```
pip install pysptk pyworld librosa tqdm
```

```
Collecting pysptk
  Downloading pysptk-1.0.1.tar.gz (461 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 461.9/461.9 kB 7.2 MB/s eta 0:00:00
  Installing build dependencies ... done
  Getting requirements to build wheel ... done
  Preparing metadata (pyproject.toml) ... done
Collecting pyworld
  Downloading pyworld-0.3.5.tar.gz (261 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 261.0/261.0 kB 17.7 MB/s eta 0:00:00
  Installing build dependencies ... done
  Getting requirements to build wheel ... done
  Preparing metadata (pyproject.toml) ... done
Requirement already satisfied: librosa in /usr/local/lib/python3.11/dist-packages (0.10.2.post1)
Requirement already satisfied: tqdm in /usr/local/lib/python3.11/dist-packages (4.67.1)
Requirement already satisfied: scipy in /usr/local/lib/python3.11/dist-packages (from pysptk) (1.13.1)
Requirement already satisfied: decorator in /usr/local/lib/python3.11/dist-packages (from pysptk) (4.4.2)
Requirement already satisfied: cython>=0.28.0 in /usr/local/lib/python3.11/dist-packages (from pysptk) (3.0.11)
Requirement already satisfied: numpy in /usr/local/lib/python3.11/dist-packages (from pyworld) (1.26.4)
Requirement already satisfied: audioread>=2.1.9 in /usr/local/lib/python3.11/dist-packages (from librosa) (3.0.1)
Requirement already satisfied: scikit-learn>=0.20.0 in /usr/local/lib/python3.11/dist-packages (from librosa) (1.6.0)
Requirement already satisfied: joblib>=0.14 in /usr/local/lib/python3.11/dist-packages (from librosa) (1.4.2)
Requirement already satisfied: numba>=0.51.0 in /usr/local/lib/python3.11/dist-packages (from librosa) (0.60.0)
Requirement already satisfied: soundfile>=0.12.1 in /usr/local/lib/python3.11/dist-packages (from librosa) (0.13.0)
Requirement already satisfied: pooch>=1.1 in /usr/local/lib/python3.11/dist-packages (from librosa) (1.8.2)
Requirement already satisfied: soxr>=0.3.2 in /usr/local/lib/python3.11/dist-packages (from librosa) (0.5.0.post1)
Requirement already satisfied: typing-extensions>=4.1.1 in /usr/local/lib/python3.11/dist-packages (from librosa) (4.12.2)
Requirement already satisfied: lazy-loader>=0.1 in /usr/local/lib/python3.11/dist-packages (from librosa) (0.4)
Requirement already satisfied: msgpack>=1.0 in /usr/local/lib/python3.11/dist-packages (from librosa) (1.1.0)
Requirement already satisfied: packaging in /usr/local/lib/python3.11/dist-packages (from lazy-loader>=0.1->librosa) (24.2)
Requirement already satisfied: llvmlite<0.44,>=0.43.0dev0 in /usr/local/lib/python3.11/dist-packages (from numba>=0.51.0->li
Requirement already satisfied: platformdirs>=2.5.0 in /usr/local/lib/python3.11/dist-packages (from pooch>=1.1->librosa) (4.
Requirement already satisfied: requests>=2.19.0 in /usr/local/lib/python3.11/dist-packages (from pooch>=1.1->librosa) (2.32.
Requirement already satisfied: threadpoolctl>=3.1.0 in /usr/local/lib/python3.11/dist-packages (from scikit-learn>=0.20.0->l
Requirement already satisfied: cffi>=1.0 in /usr/local/lib/python3.11/dist-packages (from soundfile>=0.12.1->librosa) (1.17.
Requirement already satisfied: pycparser in /usr/local/lib/python3.11/dist-packages (from cffi>=1.0->soundfile>=0.12.1->libr
Requirement already satisfied: charset-normalizer<4,>=2 in /usr/local/lib/python3.11/dist-packages (from requests>=2.19.0->p
Requirement already satisfied: idna<4,>=2.5 in /usr/local/lib/python3.11/dist-packages (from requests>=2.19.0->pooch>=1.1->l
Requirement already satisfied: urllib3<3,>=1.21.1 in /usr/local/lib/python3.11/dist-packages (from requests>=2.19.0->pooch>=
Requirement already satisfied: certifi>=2017.4.17 in /usr/local/lib/python3.11/dist-packages (from requests>=2.19.0->pooch>=
Building wheels for collected packages: pysptk, pyworld
  Building wheel for pysptk (pyproject.toml) ... done
  Created wheel for pysptk: filename=pysptk-1.0.1-cp311-cp311-linux_x86_64.whl size=1293723 sha256=59d47cb2a538577b60f047971
  Stored in directory: /root/.cache/pip/wheels/f7/8e/e6/be2295ab3cba2d52e826922cd3ea8bda4a18210c03584dc759
  Building wheel for pyworld (pyproject.toml) ... done
  Created wheel for pyworld: filename=pyworld-0.3.5-cp311-cp311-linux_x86_64.whl size=899906 sha256=524f7619dcada425bdb6c20d
