miditime-Texas

June 7, 2020

PART 1: MAKING MIDI for different Cities

[1]:
```
import pandas as pd
```

[2]:
```
time=pd.read_csv('data/time_series_data.csv')
others=pd.read_csv('data/other_information.csv')
location=pd.read_csv('data/location.csv')
```

[7]:
```
# find texas cities (4):
AU=time[time['Combined_Key'].str.match('Austin')].iloc[0,:]
DA=time[time['Combined_Key'].str.match('Dallas')].iloc[0,:]
EI=time[time['Combined_Key'].str.match('El Paso')].iloc[0,:]
HS=time[time['Combined_Key'].str.match('Houston')].iloc[0,:]
```

[9]:
```
#Using daily increase as New York example point out that daily numbers are␣
 ↪better in sound
AU_day=AU[1:].diff().fillna(0)
DA_day=DA[1:].diff().fillna(0)
EI_day=EI[1:].diff().fillna(0)
HS_day=HS[1:].diff().fillna(0)
```

[10]:
```
#basic range:
low_AU=min(AU_day)
high_AU=max(AU_day)
low_DA=min(DA_day)
high_DA=max(DA_day)
low_EI=min(EI_day)
high_EI=max(EI_day)
low_HS=min(HS_day)
high_HS=max(HS_day)
```

[11]:
```
from miditime.miditime import MIDITime
```

[12]:
```
#generate midi file with different base octave based on their geological␣
 ↪location, cities in a norther position have higher base octave
midi1 = MIDITime(120, 'results/Texas/AU.mid', 120, 5, 1)
midi2 = MIDITime(120, 'results/Texas/HS.mid', 120, 4, 1)
midi3 = MIDITime(120, 'results/Texas/EI.mid', 120, 6, 1)
midi4 = MIDITime(120, 'results/Texas/DA.mid', 120, 7, 1)
```

[13]:
```
#different mag_to_pitch function
def mag_to_pitch_tuned(magnitude, low, high):
    # Where does this data point sit in the domain of your data? (I.E. the min␣
 ↪magnitude is 3, the max in 5.6). In this case the optional 'True' means the␣
 ↪scale is reversed, so the highest value will return the lowest percentage.
    scale_pct = midi1.linear_scale_pct(low, high, magnitude)

    # Another option: Linear scale, reverse order
    # scale_pct = mymidi.linear_scale_pct(3, 5.7, magnitude, True)

    # Another option: Logarithmic scale, reverse order
    #scale_pct = mymidi.log_scale_pct(low, high, magnitude, True)

    # Pick a range of notes. This allows you to play in a key.
    c_major = ['C', 'D', 'E', 'F', 'G', 'A', 'B']
    #Find the note that matches your data point

    note = midi1.scale_to_note(scale_pct, c_major)

    #Translate that note to a MIDI pitch
    midi_pitch = midi1.note_to_midi_pitch(note)

    return midi_pitch
```

[14]:
```
def mag_to_pitch_tuned_Nmin(magnitude, low, high):
    # Where does this data point sit in the domain of your data? (I.E. the min␣
 ↪magnitude is 3, the max in 5.6). In this case the optional 'True' means the␣
 ↪scale is reversed, so the highest value will return the lowest percentage.
    scale_pct = midi2.linear_scale_pct(low, high, magnitude)

    # Another option: Linear scale, reverse order
    # scale_pct = mymidi.linear_scale_pct(3, 5.7, magnitude, True)

    # Another option: Logarithmic scale, reverse order
    #scale_pct = mymidi.log_scale_pct(low, high, magnitude, True)

    # Pick a range of notes. This allows you to play in a key.
    c_Nmin = ['C', 'D', 'Eb', 'F', 'G', 'Ab', 'Bb']
    #Find the note that matches your data point

    note = midi2.scale_to_note(scale_pct, c_Nmin)

    #Translate that note to a MIDI pitch
    midi_pitch = midi2.note_to_midi_pitch(note)

    return midi_pitch
```

