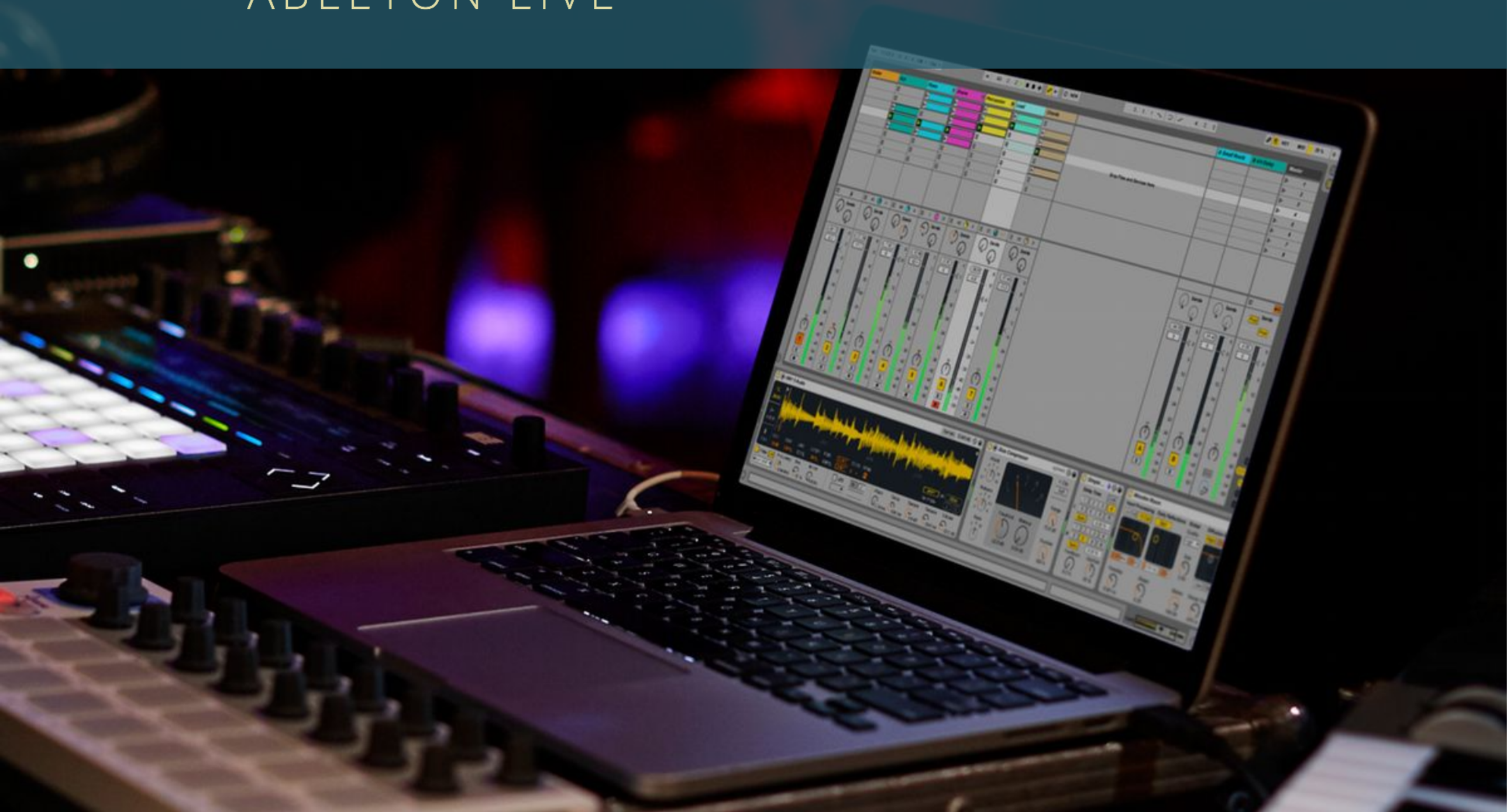
MUSIC TRANSCRIPTION WITH PYTHON

ANNA WSZEBOROWSKA

 @aniawsz

ABLETON

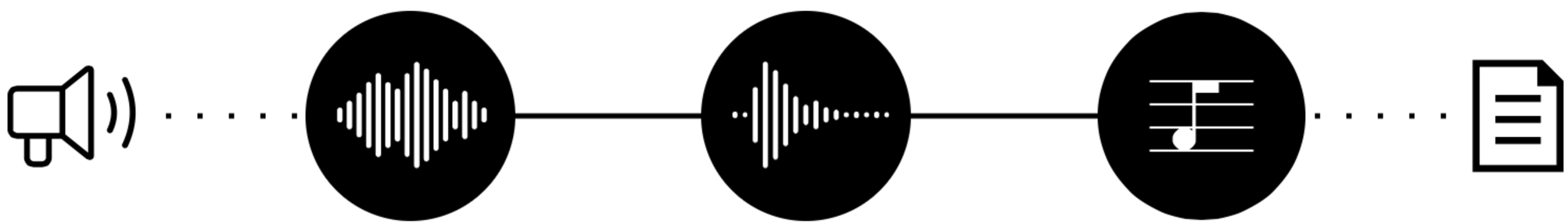
ABLETON LIVE



LINK

PUSH

TO TRANSCRIBE MUSIC



