基于智能分类算法的音乐和弦识别与分析

——基于PCP模型的实现与改进

摘 要： 和弦的辨别能力是在音乐学习当中必不可少的技能。但是，为进行过专业训练的人很难快速准确的对一首音乐中的和弦进行快速准确的听辨。随着计算机技术的发展，利用快速傅里叶变换与各种智能算法，本文拟使用上述算法，对音乐的和弦进行识别与分析。拟采用由Takuya FUJISHIMA于1999年提出的PCP（Pitch Class Profile）算法为基础，结合DNN等智能算法加以改进，增加音乐的节拍跟踪，实现对于音乐中和弦的智能识别系统。

关键词：和弦识别，智能算法，PCP，节拍跟踪

Music chord recognition and analysis based on intelligent classification algorithm

—— Implementation and improvement based on PCP model

**Abstract：** The ability to distinguish chords is an essential skill in music learning. However, it is difficult for a trained person to listen quickly and accurately to the chords in a piece of music. With the development of computer technology, we use Fast Fourier transform and various intelligent algorithms to identify and analyze the chords of music. Based on the algorithm of Pitch Class Profile (PCP) proposed by Takuya Fujishima in 1999, and combined with intelligent algorithms such as DNN, this paper proposes to improve it by adding the beat tracking of music and realizing the intelligent recognition system of chords in music.

**Key words：Chord recognition, intelligent algorithms, PCP, beat tracking**

1 音频预处理：音频采样、分帧、加窗与快速傅里叶变换

1.1音频采样

声音由物体的振动产生，在显示世界中，物体振动的函数是连续的。然而，要在计算机之中储存音频，需要对音频进行采样。采样的精度通常使用两个参数进行描述：采样率（Sample rate）与采样深度（Bit Depth）

采样率是指对音频信号进行采样的频率，单位为赫兹（Hz）。根据奈奎斯特采样定理，当采样频率fs.max大于信号中最高频率fmax的2倍时(fs.max>2fmax)，采样之后的数字信号完整地保留了原始信号中的信息。故在此次实验之中，为了平衡数据量与精确性，拟使用采样率为44100Hz。

1.2 音频分帧、加窗与傅里叶变换

1.2.1傅里叶变换：

由于要对音频中的和弦进行分析，故需要将信号的时域信号转换为频域信号。使用傅里叶变换来实现这一目的。而对于一段音乐信号来说，将一整段时长超过三分钟的信号进行直接进行傅里叶变换显然无法得到准确的频域特征。故需要对信号进行分帧，将每一帧的信号进行独立的快速傅里叶（FFT）变换，来获得相对可用的频域信息。

1.2.2 音频分帧

快速傅里叶变换之前需要对音频进行分帧处理，每一帧的长度称之为FFT大小（NFFT）。在本次实验中，取NFFT为1024.

为了保证每一帧之间信号的连续性，音频帧之间存在交叉，即前一帧的结尾部分与后一帧的开头存在交叉。在本次实验中，交叉区域的大小为256.