[15]:
```
def mag_to_pitch_tuned_Hmin(magnitude, low, high):
    # Where does this data point sit in the domain of your data? (I.E. the min␣
 ↪magnitude is 3, the max in 5.6). In this case the optional 'True' means the␣
 ↪scale is reversed, so the highest value will return the lowest percentage.
    scale_pct = midi3.linear_scale_pct(low, high, magnitude)

    # Another option: Linear scale, reverse order
    # scale_pct = mymidi.linear_scale_pct(3, 5.7, magnitude, True)

    # Another option: Logarithmic scale, reverse order
    #scale_pct = mymidi.log_scale_pct(low, high, magnitude, True)

    # Pick a range of notes. This allows you to play in a key.
    c_Hmin = ['C', 'D', 'Eb', 'F', 'G', 'Ab', 'B']
    #Find the note that matches your data point

    note = midi3.scale_to_note(scale_pct, c_Hmin)

    #Translate that note to a MIDI pitch
    midi_pitch = midi3.note_to_midi_pitch(note)

    return midi_pitch
```

[16]:
```
def mag_to_pitch_tuned_Mmin(magnitude, low, high):
    # Where does this data point sit in the domain of your data? (I.E. the min␣
 ↪magnitude is 3, the max in 5.6). In this case the optional 'True' means the␣
 ↪scale is reversed, so the highest value will return the lowest percentage.
    scale_pct = midi4.linear_scale_pct(low, high, magnitude)

    # Another option: Linear scale, reverse order
    # scale_pct = mymidi.linear_scale_pct(3, 5.7, magnitude, True)

    # Another option: Logarithmic scale, reverse order
    #scale_pct = mymidi.log_scale_pct(low, high, magnitude, True)

    # Pick a range of notes. This allows you to play in a key.
    c_Mmin = ['C', 'D', 'Eb', 'F', 'G', 'A', 'B']
    #Find the note that matches your data point

    note = midi4.scale_to_note(scale_pct, c_Mmin)

    #Translate that note to a MIDI pitch
    midi_pitch = midi4.note_to_midi_pitch(note)

    return midi_pitch
```