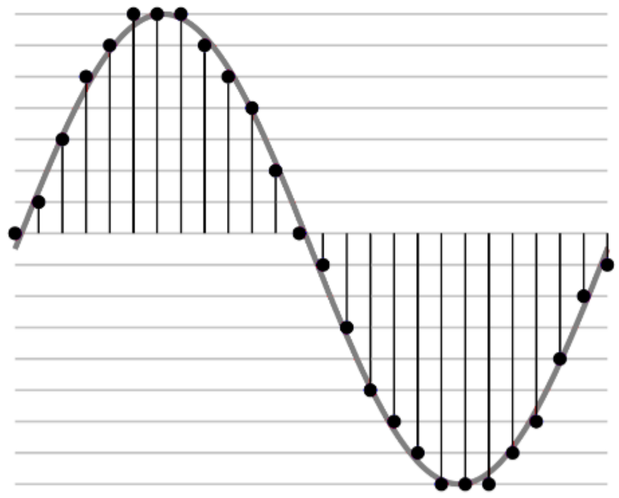
READ AUDIO STREAM

DETECT NOTES

CONVERT TO
MUSIC NOTATION

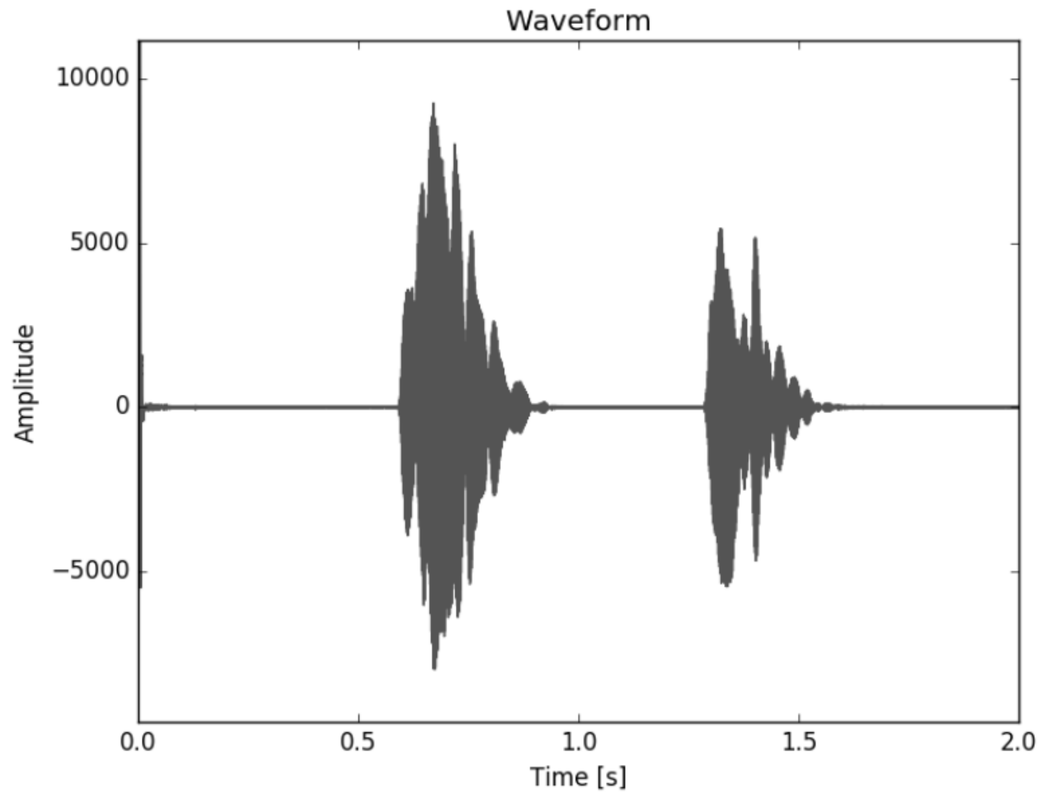
Used icons created by Nick Bluth and Kid A from Noun Project

Q1: HOW TO READ AND STORE DATA?