  Stored in directory: /root/.cache/pip/wheels/26/f0/db/ebcd5cdfe5ad7d229917d3a8db6f18f0cf40f099bf878e294d
Successfully built pysptk pyworld
Installing collected packages: pyworld, pysptk
Successfully installed pysptk-1.0.1 pyworld-0.3.5
```

```
from google.colab import drive
drive.mount('/content/drive')
```

```
Mounted at /content/drive
```

```
import os

directory_path = "/content/drive/MyDrive/NewData"

if os.path.exists(directory_path):
    print(f"The directory '{directory_path}' exists.")
else:
    print(f"The directory '{directory_path}' does not exist.")
```

```
The directory '/content/drive/MyDrive/NewData' exists.
```

```
# !wget -P /content/drive/MyDrive/NewData "http://festvox.org/cmu_arctic/cmu_arctic/packed/cmu_us_slt_arctic-0.95-release.tar.bz
```

```
# !tar -xvjf /content/drive/MyDrive/NewData/cmu_us_bdl_arctic-0.95-release.tar.bz2 -C /content/drive/MyDrive/NewData/
# !tar -xvjf /content/drive/MyDrive/NewData/cmu_us_clb_arctic-0.95-release.tar.bz2 -C /content/drive/MyDrive/NewData/
# !tar -xvjf /content/drive/MyDrive/NewData/cmu_us_jmk_arctic-0.95-release.tar.bz2 -C /content/drive/MyDrive/NewData/
# !tar -xvjf /content/drive/MyDrive/NewData/cmu_us_ksp_arctic-0.95-release.tar.bz2 -C /content/drive/MyDrive/NewData/
# !tar -xvjf /content/drive/MyDrive/NewData/cmu_us_rms_arctic-0.95-release.tar.bz2 -C /content/drive/MyDrive/NewData/
# !tar -xvjf /content/drive/MyDrive/NewData/cmu_us_slt_arctic-0.95-release.tar.bz2 -C /content/drive/MyDrive/NewData/
```

```
from os.path import join, expanduser
DATA_ROOT = join("/content", "drive", "MyDrive", "NewData")
print(DATA_ROOT)
!ls $DATA_ROOT
```

```
/content/drive/MyDrive/NewData
cmu_us_awb_arctic  cmu_us_clb_arctic  cmu_us_ksp_arctic  cmu_us_slt_arctic
cmu_us_bdl_arctic  cmu_us_jmk_arctic  cmu_us_rms_arctic
```

```
%pylab inline
rcParams["figure.figsize"] = (16,5)

!pip install nnmnkwii
```

```
Populating the interactive namespace from numpy and matplotlib
Collecting nnmnkwii
  Downloading nnmnkwii-0.1.3.tar.gz (1.5 MB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 1.5/1.5 MB 16.1 MB/s eta 0:00:00
  Installing build dependencies ... done
  Getting requirements to build wheel ... done
  Preparing metadata (pyproject.toml) ... done
Requirement already satisfied: scipy in /usr/local/lib/python3.11/dist-packages (from nnmnkwii) (1.13.1)
Requirement already satisfied: cython>=0.28.0 in /usr/local/lib/python3.11/dist-packages (from nnmnkwii) (3.0.11)
Collecting fastdtw (from nnmnkwii)
  Downloading fastdtw-0.3.4.tar.gz (133 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 133.4/133.4 kB 8.2 MB/s eta 0:00:00
  Preparing metadata (setup.py) ... done
Requirement already satisfied: scikit-learn in /usr/local/lib/python3.11/dist-packages (from nnmnkwii) (1.6.0)
Requirement already satisfied: pysptk>=0.1.17 in /usr/local/lib/python3.11/dist-packages (from nnmnkwii) (1.0.1)
Requirement already satisfied: tqdm in /usr/local/lib/python3.11/dist-packages (from nnmnkwii) (4.67.1)
Requirement already satisfied: numpy>=1.20.0 in /usr/local/lib/python3.11/dist-packages (from nnmnkwii) (1.26.4)
Requirement already satisfied: decorator in /usr/local/lib/python3.11/dist-packages (from pysptk>=0.1.17->nnmnkwii) (4.4.2)
Requirement already satisfied: joblib>=1.2.0 in /usr/local/lib/python3.11/dist-packages (from scikit-learn->nnmnkwii) (1.4.2
Requirement already satisfied: threadpoolctl>=3.1.0 in /usr/local/lib/python3.11/dist-packages (from scikit-learn->nnmnkwii)
Building wheels for collected packages: nnmnkwii, fastdtw
  Building wheel for nnmnkwii (pyproject.toml) ... done
  Created wheel for nnmnkwii: filename=nnmnkwii-0.1.3-cp311-cp311-linux_x86_64.whl size=3768491 sha256=ecbc48f13a8b23debd593
  Stored in directory: /root/.cache/pip/wheels/18/98/9e/f02110e10183689a04fdffa478e825392a6b9fdabee4ed968f
  Building wheel for fastdtw (setup.py) ... done
  Created wheel for fastdtw: filename=fastdtw-0.3.4-cp311-cp311-linux_x86_64.whl size=542099 sha256=59d9d66dad24081fe39468c4
  Stored in directory: /root/.cache/pip/wheels/5c/8a/f6/fd3df9a9714677410a5ccbf3ca519e66db4a54a1c46ea95332
Successfully built nnmnkwii fastdtw
Installing collected packages: fastdtw, nnmnkwii
Successfully installed fastdtw-0.3.4 nnmnkwii-0.1.3
```