[17]:
```
# AU, C major:
data1=AU_day.copy()
```

```
data1.index=list(range(len(data1)))
data1=data1.to_dict()
data1=[{'beat': d, "overall": data1[d]} for d in data1.keys()]
my_data_timed1 = [{'beat': midi1.beat(d['beat']), 'magnitude': d['overall']}␣
 ↪for d in data1]
note_list1 = []
counter=0
for d in my_data_timed1:
    try:
        note_list1.append([
            d['beat'],
            mag_to_pitch_tuned(d['magnitude'], low_AU, high_AU),
            100,  # velocity
            1  # duration, in beats
        ])
        counter +=1
    except Exception as e:
        print(d['beat'])
        print(counter)
        print(e)
        # some notes may do not match
midi1.add_track(note_list1)
midi1.save_midi()
```

```
60 0.0 1 100
60 0.66 1 100
60 1.31 1 100
60 1.97 1 100
60 2.63 1 100
60 3.29 1 100
60 3.94 1 100
60 4.6 1 100
60 5.26 1 100
60 5.91 1 100
60 6.57 1 100
60 7.23 1 100
60 7.89 1 100
60 8.54 1 100
60 9.2 1 100
60 9.86 1 100
60 10.51 1 100
60 11.17 1 100
60 11.83 1 100
60 12.48 1 100
60 13.14 1 100
60 13.8 1 100
60 14.46 1 100
```

```
60 15.11 1 100
60 15.77 1 100
60 16.43 1 100
60 17.08 1 100
60 17.74 1 100
60 18.4 1 100
60 19.06 1 100
60 19.71 1 100
60 20.37 1 100
60 21.03 1 100
60 21.68 1 100
60 22.34 1 100
60 23.0 1 100
60 23.66 1 100
60 24.31 1 100
60 24.97 1 100
60 25.63 1 100
60 26.28 1 100
60 26.94 1 100
60 27.6 1 100
60 28.25 1 100
60 28.91 1 100
60 29.57 1 100
60 30.23 1 100
60 30.88 1 100
60 31.54 1 100
60 32.2 1 100
60 32.85 1 100
60 33.51 1 100
60 34.17 1 100
60 34.83 1 100
60 35.48 1 100
60 36.14 1 100
60 36.8 1 100
60 37.45 1 100
60 38.11 1 100
60 38.77 1 100
60 39.43 1 100
60 40.08 1 100
60 40.74 1 100
60 41.4 1 100
65 42.05 1 100
60 42.71 1 100
60 43.37 1 100
65 44.02 1 100
60 44.68 1 100
60 45.34 1 100
60 46.0 1 100
```

```
71 46.65 1 100
60 47.31 1 100
60 47.97 1 100
65 48.62 1 100
60 49.28 1 100
65 49.94 1 100
65 50.6 1 100
60 51.25 1 100
65 51.91 1 100
60 52.57 1 100
71 53.22 1 100
65 53.88 1 100
65 54.54 1 100
60 55.2 1 100
60 55.85 1 100
60 56.51 1 100
60 57.17 1 100
60 57.82 1 100
60 58.48 1 100
60 59.14 1 100
60 59.79 1 100
60 60.45 1 100
60 61.11 1 100
60 61.77 1 100
60 62.42 1 100
60 63.08 1 100
60 63.74 1 100
65 64.39 1 100
60 65.05 1 100
60 65.71 1 100
60 66.37 1 100
60 67.02 1 100
60 67.68 1 100
60 68.34 1 100
65 68.99 1 100
60 69.65 1 100
60 70.31 1 100
65 70.97 1 100
60 71.62 1 100
60 72.28 1 100
60 72.94 1 100
60 73.59 1 100
65 74.25 1 100
65 74.91 1 100
71 75.56 1 100
60 76.22 1 100
60 76.88 1 100
65 77.54 1 100
```

```
60 78.19 1 100
65 78.85 1 100
60 79.51 1 100
60 80.16 1 100
65 80.82 1 100
60 81.48 1 100
60 82.14 1 100
71 82.79 1 100
60 83.45 1 100
60 84.11 1 100
65 84.76 1 100
65 85.42 1 100
65 86.08 1 100
60 86.74 1 100
60 87.39 1 100
60 88.05 1 100
```