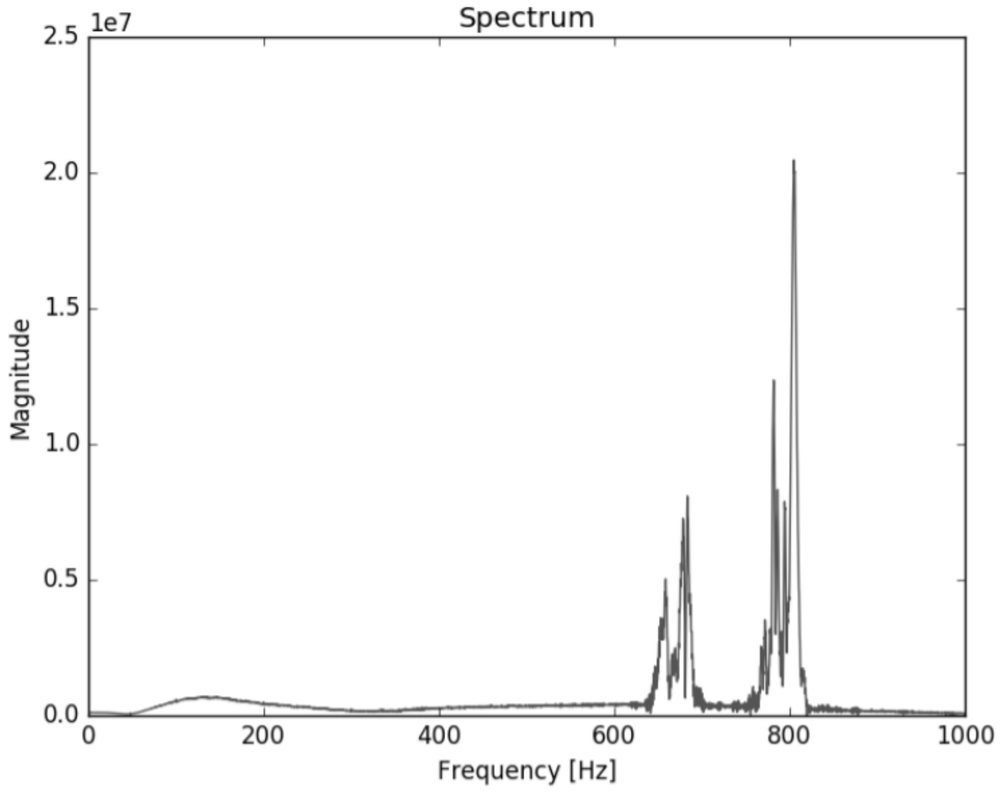
SAMPLING AND QUANTIZATION

Q2: HOW TO DETECT NOTES?



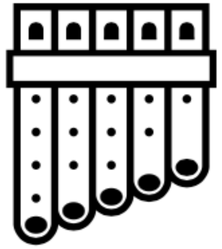
FIND ONSETS

Q2: HOW TO DETECT NOTES?



IDENTIFY PITCH

Q3: HOW TO REPRESENT A NOTE?



