentos gera notas cujos harmônicos fundamentais se encontram bem abaixo de 20kHz e acima de 20Hz. No violão, por exemplo, essas frequências vão de 82,41Hz a 1,319kHz (Mi 2 ao Mi 6). Já o piano, notadamente um instrumento de grande extensão, tem um alcance típico do Lá 0 ao Dó 8, ou seja, de 27,5Hz a 4,186kHz. Frequências muito altas ou muito baixas costumam ser associadas a ruídos e a características subjetivas como "brilho" do som. Por esses motivos, é vantajoso que tais frequências sejam excluídas da nossa análise. Como será visto no capítulo 4, foi acrescentada a opção de truncamento da tabela, de maneira que se pudesse utilizar um intervalo mais restrito para a construção dos vetores croma.

3.3.3 A questão da afinação

A Tabela 3.1 contém referências para a afinação por temperamento igual. São valores que fazem sentido para instrumentos de teclas ou com trastes. No entanto, diferem da afinação justa, que é mais próxima do que é tocado em um violino ou cantado por um coral, por exemplo. Outro caso a ser considerado é o da desafinação. Não é incomum que o instrumento esteja um pouco desafinado. Um sistema robusto deve ser capaz de manter seu desempenho em todas essas situações.

Assim, para o cálculo do vetor croma, definem-se faixas de frequência em torno das referências, como ilustrado na Figura 3.7. Essa estratégia faz com que a exatidão da afinação seja irrelevante, introduzindo a robustez desejada.

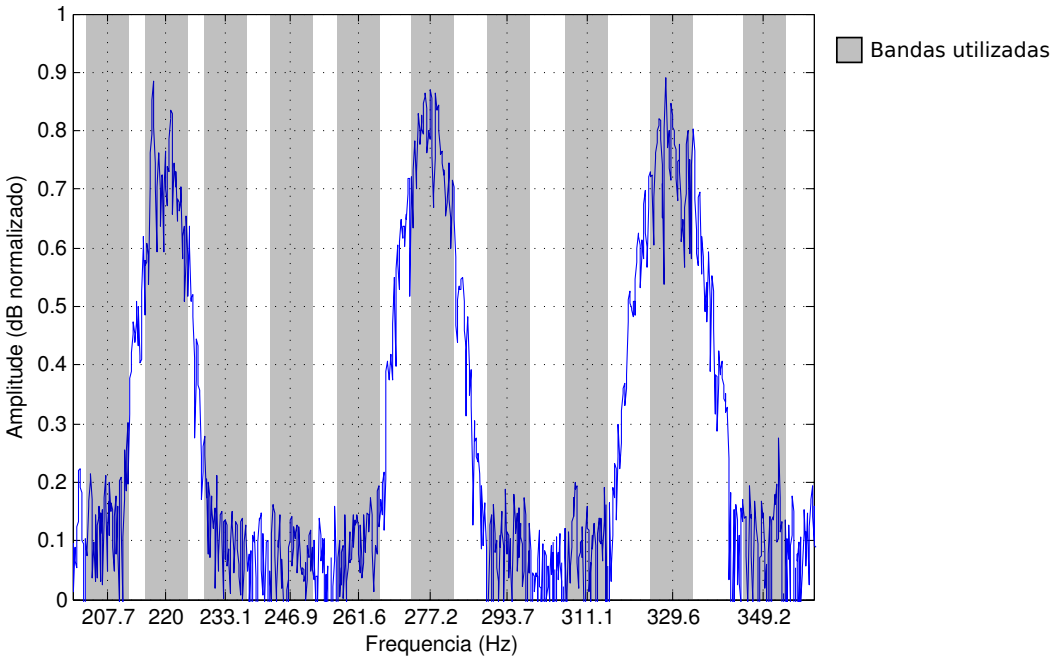


Figura 3.7: Faixas de frequência em torno dos valores de referência

3.3.4 Determinação das faixas de frequência

Dada a natureza logarítmica das notas musicais, as distâncias entre as frequências aumentam progressivamente. Portanto, o tamanho das bandas utilizadas também deve aumentar, seguindo a tendência logarítmica.

É mais prático determinar uma distância Δn em escala linear, e então convertê-la para valores de frequência. A Figura 3.8 mostra uma faixa linearizada em torno do Lá central (440Hz, ou $n = 0$).

Após escolher um Δn adequado, as fronteiras devem então ser calculadas como valores de frequência. Em torno de uma dada frequência de referência, os intervalos superiores e inferiores da banda serão, respectivamente:

$$\Delta f_+ = f_0[2^{\frac{n+\Delta n}{12}} - 2^{\frac{n}{12}}] \tag{3.3}$$

$$\Delta f_- = f_0[2^{\frac{n-\Delta n}{12}} - 2^{\frac{n}{12}}] \tag{3.4}$$

Note que as equações 3.3 e 3.4 são apenas aplicações diretas da equação 3.2. Pode-se calcular Δf_+e Δf_- também como percentuais da frequência atual:

$$\Delta f_{+\%} = (2^{\frac{\Delta n}{12}} - 1) \cdot 100 \tag{3.5}$$

$$\Delta f_{-\%} = (2^{\frac{-\Delta n}{12}} - 1) \cdot 100 \tag{3.6}$$

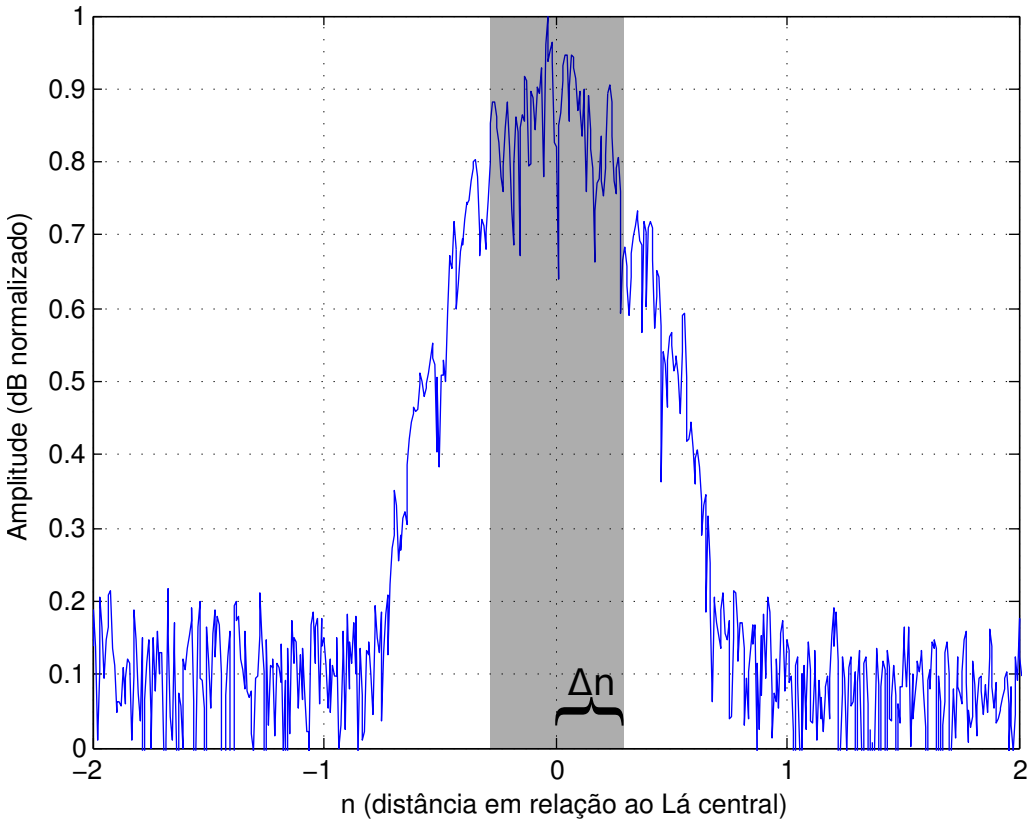


Figura 3.8: Faixa em torno do Lá central, comΔn em evidência

O tamanho das faixas é um ponto importante. Elas não podem ser demasiado grandes, sob o risco de englobarem frequências intermediárias, que não nos interessam. Também não podem ser pequenas, pois não incluiriam pequenas desafinações. Somos obrigados a encontrar uma solução de compromisso entre robustez e seletividade.

Essa solução foi determinada de maneira empírica. O valor máximo para que não ocorra sobreposição das faixas é $\Delta n = 0.5$. A maior diferença entre notas das afinações justa e temperada, que ocorre no intervalo de sétima menor, corresponde a $\Delta n \simeq 0.31$. Após alguns testes com valores intermediários, foi escolhido $\Delta n = 0.35$.

3.3.5 Extração dos valores

Uma prática comum encontrada na literatura é a de se utilizar toda a energia contida nas faixas para a construção do vetor croma. Essa abordagem, porém, acaba introduzindo muita energia indesejada; ruídos atonais, por exemplo, podem estar distribuídos dentro das bandas de frequência.

Os harmônicos musicais, por outro lado, produzem picos evidentes. É mais sensato utilizar essa característica a nosso favor. Assim, foi escolhido o valor de pico dentro de cada faixa de frequência para compor os vetores croma. Todos os valores de pico nas bandas correspondentes à nota Dó, em todas as oitavas, foram somados e inseridos na primeira posição do vetor. Em seguida todos os picos correspondentes à nota Dó#, e assim por diante. A Figura 3.9 mostra esta operação em um trecho do espectro.