[18]:
```
# HS, C Natural minor:
data2=HS_day.copy()
data2.index=list(range(len(data2)))
data2=data2.to_dict()
data2=[{'beat': d, "overall": data2[d]} for d in data2.keys()]
my_data_timed2 = [{'beat': midi2.beat(d['beat']), 'magnitude': d['overall']}␣
 ↪for d in data2]
note_list2 = []
counter=0
for d in my_data_timed2:
    try:
        note_list2.append([
            d['beat'],
            mag_to_pitch_tuned_Nmin(d['magnitude'], low_HS, high_HS),
            100,  # velocity
            1  # duration, in beats
        ])
        counter +=1
    except Exception as e:
        print(d['beat'])
        print(counter)
        print(e)
        # some notes may do not match
midi2.add_track(note_list2)
midi2.save_midi()
```

```
48 0.0 1 100
48 0.66 1 100
48 1.31 1 100
48 1.97 1 100
```

```
48 2.63 1 100
48 3.29 1 100
48 3.94 1 100
48 4.6 1 100
48 5.26 1 100
48 5.91 1 100
48 6.57 1 100
48 7.23 1 100
48 7.89 1 100
48 8.54 1 100
48 9.2 1 100
48 9.86 1 100
48 10.51 1 100
48 11.17 1 100
48 11.83 1 100
48 12.48 1 100
48 13.14 1 100
48 13.8 1 100
48 14.46 1 100
48 15.11 1 100
48 15.77 1 100
48 16.43 1 100
48 17.08 1 100
48 17.74 1 100
48 18.4 1 100
48 19.06 1 100
48 19.71 1 100
48 20.37 1 100
48 21.03 1 100
48 21.68 1 100
48 22.34 1 100
48 23.0 1 100
48 23.66 1 100
48 24.31 1 100
48 24.97 1 100
48 25.63 1 100
48 26.28 1 100
48 26.94 1 100
48 27.6 1 100
48 28.25 1 100
48 28.91 1 100
48 29.57 1 100
48 30.23 1 100
48 30.88 1 100
48 31.54 1 100
48 32.2 1 100
48 32.85 1 100
48 33.51 1 100
```

```
48 34.17 1 100
48 34.83 1 100
48 35.48 1 100
48 36.14 1 100
48 36.8 1 100
48 37.45 1 100
48 38.11 1 100
48 38.77 1 100
48 39.43 1 100
50 40.08 1 100
48 40.74 1 100
50 41.4 1 100
50 42.05 1 100
48 42.71 1 100
50 43.37 1 100
50 44.02 1 100
50 44.68 1 100
48 45.34 1 100
48 46.0 1 100
50 46.65 1 100
50 47.31 1 100
50 47.97 1 100
50 48.62 1 100
53 49.28 1 100
50 49.94 1 100
48 50.6 1 100
51 51.25 1 100
50 51.91 1 100
51 52.57 1 100
53 53.22 1 100
50 53.88 1 100
51 54.54 1 100
50 55.2 1 100
50 55.85 1 100
53 56.51 1 100
48 57.17 1 100
50 57.82 1 100
51 58.48 1 100
50 59.14 1 100
50 59.79 1 100
50 60.45 1 100
48 61.11 1 100
48 61.77 1 100
50 62.42 1 100
50 63.08 1 100
51 63.74 1 100
50 64.39 1 100
50 65.05 1 100
```

```
48 65.71 1 100
50 66.37 1 100
50 67.02 1 100
50 67.68 1 100
50 68.34 1 100
50 68.99 1 100
50 69.65 1 100
50 70.31 1 100
50 70.97 1 100
50 71.62 1 100
48 72.28 1 100
53 72.94 1 100
50 73.59 1 100
55 74.25 1 100
51 74.91 1 100
50 75.56 1 100
48 76.22 1 100
51 76.88 1 100
51 77.54 1 100
53 78.19 1 100
50 78.85 1 100
53 79.51 1 100
51 80.16 1 100
50 80.82 1 100
53 81.48 1 100
56 82.14 1 100
50 82.79 1 100
50 83.45 1 100
58 84.11 1 100
51 84.76 1 100
56 85.42 1 100
50 86.08 1 100
50 86.74 1 100
50 87.39 1 100
48 88.05 1 100
```