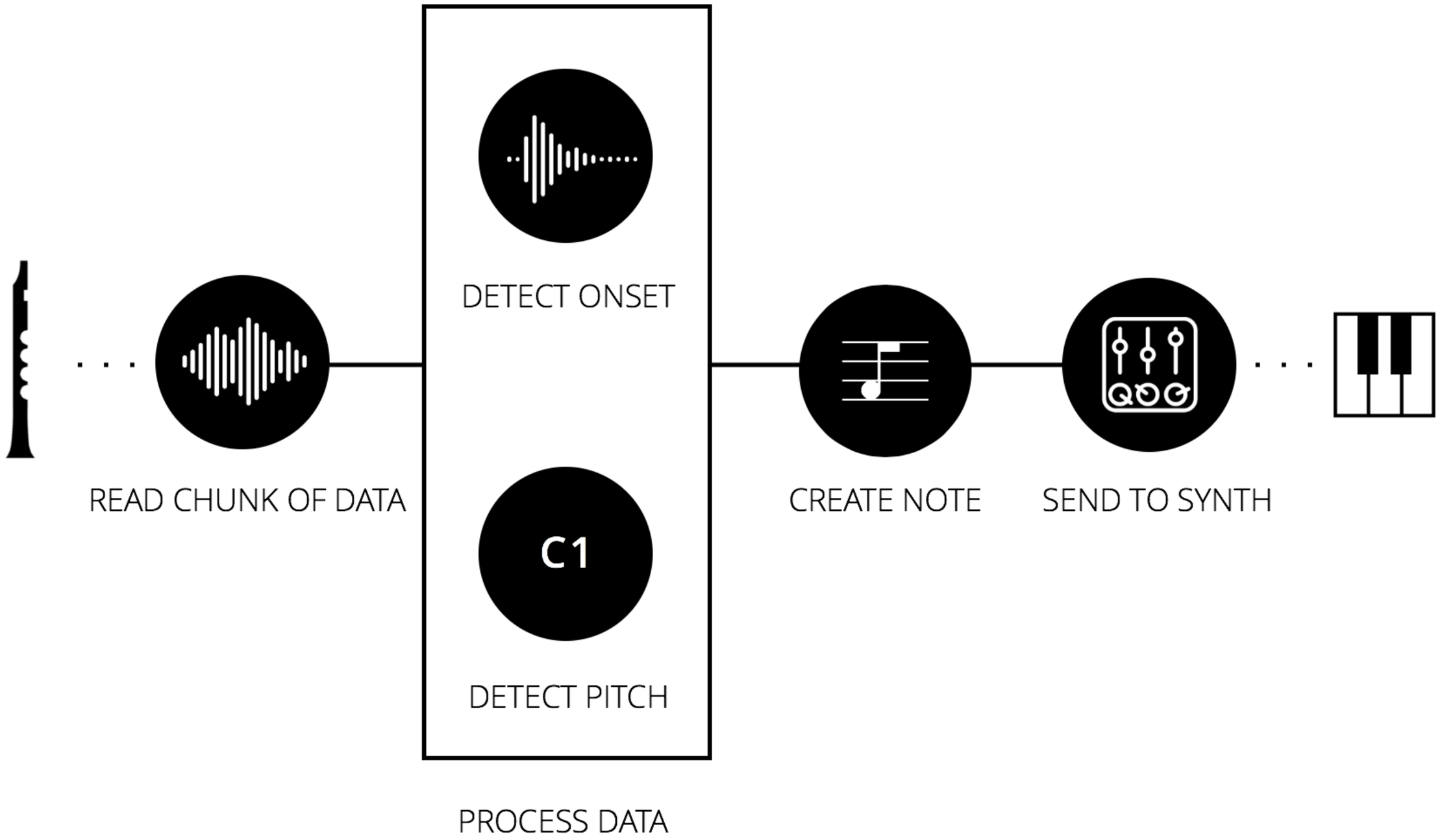
Q1: HOW TO READ AND STORE DATA?

Q2: HOW TO DETECT NOTES?

Q3: HOW TO REPRESENT A NOTE?

DEMO

(WHAT COULD POSSIBLY GO WRONG)



Used icons created by Nick Bluth, Kid A and iconsmind.com from Noun Project

READING DATA

```
pya = PyAudio()
self._stream = pya.open(
    format=paInt16,
    channels=1,
    rate=SAMPLE_RATE,
    input=True,
    frames_per_buffer=WINDOW_SIZE,
    stream_callback=self._process_frame,
)
self._stream.start_stream()
```

PYAUDIO - PYTHON BINDING FOR PORTAUDIO

READING DATA

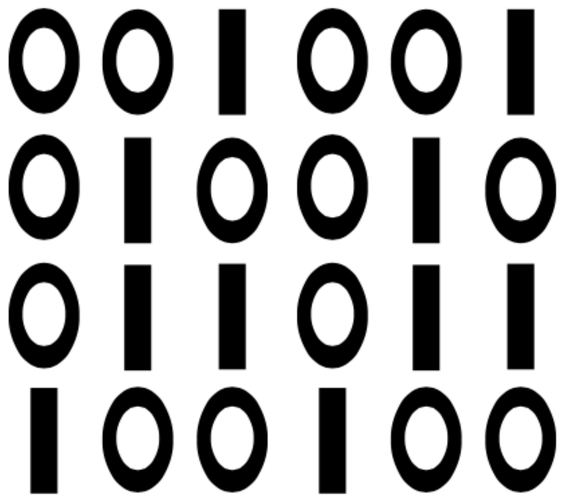
```
def callback(self, data, frame_count, time_info, status_flag):
    data_array = np.fromstring(data, dtype=np.int16)
    self._spectral_analyser.process_data(data_array)
    return (data, paContinue)
```

