EP1 MAC300 - Relatório

Fernando Omar Aluani (NUSP: 6797226)

September 21, 2012

1 Modelando Som com Série de Fourier

Podemos usar Séries de Fourier para modelar um som (uma onda sonora, no domínio do tempo) no domínio da frequência, o que ajuda muito para analisar as propriedades do som (como harmônicos e notas).

O som originalmente está em tempo-contínuo, porém como não podemos analisá-lo dessa maneira, então primeiramente amostramos o som para que tenhamos a representação do som em tempo-discreto. Então dividimos o som (em tempo-discreto) em intervalos constantes, para analisarmos cada intervalo aplicando a Transformada Discreta de Fourier (DFT) em tempo-discreto.

A DFT devolve a representação desse intervalo do som no domínio da frequência - na forma de um conjunto de ondas senoidais que somadas resultam na onda original do som, e a frequência de cada uma dessas senóides é uma fração da frequência do som original. Assim, cada senóide corresponde a uma nota, e analisando as senóides podemos descobrir qual é a "mais forte" - que corresponde ao harmônico sendo tocado no som. A partir disso também podemos descobrir qual é a frequência fundamental, e outras propriedades do som.

2 Transcrição Musical

A DFT retorna um vetor de números complexos $a + bi$. Cada posição desse vetor (e o complexo armazenado nela) corresponde a uma senóide que compõe o som original. A posição no vetor indica a frequência da senóide (uma fração do *framerate* do som original), e o módulo do número complexo ($\sqrt{a^2 + b^2}$) em tal posição indica a amplitude da senóide.

O EP, ao fazer a transcrição musical de um arquivo .wav, separa o conjunto de dados em intervalos (equivalentes a 0.1 segundos), e procura pelo maior pico (a maior amplitude) nesse intervalo, guardando essa amplitude (do pico) e a frequência equivalente como sendo a "nota" do intervalo. Ao ir fazendo isso para cada intervalo do conjunto de dados, ele vai juntando notas de frequência iguais (aumentando a duração da mesma, e salvando a

variação da amplitude dela), ou criando notas novas para frequências diferentes.

Após a análise, ao salvar as notas em um arquivo MIDI, o EP usa a frequência para determinar a nota MIDI mais próxima, e usa os valores de amplitude que foram salvos para determinar a variação de volume na nota. Isso e o instrumento que escolhi para salvar os MIDIs (Ocarina) produzem um arquivo MIDI que é praticamente igual ao .wav original (nos vários testes que fiz com os arquivos de exemplos disponibilizados pela monitora no PACA).

3 Comparação dos Métodos de DFT

O EP tem implementado 3 métodos de DFT, e pode executar qualquer combinação deles puramente para comparar a eficiência entre eles. Quais métodos serão usados é definido por um argumento passado na linha de comando (vide o README).

Os métodos são:

- **FFT**: método de *fast-fourier transform* do NumPy.
- **Matrix**: método básico de multiplicação pela matriz de DFT. As colunas da matriz são guardadas como vetores do NumPy, e durante a aplicação da DFT (a multiplicação com o vetor de dados) uma função de multiplicação de vetores do NumPy é usada para multiplicar o vetor de dados pelas colunas da matriz, construindo o vetor de retorno da DFT.
- **Matrix (NumPy)**: método um pouco mais inteligente de multiplicação pela matriz de DFT. A matriz DFT inteira é guardada como uma matriz do NumPy, e a aplicação da DFT consiste numa única operação de multiplicação matriz-vetor do NumPy.

Em todos meus testes, a *FFT* foi sempre a mais rápida (quase instantânea), como esperado. No meu computador (i7) com uma versão recente do NumPy, o método *Matrix(NumPy)* é bem eficiente, se aproximando do tempo do FFT, enquanto o método *Matrix* normal demora consideravelmente mais que os outros 2 métodos.

No entanto em outra máquina que testei (i5) com uma versão mais antiga do NumPy, o método *Matrix* foi mais rápido que o método *Matrix(NumPy)*, e nenhum dos dois chegou perto do *FFT*.