[19]:
```
# EI, C Harmonic minor:
data3=EI_day.copy()
data3.index=list(range(len(data3)))
data3=data3.to_dict()
data3=[{'beat': d, "overall": data3[d]} for d in data3.keys()]
my_data_timed3 = [{'beat': midi3.beat(d['beat']), 'magnitude': d['overall']}␣
 ↪for d in data3]
note_list3 = []
counter=0
for d in my_data_timed3:
    try:
```

```
        note_list3.append([
            d['beat'],
            mag_to_pitch_tuned_Hmin(d['magnitude'], low_EI, high_EI),
            100,  # velocity
            1  # duration, in beats
        ])
        counter +=1
    except Exception as e:
        print(d['beat'])
        print(counter)
        print(e)
        # some notes may do not match
midi3.add_track(note_list3)
midi3.save_midi()
```

```
72 0.0 1 100
72 0.66 1 100
72 1.31 1 100
72 1.97 1 100
72 2.63 1 100
72 3.29 1 100
72 3.94 1 100
72 4.6 1 100
72 5.26 1 100
72 5.91 1 100
72 6.57 1 100
72 7.23 1 100
72 7.89 1 100
72 8.54 1 100
72 9.2 1 100
72 9.86 1 100
72 10.51 1 100
72 11.17 1 100
72 11.83 1 100
72 12.48 1 100
72 13.14 1 100
72 13.8 1 100
72 14.46 1 100
72 15.11 1 100
72 15.77 1 100
72 16.43 1 100
72 17.08 1 100
72 17.74 1 100
72 18.4 1 100
72 19.06 1 100
72 19.71 1 100
72 20.37 1 100
```

```
72 21.03 1 100
72 21.68 1 100
72 22.34 1 100
72 23.0 1 100
72 23.66 1 100
72 24.31 1 100
72 24.97 1 100
72 25.63 1 100
72 26.28 1 100
72 26.94 1 100
72 27.6 1 100
72 28.25 1 100
72 28.91 1 100
72 29.57 1 100
72 30.23 1 100
72 30.88 1 100
72 31.54 1 100
72 32.2 1 100
72 32.85 1 100
72 33.51 1 100
72 34.17 1 100
72 34.83 1 100
72 35.48 1 100
72 36.14 1 100
72 36.8 1 100
72 37.45 1 100
74 38.11 1 100
74 38.77 1 100
74 39.43 1 100
75 40.08 1 100
79 40.74 1 100
74 41.4 1 100
74 42.05 1 100
75 42.71 1 100
75 43.37 1 100
77 44.02 1 100
77 44.68 1 100
79 45.34 1 100
77 46.0 1 100
75 46.65 1 100
77 47.31 1 100
77 47.97 1 100
77 48.62 1 100
72 49.28 1 100
74 49.94 1 100
74 50.6 1 100
83 51.25 1 100
74 51.91 1 100
```

```
77 52.57 1 100
77 53.22 1 100
77 53.88 1 100
74 54.54 1 100
74 55.2 1 100
75 55.85 1 100
75 56.51 1 100
74 57.17 1 100
74 57.82 1 100
72 58.48 1 100
74 59.14 1 100
77 59.79 1 100
75 60.45 1 100
83 61.11 1 100
74 61.77 1 100
72 62.42 1 100
74 63.08 1 100
72 63.74 1 100
75 64.39 1 100
75 65.05 1 100
77 65.71 1 100
75 66.37 1 100
72 67.02 1 100
74 67.68 1 100
75 68.34 1 100
77 68.99 1 100
75 69.65 1 100
75 70.31 1 100
74 70.97 1 100
74 71.62 1 100
74 72.28 1 100
75 72.94 1 100
75 73.59 1 100
77 74.25 1 100
80 74.91 1 100
79 75.56 1 100
75 76.22 1 100
77 76.88 1 100
74 77.54 1 100
77 78.19 1 100
79 78.85 1 100
77 79.51 1 100
77 80.16 1 100
74 80.82 1 100
77 81.48 1 100
80 82.14 1 100
75 82.79 1 100
79 83.45 1 100
```

```
79 84.11 1 100
79 84.76 1 100
74 85.42 1 100
74 86.08 1 100
74 86.74 1 100
74 87.39 1 100
74 88.05 1 100
```