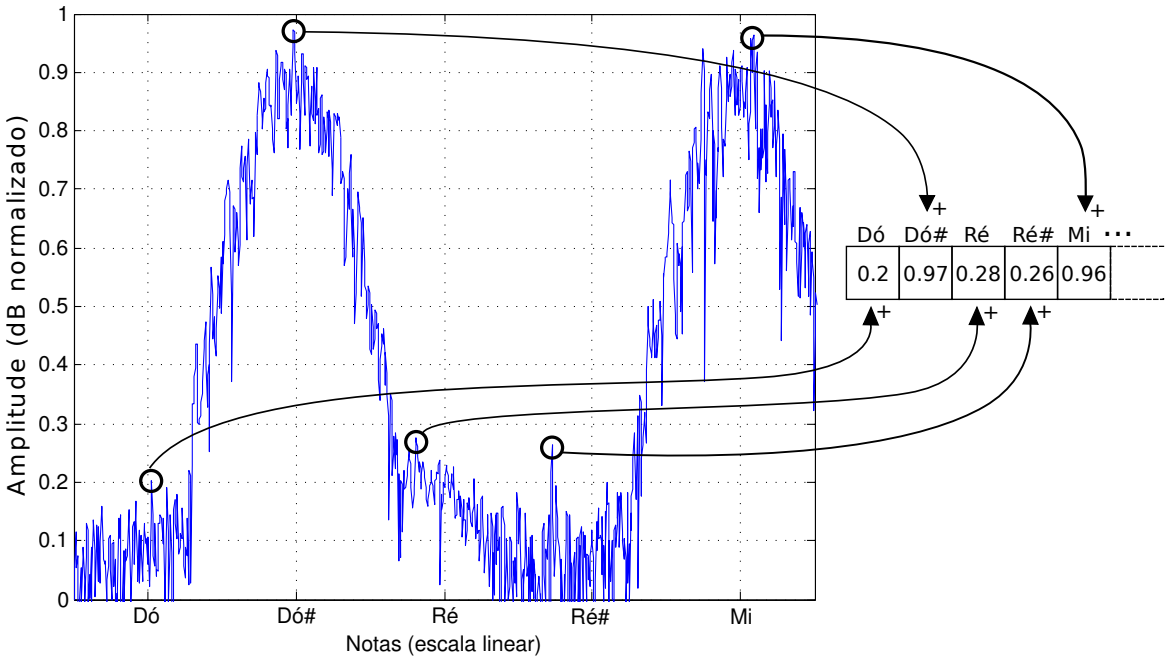


Figura 3.9: Extração de picos em um trecho do espectro para construção do vetor croma

3.3.6 Ajustes e normalização

É importante, para nossa análise, eliminar qualquer influência da intensidade total do som. O vetor croma deve ser normalizado; isto é feito dividindo-se cada elemento do vetor pelo maior elemento existente, em módulo.

Também é desejável que a referência inicial utilizada no cálculo da FFT em dB não se reflita nos valores do vetor. A Figura 3.10 exemplifica tal influência. Nela vemos dois vetores normalizados gerados a partir do mesmo sinal de entrada mas utilizando referências diferentes na equação 3.1. Vê-se que, dependendo da referência escolhida, podemos ter desde valores negativos até valores que se acumulam nas proximidades do limite máximo do vetor.

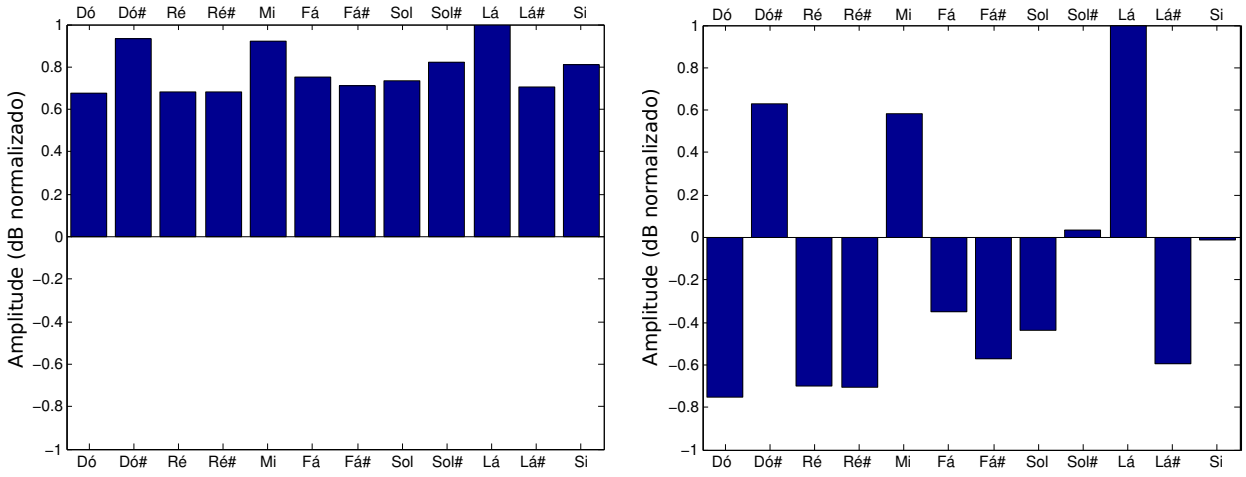
Temos duas opções para contornar este problema:

- Deslocar a média dos valores do vetor para a posição central;
- Deslocar o menor valor para a posição zero.

Ambas devem ser realizadas antes da normalização, de modo que o vetor final contenha valores bem distribuídos entre zero e um. As duas opções foram implementadas e comparadas neste trabalho. A Figura 3.11 possui vetores croma gerados a partir do mesmo sinal que resultou nos vetores da Figura 3.10, mas utilizando o ajuste de média e o ajuste de menor valor, respectivamente.

3.4 Classificação

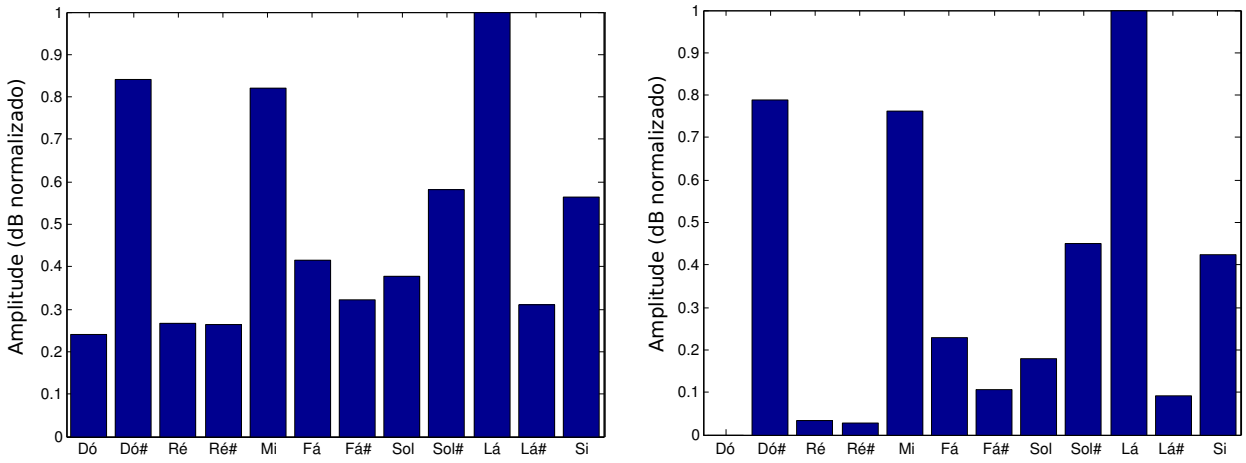
Obtidos os vetores croma, pode-se passar para a etapa de classificação. Há divergência na literatura sobre o melhor meio de se classificar os acordes. Muitos autores, como Cho e Bello [9],



(a) Utilizando $p_r = 5 \cdot 10^{-6}$

(b) Utilizando $p_r = 5$

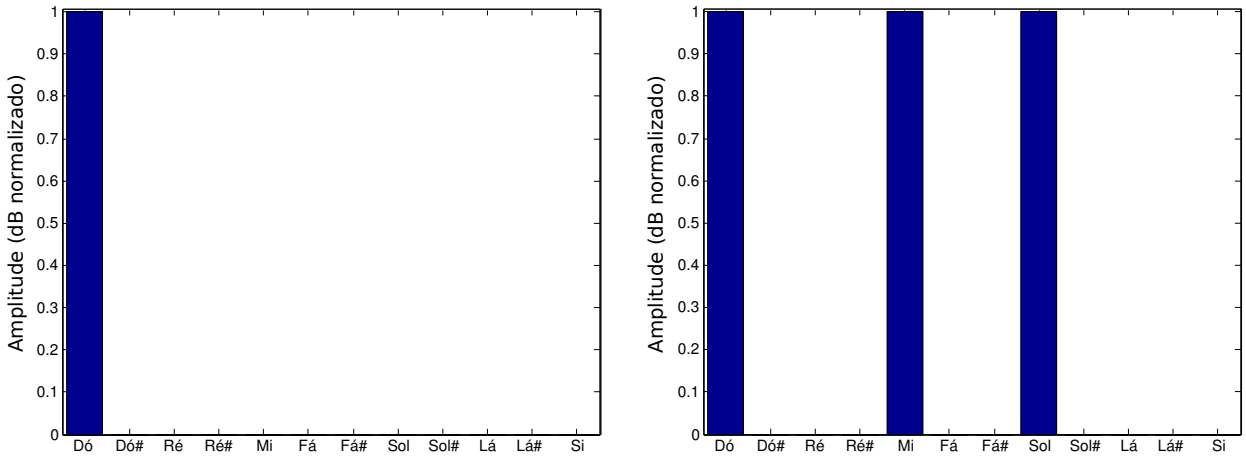
Figura 3.10: Influência da referência da equação 3.1 nos vetores gerados



(a) Média ajustada para 0.5

(b) Menor valor ajustado para 0

Figura 3.11: Vetor croma após diferentes ajustes



(a) Nota Dó

(b) Acorde C

Figura 3.12: Modelos simples

utilizaram modelos ocultos de Markov (HMM, na sigla em inglês), com valores iniciais baseados em conhecimento musical. Outro método comumente utilizado consiste em definir vetores-modelo para todos os acordes possíveis e então encontrar o que mais se aproxima do vetor calculado. Essa foi a abordagem de Oudre et al. [10].

Apesar da maior complexidade no uso de HMMs, os resultados obtidos com os dois métodos têm sido similares. Assim sendo, optou-se, por simplicidade, pela comparação com modelos de acordes. O algoritmo é suficientemente rápido para nos dar resultados em tempo real, um dos objetivos do trabalho. Além disso, essa abordagem nos deu margem para trabalhar com otimização de parâmetros, como será visto na seção 3.5.