Generate | a slider using jupyter widgets | Close

```
import nnmnkwii

from nnmnkwii.datasets import PaddedFileSourceDataset
from nnmnkwii.datasets.cmu_arctic import CMUArcticWavFileDataSource
from nnmnkwii.preprocessing.alignment import DTWAligner
from nnmnkwii.preprocessing import trim_zeros_frames, remove_zeros_frames, delta_features
from nnmnkwii.util import apply_each2d_trim
from nnmnkwii.metrics import melcd
from nnmnkwii.baseline.gmm import MLPG

from os.path import basename, splitext
import sys
import time

import numpy as np
from scipy.io import wavfile
from sklearn.mixture import GaussianMixture
from sklearn.model_selection import train_test_split
import pyworld
import pysptk
from pysptk.synthesis import MLSADF, Synthesizer
import librosa
import librosa.display
import IPython
from IPython.display import Audio
```

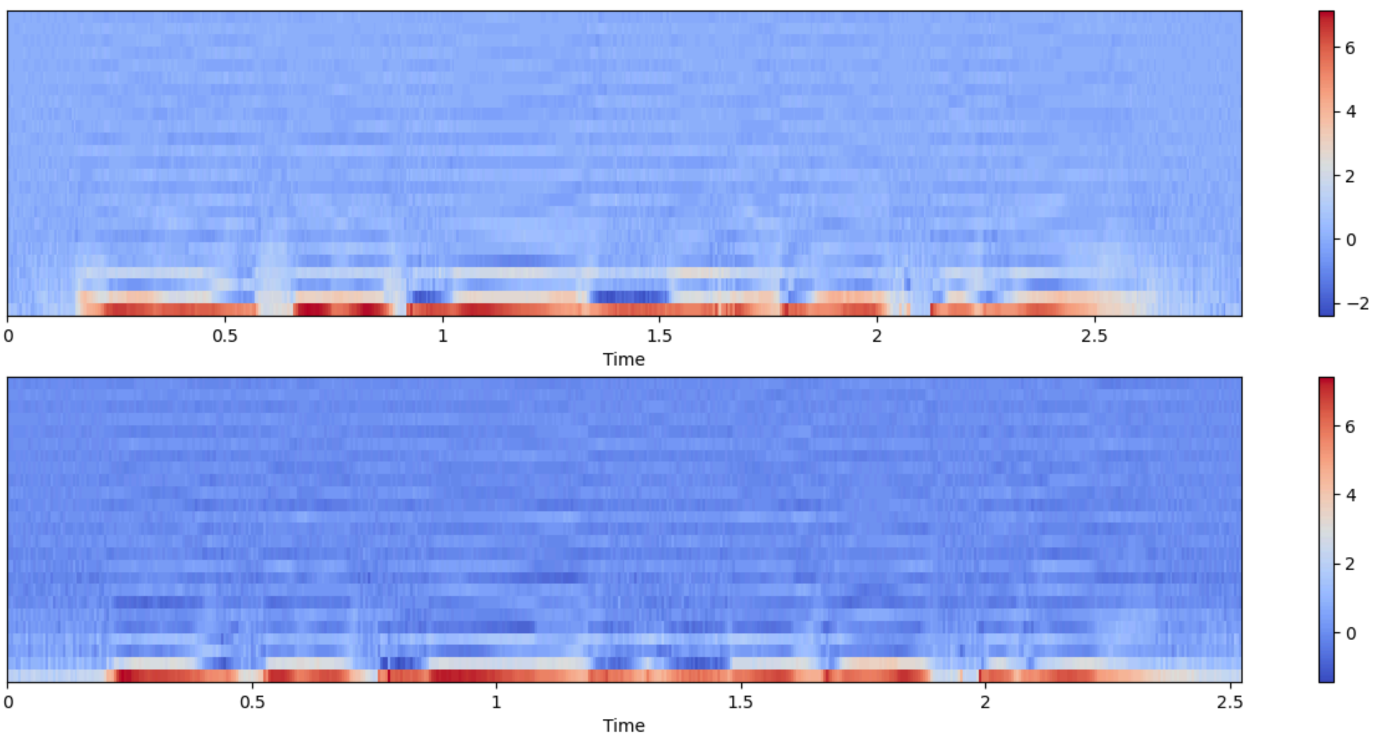
```
fs = 16000
fftlen = pyworld.get_cheaptrick_fft_size(fs)
alpha = pysptk.util.mcepalpha(fs)
order = 24
frame_period = 5
hop_length = int(fs * (frame_period * 0.001))
max_files = 100 # number of utterances to be used.
test_size = 0.03
use_delta = True

if use_delta:
    windows = [
        (0, 0, np.array([1.0])),
        (1, 1, np.array([-0.5, 0.0, 0.5])),
        (1, 1, np.array([1.0, -2.0, 1.0])),
    ]
else:
    windows = [
        (0, 0, np.array([1.0])),
    ]


class MyFileDataSource(CMUArcticWavFileDataSource):
    def __init__(self, *args, **kwargs):
        super(MyFileDataSource, self).__init__(*args, **kwargs)
        self.test_paths = None

    def collect_files(self):
        paths = super(
            MyFileDataSource, self).collect_files()
        paths_train, paths_test = train_test_split(
            paths, test_size=test_size, random_state=1234)

        # keep paths for later testing
        self.test_paths = paths_test

        return paths_train

    def collect_features(self, path):
        fs, x = wavfile.read(path)
        x = x.astype(np.float64)
        f0, timeaxis = pyworld.dio(x, fs, frame_period=frame_period)
        f0 = pyworld.stonemask(x, f0, timeaxis, fs)
        spectrogram = pyworld.cheaptrick(x, f0, timeaxis, fs)
        spectrogram = trim_zeros_frames(spectrogram)
        mc = pysptk.sp2mc(spectrogram, order=order, alpha=alpha)
        return mc


clb_source = MyFileDataSource(data_root=DATA_ROOT,
                                         speakers=["clb"], max_files=max_files)
slt_source = MyFileDataSource(data_root=DATA_ROOT,
                                         speakers=["slt"], max_files=max_files)


X = PaddedFileSourceDataset(clb_source, 1200).asarray()
Y = PaddedFileSourceDataset(slt_source, 1200).asarray()
print(X.shape)
print(Y.shape)
```

```
(97, 1200, 25)
(97, 1200, 25)
```