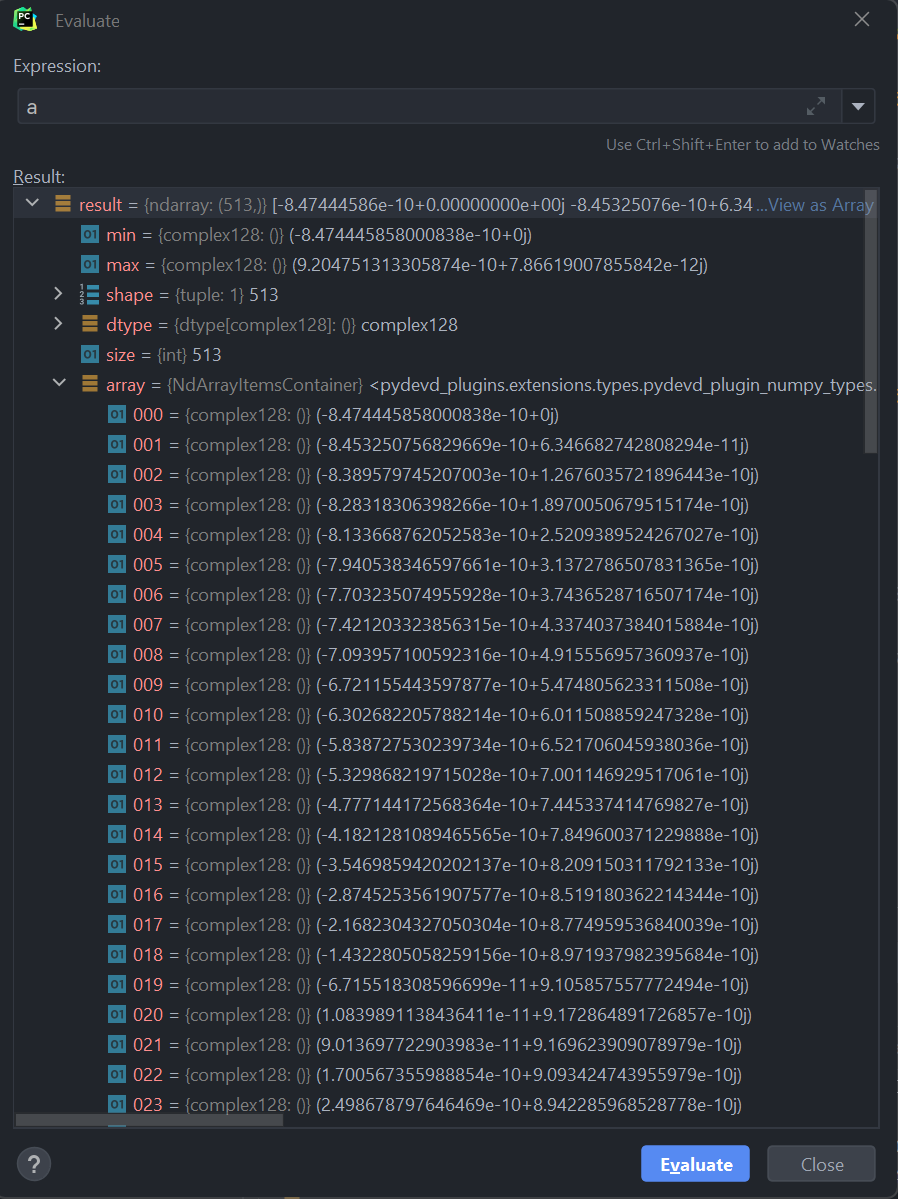
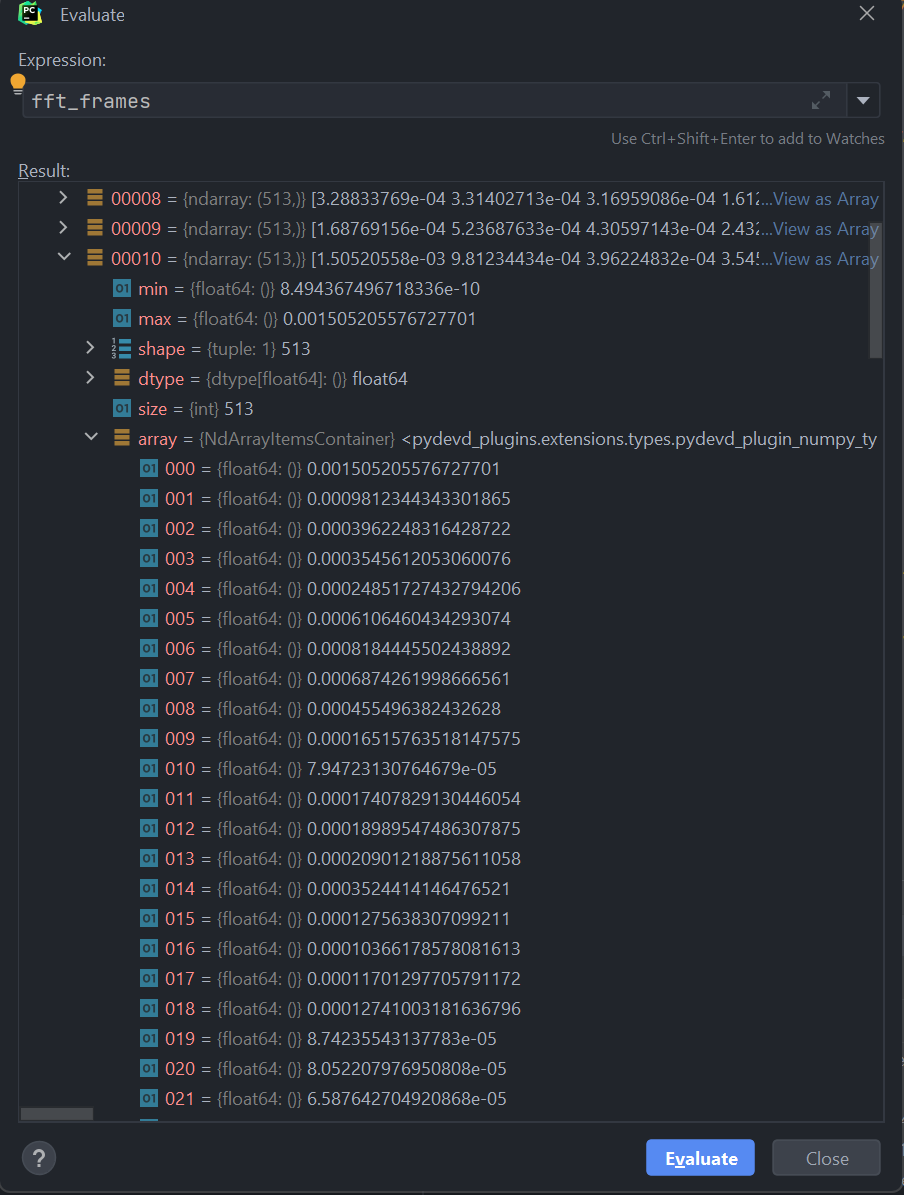
将信号截断之后，频谱会发生畸变，这种现象称之为频谱能量泄露。为了防止频谱能量泄露对后续的FFT产生影响，故需要对信号进行加窗处理。本次实验中，使用汉宁（Hanning）窗（升余弦窗）对信号进行加窗处理。汉宁（Hanning）窗的数学定义如下所示：