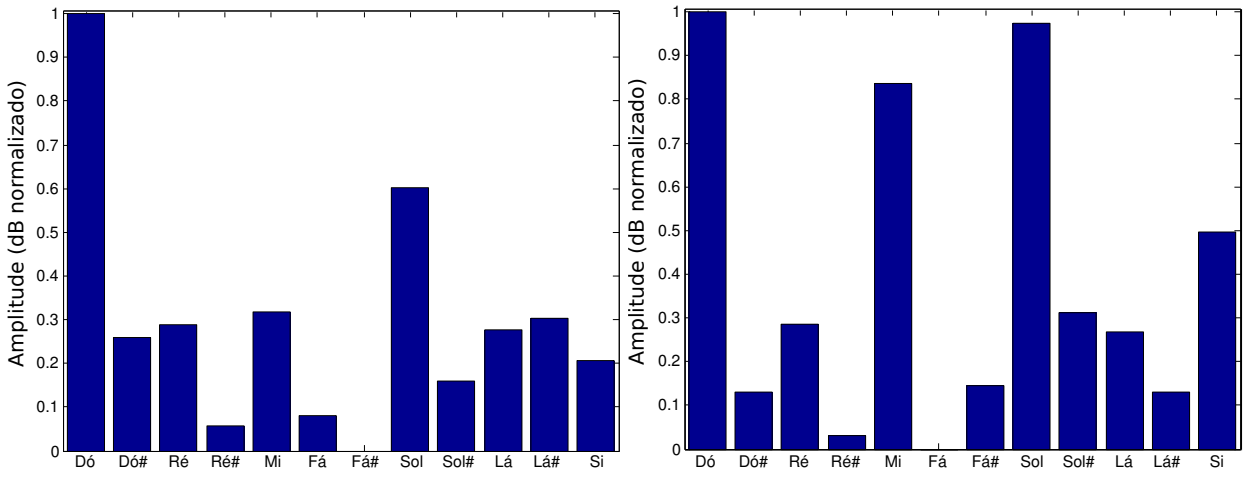
3.4.1 Modelos de acordes

Modelos de acordes podem ser definidos utilizando-se teoria musical e análise espectral. Os modelos mais simples representam as notas utilizando apenas os seus harmônicos fundamentais. A nota dó, por exemplo, seria representada pelo vetor-modelo da Figura 3.12(a). Consequentemente, ao unirmos três notas para formar o acorde de C (dó maior), teríamos o modelo da Figura 3.12(b).

Essa foi a representação utilizada no trabalho pioneiro de Fujishima [1], bem como no trabalho de Stark e Plumbley [2]; no entanto, importantes informações espectrais estão sendo negligenciadas neste modelo. A falta dos demais harmônicos é uma limitação forte ao sucesso da classificação.

Vetores croma extraídos de sinais reais mostram que, para uma nota simples, outros harmônicos são bem pronunciados, em especial o terceiro harmônico. A Figura 3.13(a) mostra o vetor gerado a partir de uma nota dó tocada em um violão. Note a presença forte do terceiro harmônico (Sol). Um acorde de C (dó maior) tocado no mesmo instrumento produziu o vetor croma da Figura 3.13(b). Percebe-se que estes são bastante diferentes dos modelos simples da Figura 3.12.

Tentativas de se incluir mais harmônicos na construção dos vetores-modelo foram feitas por outros autores. Não fica claro, porém, o porquê da escolha de alguns harmônicos específicos, nem a



(a) Nota Dó

(b) Acorde C

Figura 3.13: Vetores extraídos de sons reais

maneira como foram calculados os seus valores. Instrumentos diferentes produzem harmônicos em quantidades e intensidades diferentes, o que inviabiliza uma decisão exata. Há variações até mesmo entre regiões da escala de um mesmo instrumento. Qualquer escolha que se faça será arbitrária.

A partir deste problema surgiu a ideia de se utilizarem algoritmos de otimização para construir melhores modelos. A seção 3.5 descreve esse processo. Para fins de comparação, também foram utilizados modelos contendo apenas os harmônicos fundamentais e modelos contendo quantidades definidas de outros harmônicos, como será visto nos testes do capítulo 4.

Todos os 144 modelos podem ser criados a partir de um único vetor-modelo. Este vetor deve ser uma nota, com seus 12 parâmetros correspondendo aos graus da escala cromática. Definindo a nota Dó como modelo inicial, as demais notas conterão os mesmos valores, deslocados à direita em tantas posições quanto for a distância à nota Dó na escala. Esse processo é exemplificado na Figura 3.14.

Os acordes podem então ser construídos a partir dos modelos das notas. Para isso, linearizam-se as amplitudes dos modelos das notas, que estão normalizadas e em dB, somam-se as notas equivalentes a cada acorde e converte-se novamente para dB. Ainda, para que os valores permaneçam entre zero e um, realiza-se um dos ajustes abordados na seção 3.3.6, normalizando o vetor em seguida. Como exemplo, o acorde de Sol maior (G), formado pelas notas Sol, Si e Ré, será dado pela expressão:

$$\boldsymbol{G} = 20 \cdot \log_{10}(10^{\frac{\boldsymbol{Sol}}{20}} + 10^{\frac{\boldsymbol{Si}}{20}} + 10^{\frac{\boldsymbol{Re}}{20}}) \tag{3.7}$$

Lembrando que esse valor deverá ainda sofrer ajuste de média ou de mínimo, conforme escolha, e normalização.

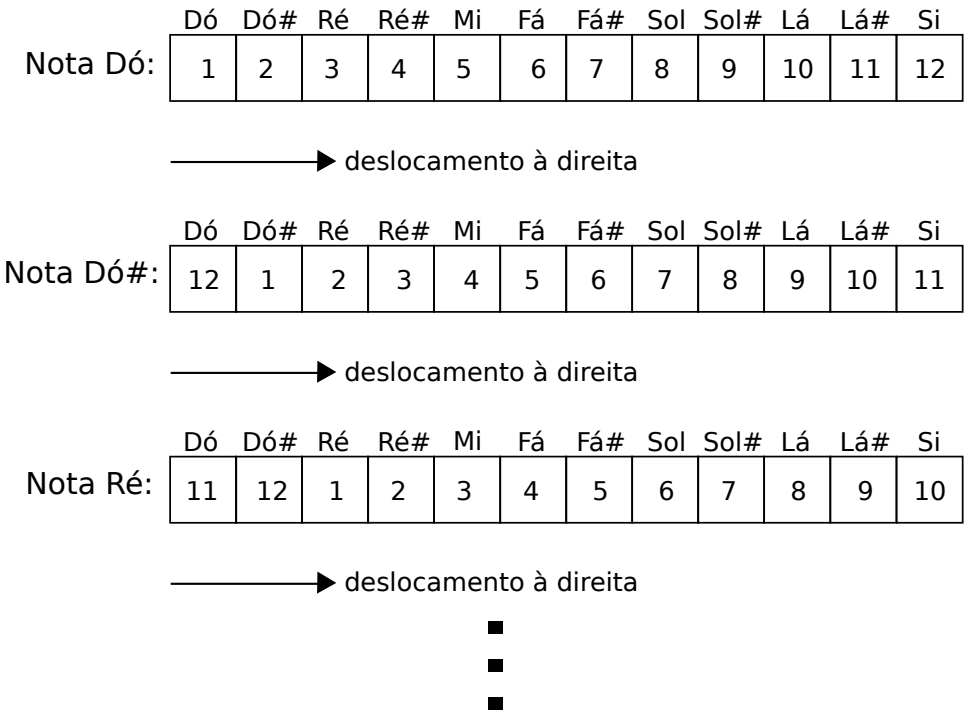


Figura 3.14: Construção dos demais modelos de notas a partir da nota Dó

3.4.2 Calculando os resultados

A comparação entre vetor e modelos deve se dar através de alguma regra definida. Uma das possibilidades é o uso da distância euclideana, aproveitando o fato de que quanto menor a distância maior a similaridade entre vetor e modelo. Essa foi a abordagem de Fujishima [1]. Outras medidas de similaridade também são comuns na literatura, como o produto escalar, utilizado por Stark e Plumbley [2]. A comparação por distância euclideana foi escolhida por atender bem aos propósitos do trabalho e por ser relativamente simples, mas nada impedia que o produto escalar ou outras medidas fossem utilizadas.

Através da equação 3.8, é calculada a distância euclideana do vetor croma a cada um dos modelos de acordes.

$$\boldsymbol{D}(k) = \sqrt{\sum_{n=1}^{12}(\boldsymbol{C}(n) - \boldsymbol{M}_k(n))^2}, \text{ para k = 1, 2, ..., Q} \tag{3.8}$$

Onde:

C é o vetor croma;

$\mathbf{M}_k$ é um vetor-modelo;

Q é o número total de modelos, e

D é um vetor contendo a distância calculada para cada modelo.

Como os vetores estão sempre normalizados, o máximo valor que $\boldsymbol{D}(k)$ pode assumir é $\sqrt{12}$.

Levando isso em consideração, calculamos a "proximidade" a um modelo como cada um dos valores $\boldsymbol{P}(k)$ dados pela equação:

$$\boldsymbol{P}(k) = 1 - \frac{\boldsymbol{D}(k)}{\sqrt{12}}, \text{ para k = 1, 2, ..., Q} \tag{3.9}$$

Finalmente, a saída do sistema será o acorde responsável pelo maior valor de $\boldsymbol{P}(k)$.

Seria possível utilizar apenas o somatório da equação 3.8, eliminando a operação de raiz, já que ela é monotônica, ou seja, não altera as relações de igualdade e de desigualdade entre os resultados. Isto economizaria tempo de processamento. No entanto, considerando que os resultados obtidos foram suficientemente rápidos, a equação original da distância euclideana foi mantida.

3.5 Otimização de parâmetros

Os modelos de notas e acordes utilizados na classificação são muito importantes. É impossível defini-los analiticamente, pois cada instrumento possui peculiaridades harmônicas distintas. Optou-se, assim, pelo uso de otimização.