One of the pairs of our dataset visualized

```
# Plotting util
def plot_parallel(x,y):
    figure(figsize=(16,7))
    subplot(2,1,1)
    librosa.display.specshow(trim_zeros_frames(x).T, sr=fs, hop_length=hop_length, x_axis="time")
    colorbar()
    subplot(2,1,2)
    librosa.display.specshow(trim_zeros_frames(y).T, sr=fs, hop_length=hop_length, x_axis="time")
    colorbar()
```

```
idx = 22 # any
plot_parallel(X[idx],Y[idx])
```

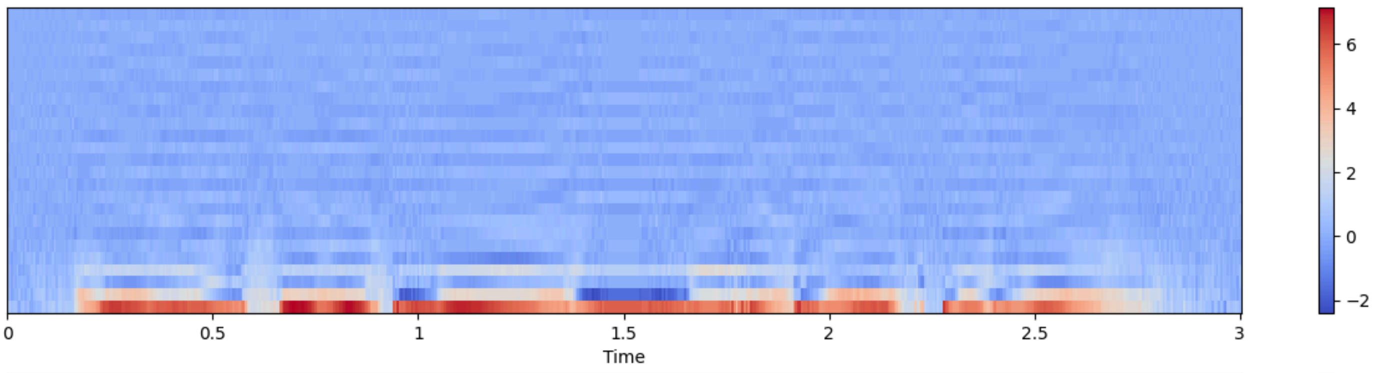


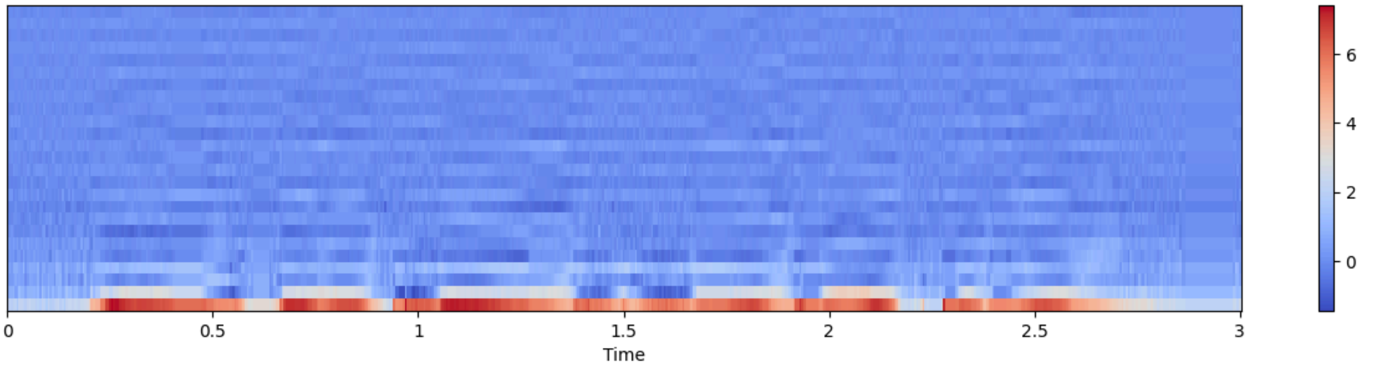