将每一帧的音频信号与窗函数相乘即可得到加窗信号：

其中，为加窗信号，为原始信号，为窗函数。

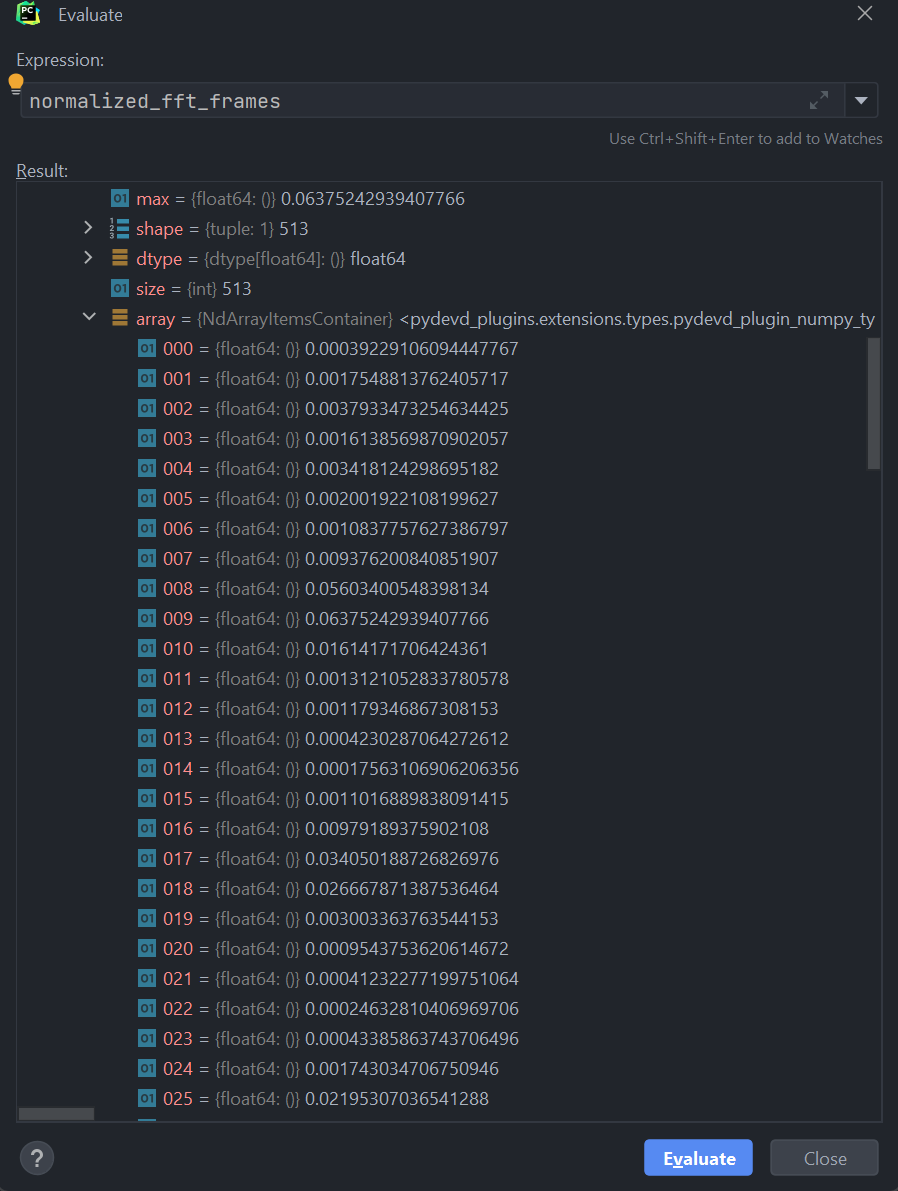
对加窗信号进行快速傅里叶变换：

在本次实验中，使用numpy.fft.rfft()函数对信号进行处理。处理的结果为一个长度为NFFT/2+1的一维array。numpy.fft.rfft()相比于numpy.fft.fft()抛弃了一对共轭的复数结果，故输出的长度为NFFT/2+1。由于仅对频率bin的强度进行考察，故此处将复数结果取模。numpy.fft. rfftfreq()函数用于生成与频率bin对应的频率点位。