A solução de otimização escolhida foi um algoritmo genético de evolução "egoísta", ou seja, não há troca de genes entre as soluções, apenas mutações. Dos 396 arquivos de áudio disponíveis, foi utilizada a metade (198) para a otimização em si. Destes, setenta porcento foram separados aleatoriamente para o uso na evolução das soluções, enquanto os trinta porcento restantes formaram um grupo de validação. A outra metade dos arquivos de áudio foi separada para a fase de testes, que será descrita na seção 4.2.

3.5.1 Evolução

Define-se uma população inicial de 100 indivíduos, em que cada um deles é um vetor de doze posições, correspondendo a um possível modelo para a nota Dó. Eles são utilizados para construir os demais modelos de notas e acordes, os quais são testados no conjunto de arquivos de treinamento. Os indivíduos são então classificados por desempenho, e são escolhidos os que obtiveram os 10 melhores resultados. Estes serão utilizados para gerar a população seguinte, enquanto os demais são descartados.

Os genes são cada uma das doze posições que formam o vetor do indivíduo. Os dez vetores mais bem sucedidos são mantidos na geração seguinte, e dão origem aos 90 restantes, que são resultado de mutações em seus genes. Para gerar um indivíduo, primeiramente se escolhe um dos vencedores, de maneira aleatória. Então, cada gene pode ou não mutar, com a probabilidade de mutação fixada em 1/5. Como não queremos repetições dos indivíduos bem-sucedidos (que foram mantidos nesta geração), eliminamos a possibilidade de não ocorrer mutação alguma. A probabilidade de que ocorra pelo menos uma mutação será, portanto, igual a um. Considerando esta condição para reescalonar a distribuição binomial, chega-se ao gráfico da Figura 3.15.

Durante a mutação, o gene pode ter seu valor atual somado ou subtraído de um valor aleatório, contido dentro de um intervalo pré-definido. Este intervalo começa com um valor de 0,5 e decresce

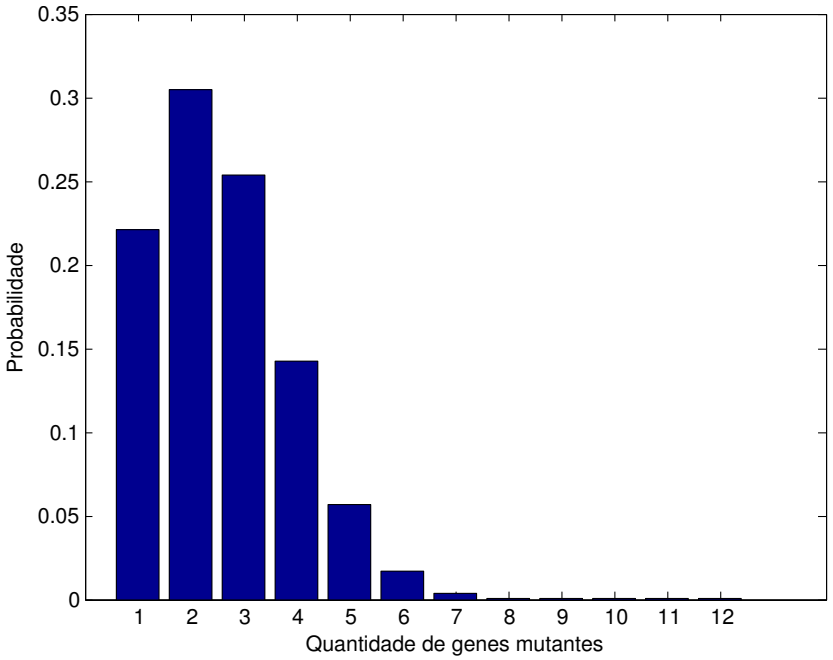


Figura 3.15: Probabilidade de ocorrência de diferentes quantidades de genes mutantes

linearmente a cada nova geração, até um mínimo de 0,01 na 50ª geração, a última permitida. Caso o valor do gene, após a mutação, se torne negativo, ele é automaticamente levado para zero. Analogamente, se o valor ultrapassar a unidade, será considerado igual a um.

Não se permite a geração de indivíduos iguais. Caso o indivíduo gerado já exista, ele é descartado, e a mutação é então repetida. Ao final, teremos os 10 indivíduos mais bem-sucedidos da geração anterior e 90 novos, resultantes de mutações dos 10 primeiros.

3.5.2 Validação

Em cada ciclo, após a escolha das 10 melhores soluções, elas são testadas no conjunto de arquivos de validação. A qualidade dos resultados é utilizada então como critério de parada. Para isso, toma-se a média dos resultados e o seu maior valor. Caso ambos se tornem piores por três ciclos consecutivos, os conjunto de dez vencedores da população anterior a estes ciclos é considerado a melhor solução.

Deste modo evita-se o overfitting dos indivíduos, escolhendo um grupo de vetores-modelo mais generalista.

3.5.3 Resultados

O algoritmo foi executado com dois pontos de inicialização diferentes. Primeiramente, começou-se com uma população cujos indivíduos possuíam genes aleatórios. Em outra tentativa, iniciou-se o algoritmo com todos os indivíduos iguais, com os valores mostrados na Figura 3.16.

Os dez melhores resultados ao final de cada uma das tentativas foram testados no conjunto de arquivos separado para essa função. Os testes foram realizados utilizando-se a interface descrita

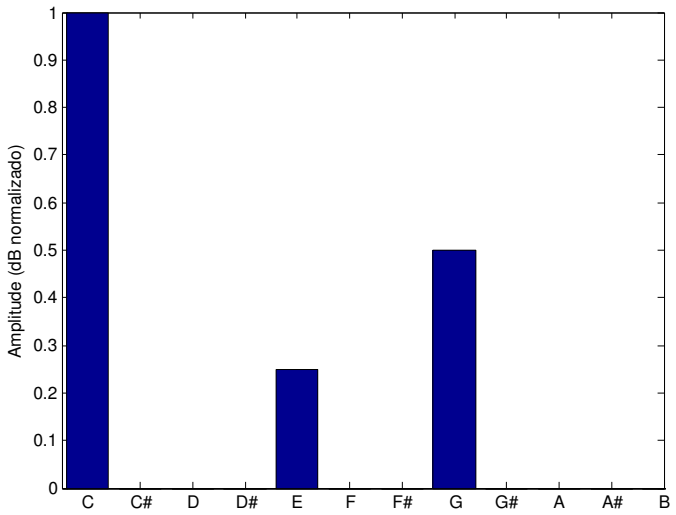
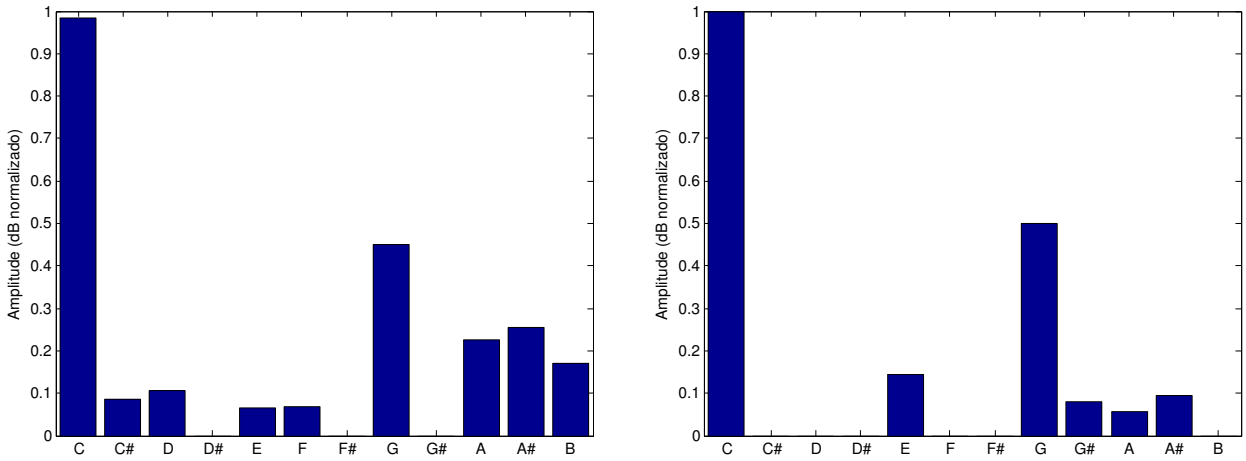


Figura 3.16: Indivíduo utilizado em uma das inicializações



(a) Inicialização aleatória

(b) Inicialização pré-determinada

Figura 3.17: Vetores-modelo resultantes do processo de otimização

no capítulo 4. O vetor que resultou nos melhores resultados em cada um dos dois grupos foi considerado como solução de otimização. Temos, assim, os dois vetores da Figura 3.17, gerados a partir de inicialização aleatória e de inicialização pré-determinada, respectivamente.

Percebe-se a presença dos componentes esperados para um modelo de nota Dó. O harmônico fundamental manteve-se como o maior componente, e a presença dos terceiro e quinto harmônicos é refletida nos valores das posições Mi e Sol. No entanto, não era esperada a presença recorrente de valores em Lá e Lá#, nem a existência de forte componente Si no primeiro modelo. Para tentar entender porque isso acontece, vamos partir de uma nota musical qualquer, digamos, o Dó1, cuja frequência fundamental é de 32,70Hz. Verificando a posição de seus harmônicos em relação à Tabela 3.1 e considerando as faixas de frequência das equações 3.5 e 3.6, geramos a Tabela 3.2.

Estes dados nos mostram, então, as prováveis origens de alguns dos valores encontrados. O Lá# pode ser encontrado nos 7º e 14º harmônicos. A presença de Ré no 9º harmônico deve estar relacionada com a existência deste componente em um dos resultados. Já a existência do

Harmônico	Frequência (Hz)	Correspondência
1º	32,70	Dó
2º	65,40	Dó
3º	98,10	Sol
4º	130,8	Dó
5º	163,5	Mi
6º	196,2	Sol
7º	228,9	Lá#
8º	261,6	Dó
9º	294,3	Ré
10º	327,0	Mi
11º	359,7	Entre Fá e Fá#
12º	392,4	Sol
13º	425,1	Entre Sol# e Lá
14º	457,8	Lá#
15º	490,5	Si

Tabela 3.2: Harmônicos de uma nota Dó

componente Si poderia ser explicada pelo 15º harmônico. Fica ainda, porém, sem explicação a forte presença do componente Lá nos vetores otimizados, e também a presença de Dó# e Fá no primeiro modelo. Pequenas desafinações poderiam levar alguns dos harmônicos a entrarem em regiões típicas de notas Lá, Do# e Fá. Outra possibilidade é a de que um ruído de gravação tenha se repetido em muitos dos arquivos, inserindo componentes espectrais específicas. Em trabalhos futuros, pode-se utilizar um novo conjunto de arquivos, gravado em outras condições, para averiguar essa hipótese.