Now Aligning Source and Target Features

```
X_aligned, Y_aligned = DTWAligner(verbose=0, dist=melcd).transform((X, Y))
```

Let's see how the data looks like now

```
plot_parallel(X_aligned[idx],Y_aligned[idx])
```





Dropping the 1st dimension or [0] of the mel-cepstrum, coz it defines the energy or loudness of voice and we dont consider that as a spectral property to be retained while voice conversion

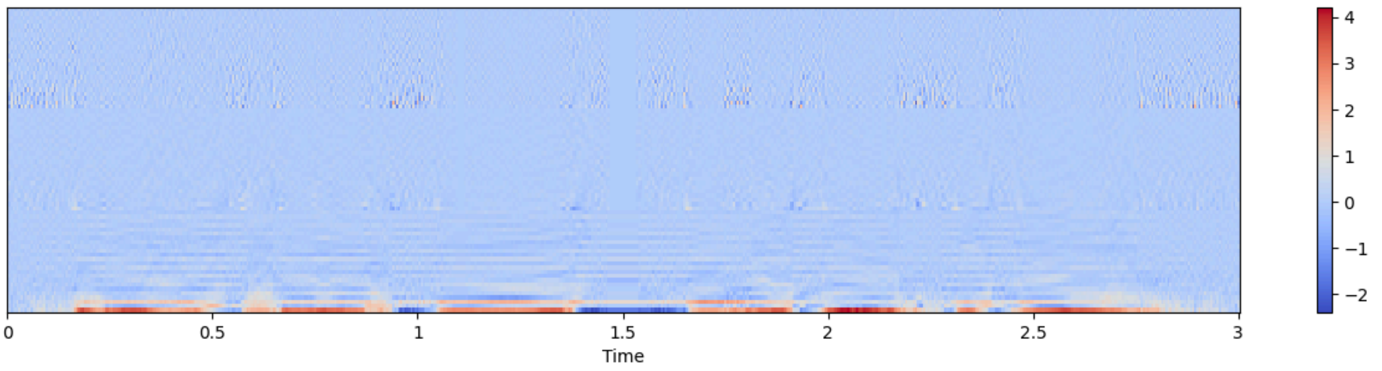
```
# Drop 1st (power) dimension
X_aligned, Y_aligned = X_aligned[:, :, 1:], Y_aligned[:, :, 1:]
```