NON-BLOCKING MODE

CALLBACK CALLED IN A SEPARATE THREAD

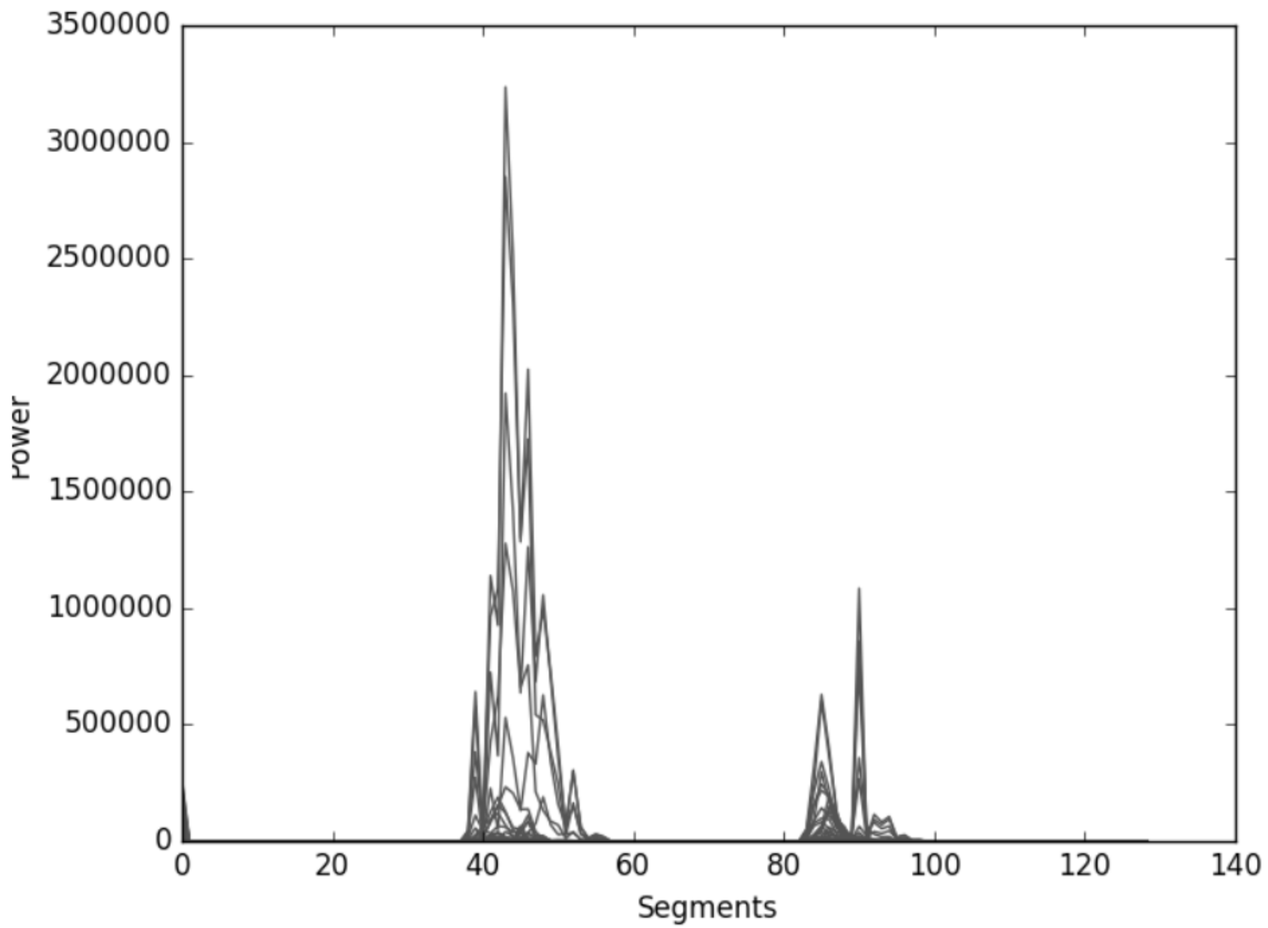
MUST RETURN FRAME_COUNT FRAMES AND A FLAG