Capítulo 4

Testes e Aplicações

4.1 Interface gráfica

Para que fosse possível alterar os parâmetros e verificar o comportamento do sistema de maneira rápida e cômoda, foi desenvolvida uma interface gráfica de usuário. Ela possui duas janelas, uma para seleção de entradas e parâmetros e outra para a exibição dos resultados. A interface foi editada com a ferramenta Guide, do Matlab.

4.1.1 Janela de seleção de entradas e parâmetros

A primeira janela da interface pode ser vista na Figura 4.1 e possui diversos campos, que são descritos a seguir:

- Entradas
  - Permite escolher dois tipos de entrada, a saber: arquivos de áudio ou captura do microfone. Permite ainda utilizar apenas a metade central dos arquivos de áudio, excluindo trechos do começo e do fim. Esta opção é útil para diminuir o tamanho de arquivos com longos silêncios iniciais e finais.
- Acordes e notas
  - Pode-se restringir os modelos a serem considerados na classificação. É possível, por exemplo, testar o desempenho do sistema ao classificar apenas tríades ou apenas acordes maiores e menores.
- Análise espectral
  - O módulo da FFT ou o seu valor em dB podem ser escolhidos para serem utilizados na etapa de construção dos vetores croma.
  - Também se pode definir o alcance da tabela de frequências de referência, efetivamente truncando a Tabela 3.1 nos limites que o usuário julgar adequados.

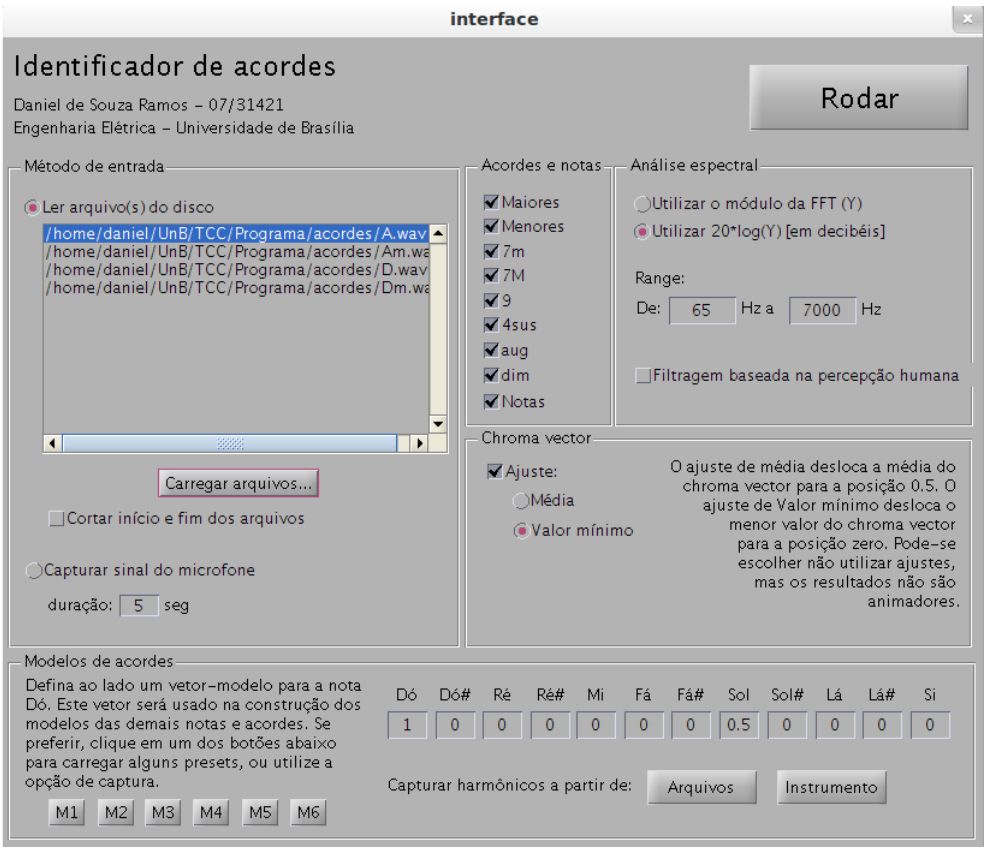


Figura 4.1: Janela para seleção de entradas e parâmetros

  - Por fim, é dada a opção de se utilizar o filtro baseado na percepção humana descrito na seção 3.3.1.2.

- Ajustes

  - Escolhe-se o tipo de ajuste que se deseja usar na construção dos vetores croma. Pode-se ajustar a média para o valor 0,5 ou ajustar o menor valor para zero. Também é possível não utilizar ajuste algum.

- Modelos

  - Aqui nos é dada a opção de criar os modelos de notas e acordes que serão comparados aos vetores croma. Isso é feito editando-se os valores do modelo da nota Dó. As demais notas e acordes são gerados a partir deste vetor, como foi explicado na seção 3.4.1.
  - Pode-se gerar o modelo da nota Dó a partir de um arquivo de áudio ou a partir de captura de som do microfone. Neste caso, as configurações utilizadas para gerar o modelo serão as mesmas que estiverem definidas nos campos "Análise espectral" e "Ajustes".
  - São disponibilizados cinco modelos pré-definidos para a nota Dó. São os vetores de $\boldsymbol{M1}$ a $\boldsymbol{M5}$ representados na Tabela 4.1. Perceba que o vetor $\boldsymbol{M1}$ possui apenas o harmônico fundamental. Já o modelo $\boldsymbol{M2}$ considera uma quantidade arbitrária de terceiros harmônicos. Partindo deste, um valor arbitrário é acrescentado para o quinto harmônico no vetor $\boldsymbol{M3}$. Por fim, $\boldsymbol{M4}$ e $\boldsymbol{M5}$ são vetores que foram gerados na fase de

	Dó	**Dó#**	**Ré**	**Ré#**	**Mi**	**Fá**	**Fá#**	**Sol**	**Sol#**	**Lá**	**Lá#**	**Si**
***M*1**	1	0	0	0	0	0	0	0	0	0	0	0
***M*2**	1	0	0	0	0	0	0	0,5	0	0	0	0
***M*3**	1	0	0	0	0,25	0	0	0,5	0	0	0	0
***M*4**	0.98	0.09	0,11	0	0,06	0,07	0	0,45	0	0,23	0,26	0,17
***M*5**	1	0	0	0	0,14	0	0	0,5	0,08	0,06	0,09	0

Tabela 4.1: Vetores-modelo pré-definidos

otimização. O primeiro foi gerado com inicialização aleatória, enquanto que o segundo utilizou o vetor ***M*3** como inicialização, como visto na seção 3.5.

4.1.2 Janela de resultados

A janela de resultados altera sua aparência conforme a escolha do método de entrada. Quando a entrada são arquivos de áudio, a janela tem a aparência da Figura 4.2(a).

Temos aqui um relatório completo. A tabela à esquerda mostra os acordes e notas esperados, baseando-se no nome dos arquivos. A tabela central contém os resultados para cada arquivo. Vêem-se os três modelos que mais se aproximaram, bem como o modelo mais distante, com suas respectivas certezas.

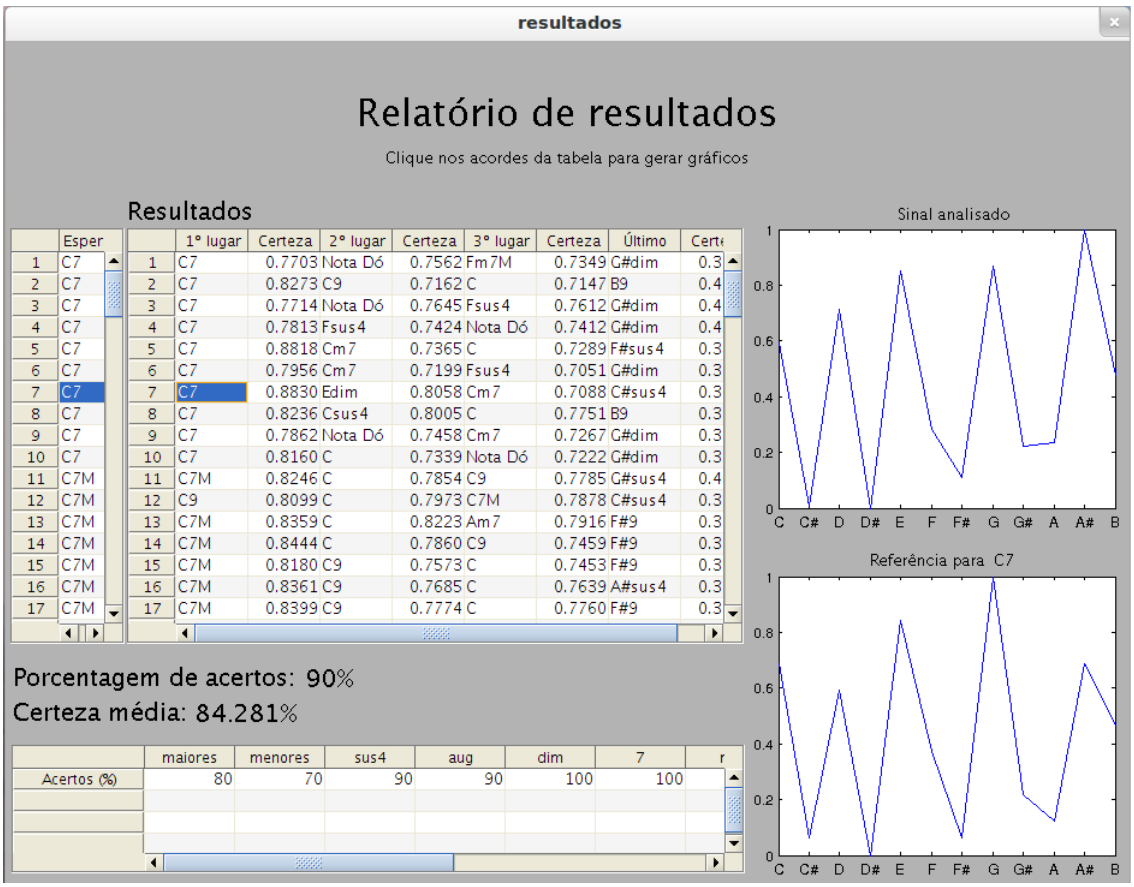
Clicar em um dos valores da tabela de resultados faz com que sejam desenhados dois gráficos à direita. O gráfico superior mostra os valores do vetor croma de entrada, enquanto o inferior exibe o vetor-modelo selecionado. Assim, tem-se uma boa visão da semelhança ou da diferença entre modelos e entradas.