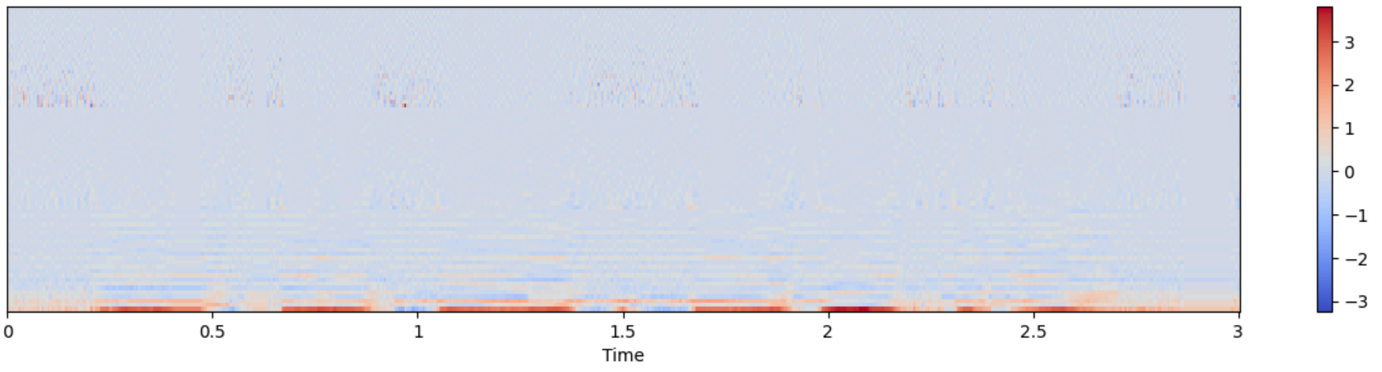
Double-click (or enter) to edit

Append delta features

```
static_dim = X_aligned.shape[-1]
if use_delta:
    X_aligned = apply_each2d_trim(delta_features, X_aligned, windows)
    Y_aligned = apply_each2d_trim(delta_features, Y_aligned, windows)
```

```
plot_parallel(X_aligned[idx],Y_aligned[idx])
```





```
XY = np.concatenate((X_aligned, Y_aligned), axis=-1).reshape(-1, X_aligned.shape[-1]*2)
print(XY.shape)
```

```
(116400, 144)
```

```
XY = remove_zeros_frames(XY)
print(XY.shape)
```

```
(68186, 144)
```

```
gmm = GaussianMixture(
    n_components=64, covariance_type="full", max_iter=100, verbose=1)

%time gmm.fit(XY)
```

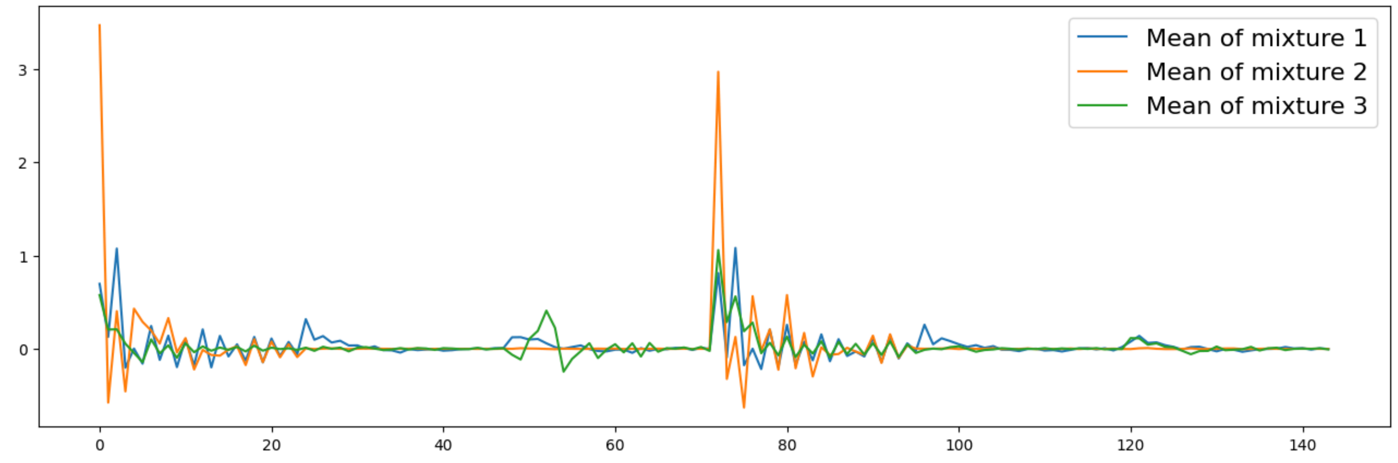
```
Initialization 0
  Iteration 10
  Iteration 20
  Iteration 30
  Iteration 40
Initialization converged.
CPU times: user 37min 16s, sys: 13min 45s, total: 51min 2s
Wall time: 31min 57s
▾         GaussianMixture
GaussianMixture(n_components=64, verbose=1)
```