[20]:
```
# DA, C Melodic minor:
data4=DA_day.copy()
data4.index=list(range(len(data4)))
data4=data4.to_dict()
data4=[{'beat': d, "overall": data4[d]} for d in data4.keys()]
my_data_timed4 = [{'beat': midi4.beat(d['beat']), 'magnitude': d['overall']}␣
 ↪for d in data4]
note_list4= []
counter=0
for d in my_data_timed4:
    try:
        note_list4.append([
            d['beat'],
            mag_to_pitch_tuned_Mmin(d['magnitude'], low_DA, high_DA),
            100,  # velocity
            1  # duration, in beats
        ])
        counter +=1
    except Exception as e:
        print(d['beat'])
        print(counter)
        print(e)
        # some notes may do not match
midi4.add_track(note_list4)
midi4.save_midi()
```

```
84 0.0 1 100
84 0.66 1 100
84 1.31 1 100
84 1.97 1 100
84 2.63 1 100
84 3.29 1 100
84 3.94 1 100
84 4.6 1 100
84 5.26 1 100
84 5.91 1 100
84 6.57 1 100
84 7.23 1 100
84 7.89 1 100
```

```
84 8.54 1 100
84 9.2 1 100
84 9.86 1 100
84 10.51 1 100
84 11.17 1 100
84 11.83 1 100
84 12.48 1 100
84 13.14 1 100
84 13.8 1 100
84 14.46 1 100
84 15.11 1 100
84 15.77 1 100
84 16.43 1 100
84 17.08 1 100
84 17.74 1 100
84 18.4 1 100
84 19.06 1 100
84 19.71 1 100
84 20.37 1 100
84 21.03 1 100
84 21.68 1 100
84 22.34 1 100
84 23.0 1 100
84 23.66 1 100
84 24.31 1 100
84 24.97 1 100
84 25.63 1 100
84 26.28 1 100
84 26.94 1 100
84 27.6 1 100
84 28.25 1 100
84 28.91 1 100
84 29.57 1 100
84 30.23 1 100
84 30.88 1 100
84 31.54 1 100
84 32.2 1 100
84 32.85 1 100
84 33.51 1 100
84 34.17 1 100
84 34.83 1 100
84 35.48 1 100
84 36.14 1 100
84 36.8 1 100
84 37.45 1 100
84 38.11 1 100
84 38.77 1 100
84 39.43 1 100
```

```
84 40.08 1 100
84 40.74 1 100
86 41.4 1 100
84 42.05 1 100
84 42.71 1 100
84 43.37 1 100
84 44.02 1 100
84 44.68 1 100
84 45.34 1 100
84 46.0 1 100
86 46.65 1 100
86 47.31 1 100
84 47.97 1 100
84 48.62 1 100
84 49.28 1 100
84 49.94 1 100
84 50.6 1 100
84 51.25 1 100
87 51.91 1 100
84 52.57 1 100
87 53.22 1 100
84 53.88 1 100
86 54.54 1 100
84 55.2 1 100
86 55.85 1 100
84 56.51 1 100
84 57.17 1 100
86 57.82 1 100
84 58.48 1 100
86 59.14 1 100
84 59.79 1 100
84 60.45 1 100
86 61.11 1 100
84 61.77 1 100
84 62.42 1 100
87 63.08 1 100
84 63.74 1 100
84 64.39 1 100
87 65.05 1 100
84 65.71 1 100
89 66.37 1 100
91 67.02 1 100
87 67.68 1 100
86 68.34 1 100
89 68.99 1 100
89 69.65 1 100
93 70.31 1 100
89 70.97 1 100
```

```
87 71.62 1 100
95 72.28 1 100
86 72.94 1 100
87 73.59 1 100
86 74.25 1 100
87 74.91 1 100
89 75.56 1 100
84 76.22 1 100
87 76.88 1 100
89 77.54 1 100
93 78.19 1 100
84 78.85 1 100
91 79.51 1 100
91 80.16 1 100
87 80.82 1 100
91 81.48 1 100
95 82.14 1 100
91 82.79 1 100
93 83.45 1 100
91 84.11 1 100
87 84.76 1 100
93 85.42 1 100
89 86.08 1 100
86 86.74 1 100
87 87.39 1 100
87 88.05 1 100
```

PART 2: Simple MASH UP

[21]:
```
from mido import MidiFile
```

[22]:
```
AU_mid=MidiFile('results/Texas/AU.mid')
DA_mid=MidiFile('results/Texas/DA.mid')
EI_mid=MidiFile('results/Texas/EI.mid')
HS_mid=MidiFile('results/Texas/HS.mid')
```

[23]:
```
AU_mid.tracks.extend(DA_mid.tracks)
AU_mid.tracks.extend(EI_mid.tracks)
AU_mid.tracks.extend(HS_mid.tracks)
AU_mid.save('results/Texas.mid')
```

[ ]:
```

```