Abaixo da tabela central temos os percentuais globais de acerto e de certeza. Além disso, na tabela inferior são exibidos os percentuais de acerto para cada categoria de entrada. Desta maneira, pode ser analisada a eficiência do sistema na classificação das notas e dos diversos tipos de acordes.

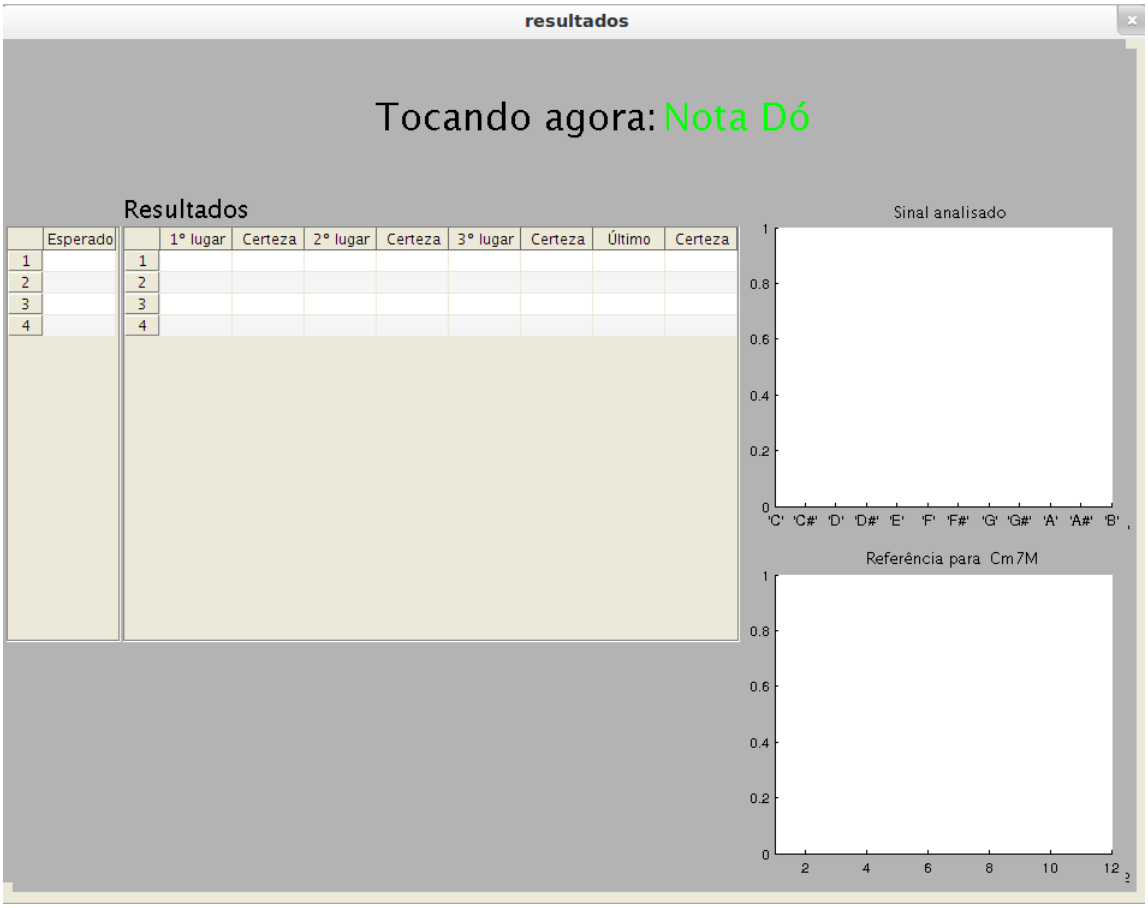
Quando o método de entrada escolhido é a captura de áudio do microfone, porém, teremos uma janela semelhante à vista na Figura 4.2(b). Quando a janela se abre, há uma contagem regressiva e é iniciada a captura do áudio. Os acordes e notas são identificados e exibidos em texto na parte superior da janela conforme vão sendo tocados. Ao final, a lista com todas as classificações realizadas é exibida na tabela central. Gráficos também podem ser traçados clicando-se sobre os valores da tabela.

4.2 Resultados

Metade dos arquivos de entrada existentes foram utilizados na etapa de otimização, como visto na seção 3.5. Não se devem utilizar os mesmos arquivos para os testes, pois um possível *overfitting* poderia trazer resultados artificialmente melhores. Para que a comparação entre resultados pudesse



(a) Janela de resultados para arquivos



(b) Janela de resultados durante a captura de áudio de microfone

Figura 4.2: Janelas de Resultados

Faixa de frequências	Filtragem	Ajuste	Modelo	Acertos
20Hz a 20kHz	Não	Mínimo	$\boldsymbol{M1}$	48,99%
20Hz a 20kHz	Sim	Mínimo	$\boldsymbol{M1}$	47,98%
65Hz a 7kHz	Não	Média	$\boldsymbol{M1}$	60,10%
65Hz a 7kHz	Não	Mínimo	$\boldsymbol{M1}$	66,67%
65Hz a 7kHz	Sim	Mínimo	$\boldsymbol{M1}$	67,17%
65Hz a 7kHz	Não	Mínimo	$\boldsymbol{M2}$	75,76%
65Hz a 7kHz	Não	Mínimo	$\boldsymbol{M3}$	83,33%

Tabela 4.2: Resultados de testes para várias configurações

Faixa de frequências	Filtragem	Ajuste	Modelo	Acertos
20Hz a 20kHz	Não	Mínimo	$\boldsymbol{M1}$	71,01%
65Hz a 7kHz	Não	Mínimo	$\boldsymbol{M1}$	71,01%

Tabela 4.3: Influência do corte de frequências em sons sintetizados

ser coerente, todos os parâmetros (mesmo os que não passaram por otimização) foram testados com apenas a metade restante dos arquivos.

Testaram-se configurações diversas de entrada. Os testes com modelos não otimizados mais relevantes estão mostrados na Tabela 4.2.

Percebem-se dois fatores principais de melhoria: a restrição da faixa de frequências e o tipo de modelo utilizado. O uso do ajuste de média nos deu resultados um pouco piores do que o uso do ajuste de mínimo e foi descartado.

Aparentemente, a principal contribuição do corte das frequências inferiores e superiores foi a eliminação de ruído. Um meio de verificar essa hipótese é realizar os testes utilizando apenas sons sintetizados, que não possuem ruído de gravação ou ambiente. Estes testes foram feitos e a Tabela 4.3 mostra os resultados. Nela vemos exatamente o que foi previsto: a restrição da faixa de frequências não traz melhoria alguma quando não há ruído.

É interessante também notar a melhoria trazida pelos modelos que possuem um maior número de componentes. O vetor-modelo $\boldsymbol{M3}$, responsável pela inclusão de valores arbitrários correspondentes ao terceiro e ao quinto harmônicos, melhorou em mais de 16% a resposta em relação ao vetor $\boldsymbol{M1}$, que considera apenas as fundamentais. Esta grande variação foi a principal motivação para que se realizasse a otimização dos modelos.

Ao contrário do esperado, o uso da filtragem baseada na percepção humana não teve influência significativa nos resultados. A Tabela 4.2 mostra o uso do filtro em apenas duas configurações diferentes (ambas com ajuste de mínimo e com o modelo $\boldsymbol{M1}$), mas os testes em outras configurações também deram resultados muito semelhantes aos obtidos sem o filtro. A expectativa inicial era de que a etapa de filtragem eliminasse a necessidade de realizar o corte nas frequências. Como a expectativa não se concretizou, e como não houve outra vantagem significativa, o uso do filtro foi descartado.

Faixa de frequências	Filtragem	Ajuste	Modelo	Acertos
65Hz a 7kHz	Não	Mínimo	$\boldsymbol{M4}$	86,36%
65Hz a 7kHz	Não	Mínimo	$\boldsymbol{M5}$	86,87%

Tabela 4.4: Resultados de testes com os modelos otimizados

	Maior	Menor	sus4	aug	dim	7	m7	7M	m7M	9	m9	Nota
$\boldsymbol{M4}$	75	64,7	88,2	93,8	93,8	100	76,5	87,5	94,1	87,5	88,2	88,2
$\boldsymbol{M5}$	87,5	58,8	88,2	93,8	93,8	100	88,2	75	94,1	100	88,2	76,5

Tabela 4.5: Taxas de acerto por categoria de entrada (em %)

A Tabela 4.4 mostra as configurações utilizadas e os resultados conseguidos com os vetores-modelo otimizados $\boldsymbol{M4}$ e $\boldsymbol{M5}$. Os resultados claramente melhoraram. No processo de otimização do vetor $\boldsymbol{M4}$, a população inicial foi inicializada com indivíduos aleatórios. Seu resultado foi ligeiramente pior, talvez não significativamente, do obtido como vetor $\boldsymbol{M5}$, cuja inicialização foi baseada no vetor-modelo $\boldsymbol{M3}$. Dada a semelhança entre as porcentagens de acerto, pode-se dizer que há vantagem em se inicializar a população com valores reconhecidamente bons, pois o vetor-modelo é obtido muito mais rapidamente.

