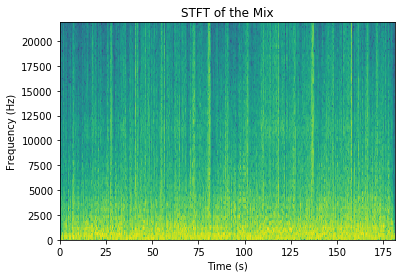
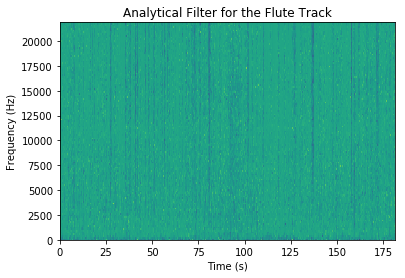
# Introduction

A fully convolutional autoencoder can be used to perform audio source separation. In this report such as network is used to extract a single instrument from a multi-instrument musical recording.

The network is a fully convolutional auto encoder. A fully convolutional autoencoder does not use any fully connected layers. The task that is performed by the network is the generation of a spectral filter that can be used to extract the instrument from the input mix. This process is conceptually similar to generating a mask in the domain of image segmentation, except that the filter is real-valued and is used to generate a new mixture rather than indicate a relevant portion of an image. Originally the network was meant to work as a filter—like a de-noising autoencoder—but it was found that the approach mentioned performs better.

**Figure 1** The STFT of the input multi-track mix shows how the frequency-domain content of the signal varies over time.

The data that is used for training and validation is a subset of the *MedleyDB* project [1]. This database consists of groups of “wav” files. Each group contains a multi-instrument file (the mix) and a collection of files that isolate each instrument (the tracks). Following processing that will be discussed later, the mix is the input to the network and one of the tracks—depending on which instrument is to be isolated—is the output of the network.

**Figure 2** The analytical filter shows the best-case performance and is used as the target in the training process.

Before training, the data is converted by a Short Time Fourier Transform (STFT) into a sequence of overlapping spectra, i.e. a spectrogram like **Figure 1**. The STFT was configured so that each column in the spectrogram represents ~6 ms of the mix. The track containing the flute, which is the instrument that will be isolated in this report, was further processed into a filter (**Figure 2**). The first ~165 seconds of the spectrograms shown in **Figure 1** and **Figure 2** show the training input (X) and output (y) datasets respectively. The last ten seconds were used for verification.

The fully convolutional autoencoder produced a validation accuracy of 82%.

# Feature engineering

* Frequency characteristics (fundamental, harmonics)
* Attack/Decay

# network architecture

* Fully convolutional autoencoder

# Results

* 82%

# moving forward

* Incorporate dense layers to better capture broadband characteristics
* Process multiple samples at a time to capture attack and decay characteristics with the convolutional layers

# references