Visualize model: Viewing the parameters of GMM

Means

```
for k in range(3):
    plot(gmm.means_[k], linewidth=1.5, label="Mean of mixture {}".format(k+1))
legend(prop={"size": 16})
```

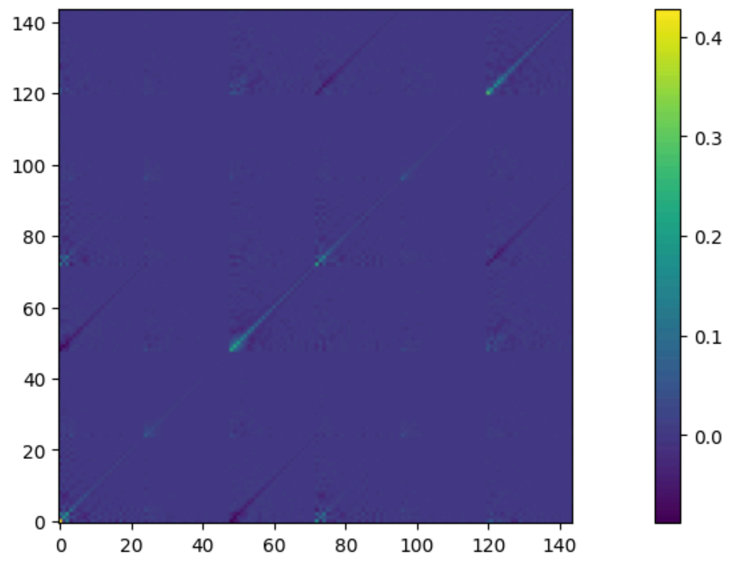
```
<matplotlib.legend.Legend at 0x7e3fd3f47650>
```



Covariances

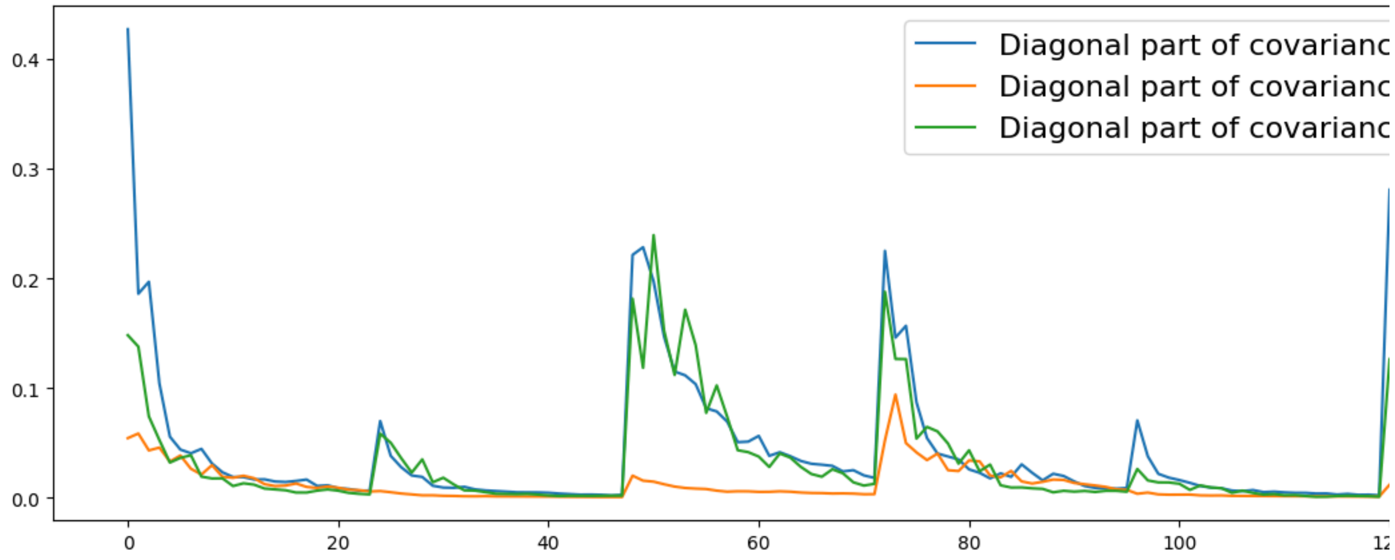
```
imshow(gmm.covariances_[0], origin="lower")
colorbar()
```

```
<matplotlib.colorbar.Colorbar at 0x7e3fd3ca39d0>
```



```
for k in range(3):
    plot(np.diag(gmm.covariances_[k]), linewidth=1.5,
         label="Diagonal part of covariance matrix, mixture {}".format(k))
legend(prop={"size": 16})
```

```
<matplotlib.legend.Legend at 0x7e3fd3e71fd0>
```