O número de acertos pode ser dividido por tipo de entrada. Vemos na Tabela 4.5 a porcentagem de acertos dos dois modelos otimizados para cada categoria de acorde, bem como para notas. Fica claro que os resultados não são homogêneos. A princípio, a quantidade de notas presentes no acorde não foi determinante; há tríades e tétrades com altas taxas de acerto, e mesmo notas simples obtiveram uma boa taxa de acerto com o modelo $\boldsymbol{M4}$. No entanto, determinadas entradas parecem trazer problemas em cada um dos dois casos. A resposta do modelo $\boldsymbol{M4}$ é particularmente ruim para acordes maiores e para acordes menores com sétima. Já o modelo $\boldsymbol{M5}$ se comporta mal diante de acordes maiores com sétima maior e diante de notas simples. Acordes menores, por sua vez, tiveram taxas baixas com ambos os modelos, embora um pouco melhores com $\boldsymbol{M4}$. Pode-se supor que a configuração específica de cada modelo implica em maior facilidade para se distinguir certos acordes do que outros.

Com isso em mente, montaram-se as Tabelas 4.6 e 4.7, que são matrizes de confusão para os dois modelos otimizados. Ressalta-se que erros dentro de uma mesma categoria (como um Cm classificado como Em) não são perceptíveis por estas tabelas. Apenas são evidentes os erros em que um acorde de uma categoria é classificado em outra (como um Cm classificado como Cm7M).

Na Tabelas 4.6 e 4.7 vê-se que há um padrão em muitos dos erros. Vários acordes menores foram classificados como menores com nona ou menores com sétima maior. Da mesma maneira, acordes maiores foram percebidos como maiores com sétima maior ou maiores com sétima. Alguns acordes que deveriam ter sido reconhecidos como maiores com sétima maior foram classificados como apenas maiores. Dada a semelhança entre as tríades maiores e menores e suas tétrades correspondentes, essas ocorrências são compreensíveis. Percebe-se que a maior parte dos erros aconteceu entre categorias harmonicamente próximas.

Pode-se inferir, a partir destes resultados, que os dois modelos otimizados convergiram para

		Resultados esperados											
		Maior	Menor	sus4	aug	dim	7	m7	7M	m7M	9	m9	Nota
Resultados obtidos	Maior	12							2		1		1
	Menor		11			1		1					
	sus4			15									
	aug				15								
	dim					15							
	7	1	1				16	1					
	m7							13					
	7M	3						1	14		1		
	m7M		2		1					16		2	
	9										14		
	m9		3	1						1		15	1
	Nota			1				1					15

Tabela 4.6: Matriz de confusão dos resultados do modelo ***M4***

		Resultados esperados											
		Maior	Menor	sus4	aug	dim	7	m7	7M	m7M	9	m9	Nota
Resultados obtidos	Maior	14	1						2				1
	Menor		10			1		1	1			1	
	sus4			15									
	aug				15								
	dim					15							
	7	2	1				16	1					2
	m7							15				1	
	7M								12				1
	m7M		1		1					16			
	9			2					1		16		
	m9		4							1		15	
	Nota												13

Tabela 4.7: Matriz de confusão dos resultados do modelo ***M5***

mínimos locais diferentes, implicando em soluções de compromisso ligeiramente distintas. Ambos mantém altas taxas de acerto globais, mas suas configurações específicas os levam a ter maior dificuldade na distinção de algumas categorias de acordes.

É interessante notar que os resultados dos dois modelos são, até certo ponto, complementares. É possível pensar em um sistema que utilize ambos os modelos, paralelamente, para classificar o mesmo sinal. Tal sistema escolheria, ao final, a saída que resultasse no maior nível de certeza. Apesar do aumento do esforço computacional, essa técnica pode nos dar resultados mais precisos.

O desempenho do sistema em tempo real foi consistente com os resultados *offline*. Testes informais mostraram que altos índices de acerto podem ser obtidos, mesmo utilizando-se trechos de frações de segundo para a análise. No entanto, o resultados se tornam visivelmente piores quando os intervalos decrescem abaixo de 0,1 segundo. Outros fatores também influenciam a qualidade da classificação. Após algum tempo soando livremente, a diminuição da amplitude dos acordes e notas pode tornar o reconhecimento difícil para o sistema. Além disso, as transições entre acordes podem gerar resultados espúrios.

A adoção de algumas estratégias melhorou consideravelmente a performance *online*. Utilizou-se, primeiramente, um limite mínimo de certeza, estabelecido empiricamente, abaixo do qual o sistema não geraria resposta alguma. Além disso, fez-se com que os dois últimos resultados fossem sempre comparados, retornando o que gerasse maior certeza, desde que acima do limite mínimo. Assim, houve redução satisfatória do número de classificações espúrias, aproximando o desempenho do conseguido na classificação a partir de arquivos.

4.3 Aplicações

4.3.1 Jogo para aprendizado interativo

Foi criado um pequeno jogo, em Matlab, no qual o usuário aprende a executar músicas em violão ou guitarra. Uma representação das notas musicais em tablatura é exibida na tela e o jogador deve acompanhá-la, executando as notas e os acordes no tempo certo.

A tela inicial do jogo contém três opções de músicas. Ao clicar em alguma delas, o usuário é levado para a tela seguinte, ilustrada na Figura 4.3, onde ele irá tocar a tablatura. Conforme toca, é exibido um sinal, ao lado esquerdo, indicando se a nota ou acorde tocado corresponde ou não ao esperado. Ao final, uma tela com a porcentagem de acertos é exibida.

A melhor configuração para o processo de reconhecimento, descrita na seção 4.2, foi a escolhida para mover a aplicação. Várias tentativas de reconhecimento são feitas em cada subdivisão rítmica. Se é detectado um acerto, ele é mantido até o aparecimento da próxima nota ou acorde.

Cabe lembrar que o processo de reconhecimento utilizado não diferencia oitavas. Como se queria apenas demonstrar o trabalho feito, essa diferenciação não é necessária. Futuramente, caso se deseje continuar o desenvolvimento da aplicação, recomenda-se incluir, em paralelo, um algoritmo de detecção de pitch. Dessa maneira, notas de oitavas diferentes não contarão como acertos.

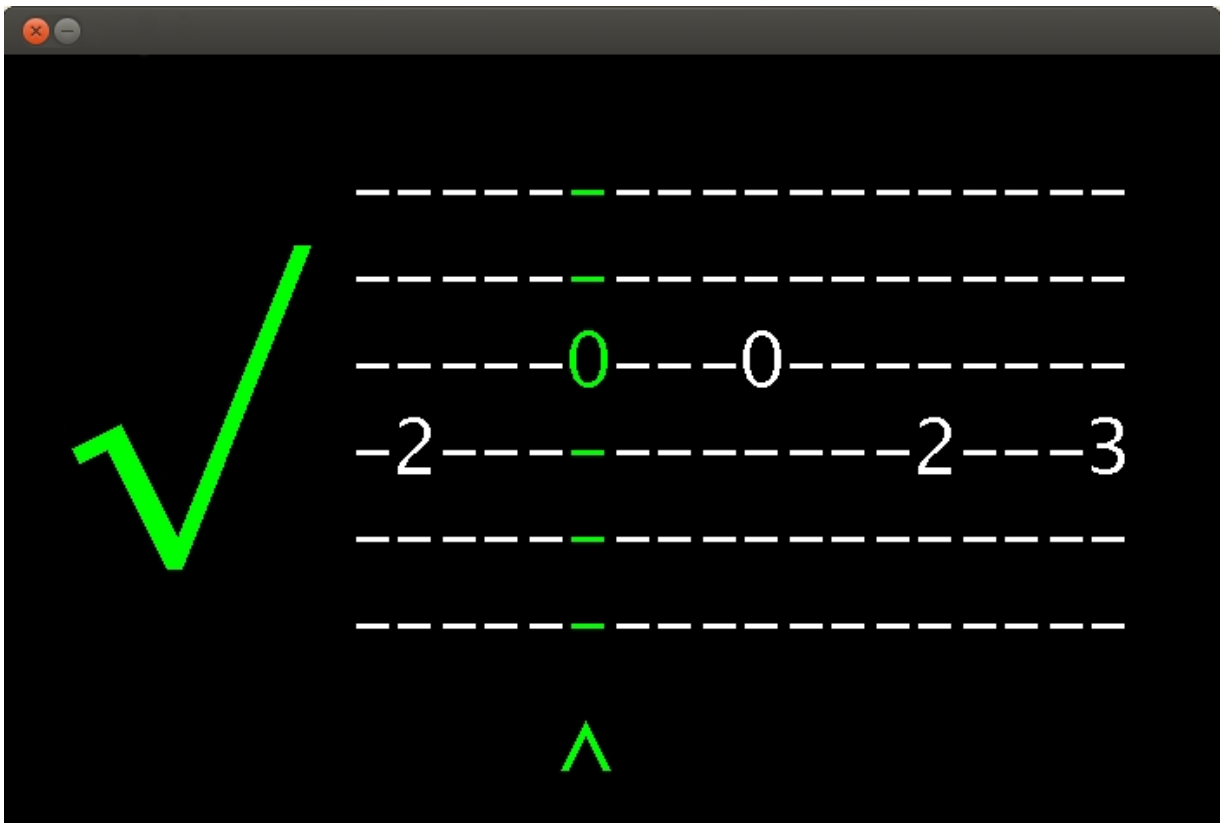


Figura 4.3: Tela principal do jogo desenvolvido

Muitas outras melhorias podem ser feitas no futuro. Uma aplicação de grande porte incluiria um grande número de músicas e também uma ferramenta para a edição das próprias tablaturas. Poderia-se exibir também partituras, o que abriria o leque de instrumentos de entrada. Outra ideia é o armazenamento de um histórico do usuário, de onde se poderiam extrair informações interessantes, como os erros mais frequentes.