图1 音频的一帧进行快速傅里叶变换的结果（左）与取模之后的结果（右）

将每一帧的FFT结果合并成一个矩阵，再与numpy.fft. rfftfreq()的结果进行对应，能得到音频在任何时刻的频率特点（频谱）。再对FFT矩阵进行规范化，为后续的数据处理做好准备。

A screenshot of a computer

Description automatically generated图2 规范化之后的FFT矩阵（左）与对应的频率点位（右）

对信号进行傅里叶变换是必须的，因为要对音频的和弦进行分析，需要得知音频某个时刻的频率特征。这是后续所有操作的基础。

2 节拍跟踪

**2.1 节拍的概念**

2.1.1 拍（Beats）

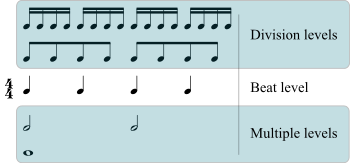
拍是音乐中组织音符的基本时间单位，所有的音符都基于这一个具有周期性的时间概念来组织。这个概念可以类比计算机科学中“时钟”的概念，它为计算机中所有的操作提供同步的标准。

图3 Beat Level表示一拍在小节中的组织方式，Division levels表示节拍的均分

2.1.2 小节（Measure）

拍按照一定的数量与形式组织成小节，组织方式通常用拍号（Signature）表示。以现代歌曲中常见的4/4拍为例子，它表示以四分音符为一拍，每小节由4拍构成。小节是乐句的基本组成。每小节的第一拍被称为下拍（Downbeat）。在本次实验中，下拍时间戳是判断和弦变换点位的主要参考标准。

2.1.3曲速（Tempo）

曲速是指每分钟的拍数（Beats per Minute，BPM），这是衡量音乐快慢的量化数据。在本次实验中，将提出一种检测歌曲曲速的算法。

2.2节拍跟踪算法

2.2.1音头

音头，是指一个音符开始发出声音的瞬间。在实际演奏中，表现为按下琴键（钢琴）的瞬间，或者鼓棒接触到鼓面的瞬间（打击乐器），也就是在谱面上显示出的音符的位置。上文提到，音符一般按照节拍为时间基础进行组织，也就是说，音头一般出现在拍上或者分拍层（Division Levels）上。

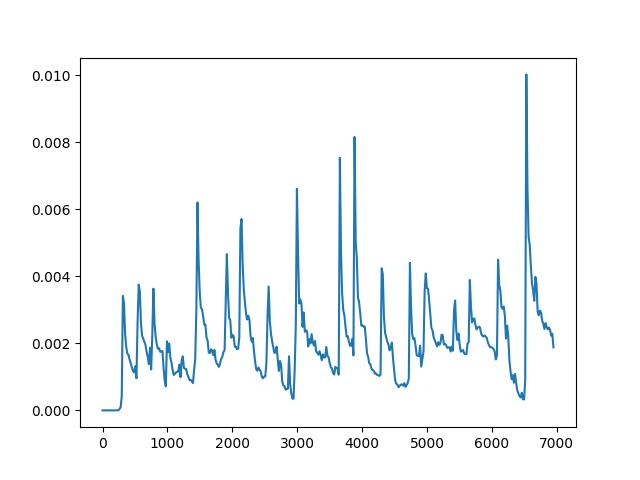
由于音头是一个音符开始发出声音的瞬间，而一个音符的延音，响度一般会缓慢衰减，即声音的能量会逐渐衰减。故我们可以对频谱的能量在时间维度上的变化来观察音头的大致位置，即能量的极大值点可以认为是一个可能的音头。

图4 一个音乐片段随时间的响度变化

图4显示了一个音乐片段的响度变化。可以看出，音频的响度符合上述音头的响度衰减特征。在本次实验中，拟提出一个算法，通过音频的周期性的响度，检测音频的BPM。

2.2.2周期乘数法

本节拟提出一个可能的BPM检测算法：周期乘数法。

构建一个周期乘数函数：

*(1)*