Testing: Parameter Generation Using MLPG

```
def test_one_utt(src_path, tgt_path, disable_mlpg=False, diffvc=True):
    # GMM-based parameter generation is provided by the library in `baseline` module
    if disable_mlpg:
        # Force disable MLPG
        paramgen = MLPG(gmm, windows=[(0,0, np.array([1.0]))], diff=diffvc)
    else:
        paramgen = MLPG(gmm, windows=windows, diff=diffvc)

    fs, x = wavfile.read(src_path)
    x = x.astype(np.float64)
    f0, timeaxis = pyworld.dio(x, fs, frame_period=frame_period)
    f0 = pyworld.stonemask(x, f0, timeaxis, fs)
    spectrogram = pyworld.cheaptrick(x, f0, timeaxis, fs)
    aperiodicity = pyworld.d4c(x, f0, timeaxis, fs)

    mc = pysptk.sp2mc(spectrogram, order=order, alpha=alpha)
    c0, mc = mc[:, 0], mc[:, 1:]
    if use_delta:
        mc = delta_features(mc, windows)
    mc = paramgen.transform(mc)
    if disable_mlpg and mc.shape[-1] != static_dim:
        mc = mc[:,:static_dim]
    assert mc.shape[-1] == static_dim
    mc = np.hstack((c0[:, None], mc))
    if diffvc:
        mc[:, 0] = 0 # remove power coefficients
        engine = Synthesizer(MLSADF(order=order, alpha=alpha), hopsize=hop_length)
        b = pysptk.mc2b(mc.astype(np.float64), alpha=alpha)
        waveform = engine.synthesis(x, b)
    else:
        spectrogram = pysptk.mc2sp(
            mc.astype(np.float64), alpha=alpha, fftlen=fftlen)
        waveform = pyworld.synthesize(
            f0, spectrogram, aperiodicity, fs, frame_period)

    return waveform


for i, (src_path, tgt_path) in enumerate(zip(clb_source.test_paths, slt_source.test_paths)):
    print("{}-th sample".format(i+1))
    wo_MLPG = test_one_utt(src_path, tgt_path, disable_mlpg=True)
    w_MLPG = test_one_utt(src_path, tgt_path, disable_mlpg=False)
    _, src = wavfile.read(src_path)
    _, tgt = wavfile.read(tgt_path)

    print("Source:", basename(src_path))
    IPython.display.display(Audio(src, rate=fs))
    print("Target:", basename(tgt_path))
    IPython.display.display(Audio(tgt, rate=fs))
    print("w/o MLPG")
    IPython.display.display(Audio(wo_MLPG, rate=fs))
    print("w/ MLPG")
    IPython.display.display(Audio(w_MLPG, rate=fs))
```

```
1-th sample
Source: arctic_a0041.wav

    0:02 / 0:02

Target: arctic_a0041.wav

    0:02 / 0:02

w/o MLPG

    0:02 / 0:02

w/ MLPG

    0:02 / 0:02

2-th sample
Source: arctic_a0036.wav

    0:02 / 0:02

Target: arctic_a0036.wav

    0:01 / 0:01

w/o MLPG

    0:02 / 0:02

w/ MLPG

    0:02 / 0:02

3-th sample
Source: arctic_a0082.wav

    0:02 / 0:02

Target: arctic_a0082.wav

    0:02 / 0:02

w/o MLPG

    0:02 / 0:02

w/ MLPG

    0:02 / 0:02
```

How different are they?

```
def vis_difference(x, y, which_dims=[0,2,3,6,8], T_max=None):
    static_paramgen = MLPG(gmm, windows=[(0,0, np.array([1.0]))], diff=False)
    paramgen = MLPG(gmm, windows=windows, diff=False)

    x = trim_zeros_frames(x)
    y = trim_zeros_frames(y)[:,:static_dim]
    y_hat1 = static_paramgen.transform(x)[:,:static_dim]
    y_hat2 = paramgen.transform(x)

    if T_max is not None and len(y) > T_max:
        y,y_hat1,y_hat2 = y[:T_max],y_hat1[:T_max],y_hat2[:T_max]

    figure(figsize=(16,4*len(which_dims)))
    for idx, which_dim in enumerate(which_dims):
        subplot(len(which_dims), 1, idx+1)
        plot(y[:,which_dim], "--", linewidth=1, label="Target")
```

```
            plot(y_hat1[:,which_dim], "-", linewidth=2, label="w/o MLPG")
            plot(y_hat2[:,which_dim], "-", linewidth=3, label="w/ MLPG")
            title("{}-th coef".format(which_dim+1), fontsize=16)
            legend(prop={"size": 16}, loc="upper right")


idx = 0
which_dims = np.arange(0, static_dim, step=2)
vis_difference(X_aligned[idx], Y_aligned[idx], T_max=300, which_dims=which_dims)
```