4.3.2 Outras aplicações possíveis

4.3.2.1 Transcrição musical

É extremamente interessante a possibilidade de se transcrever uma composição automaticamente, gerando escrita musical a partir de sinais de áudio. O sistema construído, se complementado por estratégias de detecção de oitavas e pela determinação do tempo das notas, pode servir a tal fim. A ideia é que o compositor, após simplesmente executar a música, tenha em mãos uma partitura completa, que exija o mínimo de modificações, ou um arquivo MIDI com o qual possa trabalhar.

A transcrição automática pode auxiliar tanto o compositor quanto o aprendiz. É comum, na música popular, não se dispor da partitura da composição original, que pode nem sequer existir. Vários *websites* se dedicam à transcrição de "cifras" e tablaturas de canções populares, principalmente para violão e guitarra. Um software que gerasse tais transcrições a partir da análise de arquivos de áudio seria de grande ajuda. O desafio aqui é lidar com os múltiplos sons

de instrumentos diversos que seriam ouvidos ao mesmo tempo, inclusive sons percussivos que não contribuem para a harmonia.

4.3.2.2 Improvisação algorítmica

O reconhecimento da harmonia pode servir como base para a criação de um algoritmo que execute improvisos em tempo real. Durante uma performance musical, por exemplo, o software assim desenvolvido identificaria os acordes tocados pelos demais músicos e improvisaria com sons sintetizados, guiado por técnicas de composição algorítmica.

Aqui, novamente, a detecção de andamento é importante. Além disso, algoritmos de composição e de síntese relativamente rápidos devem ser utilizados, para que se mantenha a característica de tempo real.

4.3.2.3 Autômato para "virar a página"

Um problema que incomoda muitos músicos é a necessidade de se passar para a próxima página de uma partitura. Em geral, o próprio instrumentista precisa se distrair momentaneamente para virar a página, ou um auxiliar deve acompanhar atentamente a execução para fazê-lo no momento certo.

Com a popularização dos tablets e e-readers, tem-se hoje a possibilidade de armazenar e visualizar as partituras em meio digital. Porém, ainda é necessário algum comando de toque para que o aparelho exiba a página seguinte. Um software que fosse capaz de acompanhar a execução poderia realizar a mudança no momento certo, automaticamente. Embora o andamento seja o fator principal a ser acompanhado por tal programa, há vários problemas e dificuldades envolvidas na sua detecção. O reconhecimento da harmonia seria de grande auxílio na determinação do compasso atual, possibilitando a construção de um sistema mais robusto e preciso.

Capítulo 5

Conclusões

O trabalho resultou em um sistema funcional e eficiente, cuja estrutura e fundamentos podem ser utilizados em outras aplicações e projetos da área. Conhecimentos adquiridos ao longo do curso foram essenciais para a realização do projeto. Em particular, a familiaridade com conceitos teóricos ligados à análise em frequência, e com ferramentas práticas como a programação em Matlab, garantiram a fluência do trabalho e a liberdade para que fossem tomadas quaisquer decisões de projeto que se mostrassem vantajosas. Além disso, o estudo de assuntos relativamente novos, como vetores croma e algoritmos genéticos, foi interessante e produtivo.

A comparação com resultados de outros trabahos é difícil de ser realizada, devido à pluralidade de abordagens. A escolha das entradas utilizadas é feita de maneira muito diversa entre os artigos pesquisados. Na maioria das vezes, por exemplo, o grupo de entrada não inclui notas musicais. A variedade de acordes muitas vezes é pequena, mas, em alguns casos, é muito grande. Além disso, vários dos trabalhos da literatura possuem como foco principal a extração de informação a partir de gravações de peças musicais completas. Oudre et al. [10], por exemplo, obtiveram taxas de acerto de 71,1% considerando acordes maiores, menores e com sétima, mas realizaram seus testes utilizando gravações de música popular. Estas gravações possuem elementos percussivos que influenciam negativamente na qualidade dos resultados.

Uma abordagem mais semelhante foi realizada por Stark e Plumbley [2], que gravaram áudio de duas guitarras para os testes. Eles consideraram uma variedade de acordes semelhante à utilizada neste projeto, sem incluir, porém, notas simples. Uma diferença, contudo, se destaca: a restrição do alcance a apenas duas oitavas. Isto evita não só o ruído de frequências superiores e inferiores, como também elimina a necessidade de se lidar com a maior parte dos harmônicos, que são fontes importantes de erro. Os pesquisadores obtiveram taxas de acerto de 92,7%.

Apesar da impossibilidade de uma comparação direta, o desempenho obtido com o sistema em tempo real, e diante de condições pouco favoráveis, revelou utilidade. Uma variedade relativamente grande de entradas foi considerada, incluindo notas individuais. Sons naturais compreenderam dois terços do conjunto de testes, sendo que o restante foi composto por sons sintetizados bastante diversificados. Além disso, o alcance de frequências considerado foi de mais de seis oitavas, resultando em grande influência de ruído e de harmônicos no resultado. Ainda assim, foram atingidas taxas de acerto de cerca de 86%, uma porcentagem suficientemente alta para que, com poucas melhorias,

o sistema possa ser utilizado em implementações práticas.

Além disso, a análise é realizada de maneira suficientemente rápida para ser utilizada em aplicações de tempo real. Esta característica foi testada e explorada no pequeno jogo desenvolvido, descrito na seção 4.3.1. A velocidade de reação do sistema, inclusive, foi utilizada para incrementar os resultados do reconhecimento, utilizando uma divisão do compasso musical como limite entre a detecção atual e a detecção seguinte. Há, ainda, margem para outras técnicas, como realização de diversas identificações em um intervalo pequeno de tempo, para que se escolha a que possui maior nível de certeza. Esta alternativa foi utilizada na plataforma de testes e apresentou, também, bons resultados.

O uso de algoritmos genéticos para a otimização dos modelos se mostrou bastante recompensador. Os resultados se tornaram substancialmente melhores após a otimização, evidenciando a relativa arbitrariedade existente na escolha de modelos simplificados de notas e acordes. O algoritmo desenvolvido possui a desvantagem de exigir grande esforço computacional, o que é um problema comum entre algoritmos genéticos. Muitas horas de processamento foram gastas em cada execução do algoritmo, o que limitou a quantidade de resultados apresentados. Ressalta-se, no entanto, que essa não é uma grande limitação, visto que, em teoria, o algoritmo necessita ser executado apenas uma vez, e de maneira separada do sistema, não influenciando em nenhum momento sua performance.

Muito conhecimento foi absorvido e aplicado, tanto durante a elaboração do projeto em si quanto durante a redação do relatório. A descrição do funcionamento da transformada discreta de Fourier, realizada no capítulo 2, e de como ela pode ser implementada de maneira eficiente através da transformada rápida de Fourier, trouxe uma melhor visão sobre um assunto que, apesar de importante, havia sido estudado de forma incompleta durante o curso de Engenharia Elétrica. Além disso, a pesquisa inicial e a implementação de um algoritmo genético revelaram que, longe de mero objeto acadêmico, essa classe de algoritmos possui aplicações efetivas e interessantes em problemas práticos de engenharia.

5.1 Perspectivas futuras

O trabalho realizado pode ser estendido de algumas maneiras. Pode-se utilizar gravações de sons naturais de um maior número de instrumentos, por exemplo, para construir novos modelos de acordes, possivelmente mais generalistas. O ideal é que se realizem essas gravações com diferentes sistemas de captura e em diferentes ambientes, de modo a se evitar a inserção de ruído enviesado, o que não foi possível durante este trabalho.

O projeto não tinha como finalidade a distribuição e, assim, foi construído em uma plataforma de programação científica. Uma possibilidade de trabalho futuro, portanto, é a transcrição da estrutura para uma linguagem que permita a distribuição, como C++. Assim, será possível o uso do sistema para o desenvolvimento de aplicações voltadas para o usuário comum.

O reconhecimento de características musicais é um tema de pesquisa ainda pouco explorado, com diversas possibilidades de aplicações que podem interessar a músicos e apreciadores de música.

Algumas possibilidades de sistemas baseados neste projeto foram descritos na seção 4.3, incluindo o pequeno jogo desenvolvido, mas espera-se que novas ideias e formas de uso criativo da tecnologia de reconhecimento de acordes surjam conforme resultados de áreas conexas permitam que se obtenha um conjunto mais completo de informações musicais a partir de sinais de áudio.

REFERÊNCIAS BIBLIOGRÁFICAS

[1] FUJISHIMA, T. Realtime chord recognition of musical sound: a system using common lisp music. *International Computer Music Conference (ICMC)*, p. 464–467, 1999.

[2] STARK, A. M.; PLUMBLEY, M. D. Real-time chord recognition for live performance. *International Computer Music Conference*, p. 585–593, 2009.

[3] OUDRE, L.; GRENIER, Y.; FéVOTTE, C. Chord recognition by fitting rescaled chroma vectors to chord templates. *IEEE Transactions on Audio, Speech and Language Processing*, v. 19, p. 2222–2233, 2011.

[4] HAYKIN, S.; VEEN, B. V. *Sinais e Sistemas*. Tradução de José Carlos Barbosa dos Santos. Porto Alegre: Bookman, 2001.

[5] LEE, K. Automatic chord recognition from audio using enhanced pitch class profile. *International Computer Music Conference (ICMC)*, p. 306–313, 2006.

[6] BELLO, J. P.; PICKENS, J. A robust mid-level representation for harmonic content in music signals. *International Society for Music Information Retrieval (ISMIR)*, p. 304–311, 2005.

[7] INTERNATIONAL ORGANIZATION FOR STANDARDIZATION. *ISO 226:2003, Acoustics - Normal equal-loudness-level contours.*

[8] SUZUKI, Y.; TAKESHIMA, H. Equal-loudness-level contours for pure tones. *Journal of the Acoustical Society of America*, v. 116, p. 918–933, 2004.

[9] CHO, T.; BELLO, J. P. Real-time implementation of hmm-based chord estimation in musical audio. *International Computer Music Conference (ICMC)*, p. 117–120, 2009.

[10] OUDRE, L.; GRENIER, Y.; FéVOTTE, C. Chord recognition using measures of fit, chord templates and filtering methods. *IEEE Workshop on Applications of Signal Processing to Audio and Acoustics*, p. 9–12, 2009.