STORING DATA



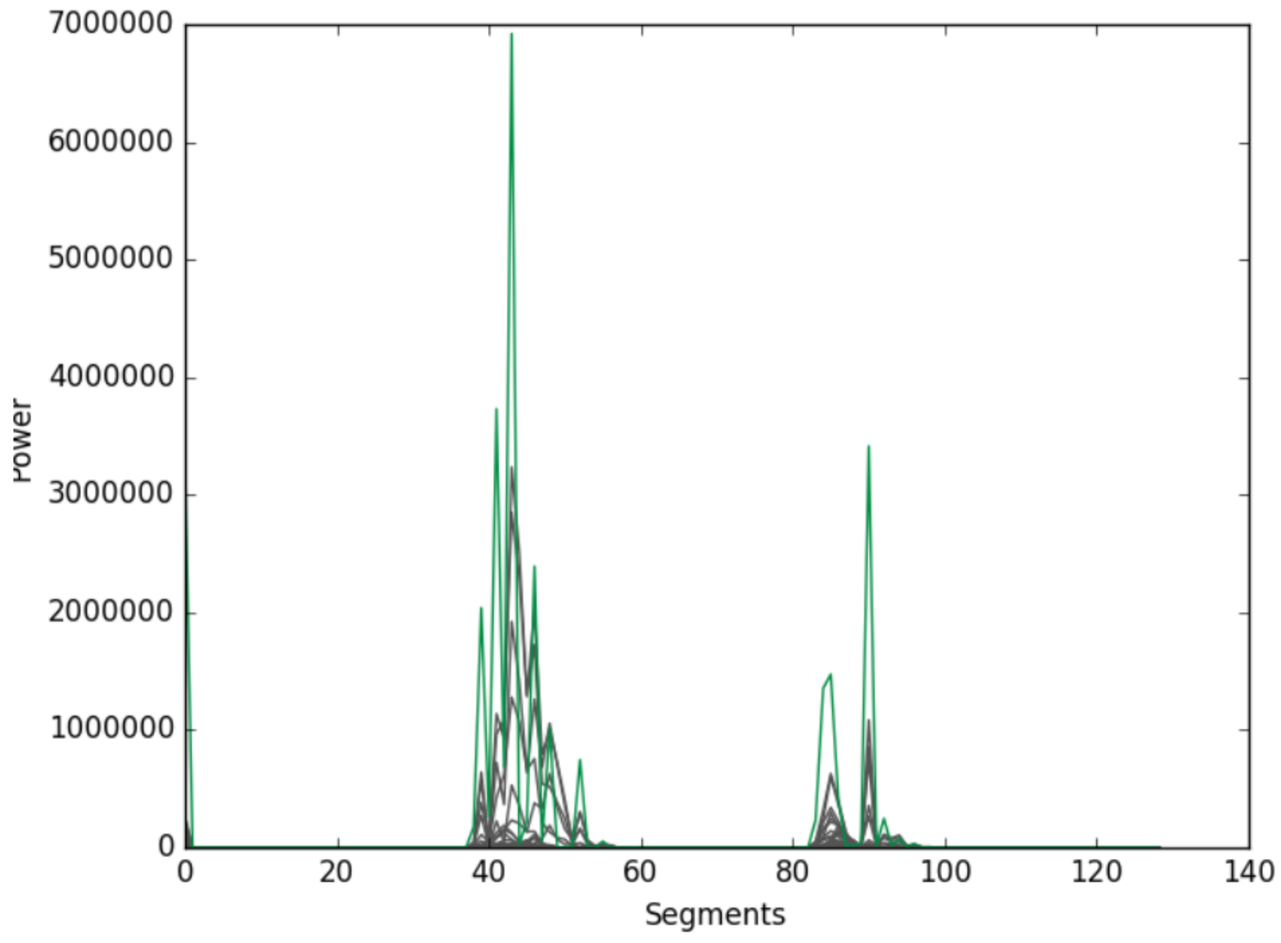
NUMPY ARRAYS

ONSET DETECTION



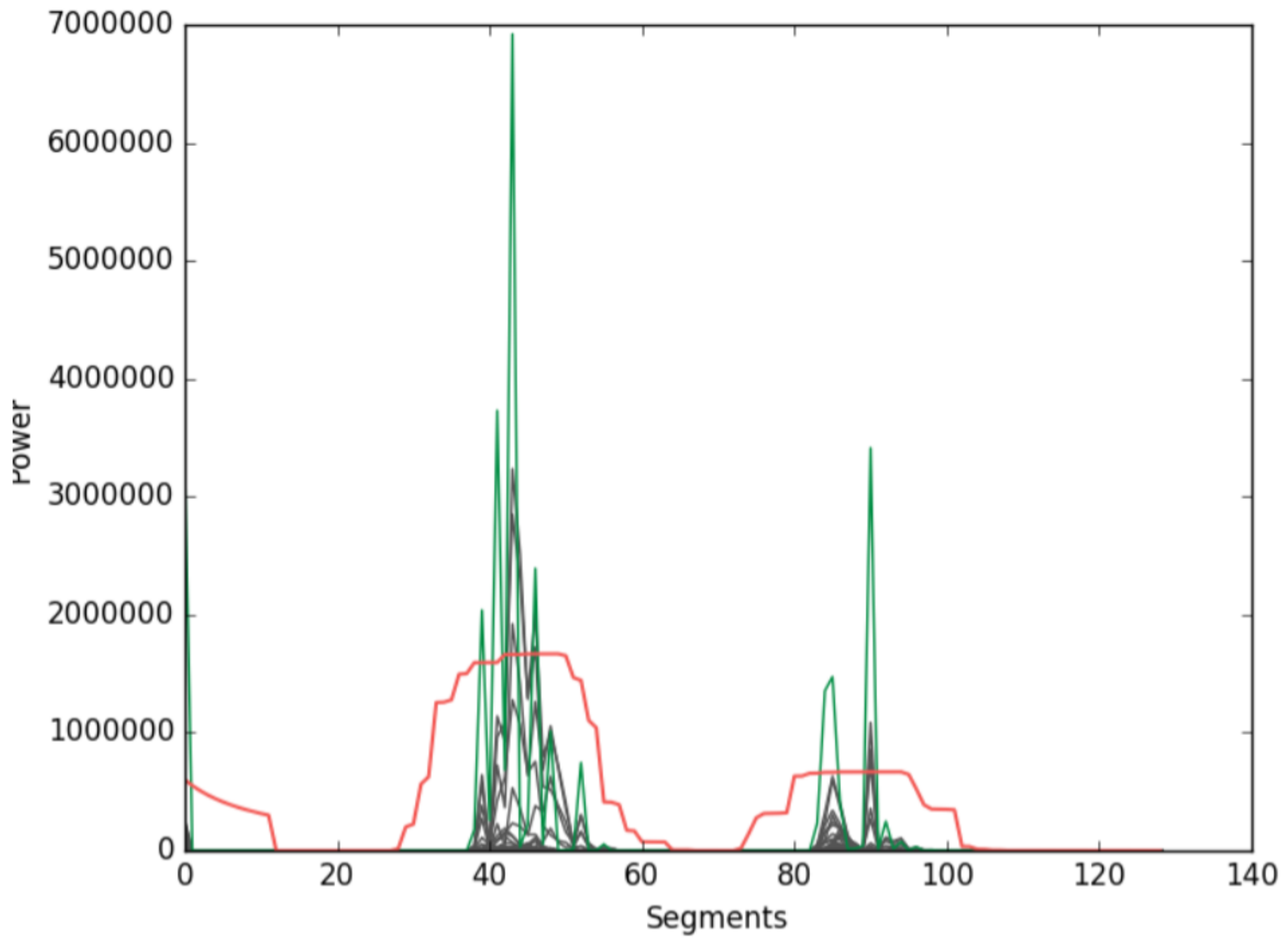
SHORT-TIME FOURIER TRANSFORM

ONSET DETECTION



SPECTRAL FLUX

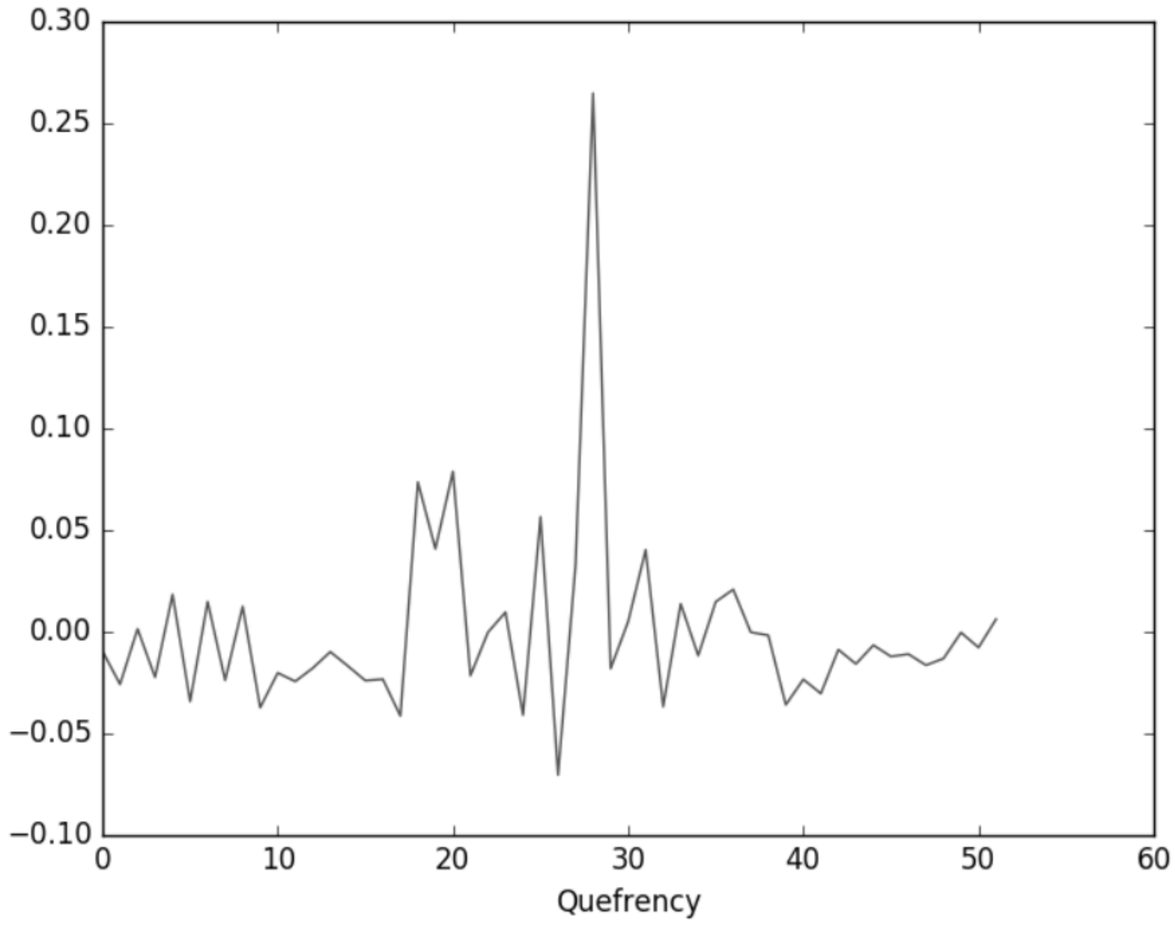
ONSET DETECTION



PICKING PEAKS

THRESHOLDING

PITCH DETECTION



TRIGGERED WHEN ONSET FOUND

CALCULATE **CEPSTRUM**
LIMITED TO A DESIRED RANGE

PITCH DETECTION

FIND MAX VALUE IN THE NARROWED CEPSTRUM

CONVERT QUEFRENCY TO FREQUENCY

fundamental_frequncy = sample_rate / cepstrum_peak_ix

PITCH DETECTION

```
start = int(fs / 1200)  # 36
end = int(fs / 500)  # 88
narrowed_cepstrum = cepstrum[start:end]

peak_ix = narrowed_cepstrum.argmax()  # 28
freq0 = fs / (start + peak_ix)  # 689 Hz
```

CORRECTION

IGNORE ONSETS OUT OF THE FREQUENCY RANGE

NOTES CREATION

MIDI PROTOCOL

NOTE ON MESSAGE

NOTES CREATION

FREQUENCY TO MIDI NOTE CONVERSION

$$d = 69 + 12 \log_2 \left(\frac{f}{440\ \mathrm{Hz}} \right)$$

MIDI NOTES NUMBER TO FREQUENCY CONVERSION CHART

C4	36	65.4063913251	48	130.8127826503	60	261.6255653006
Db	37	69.2956577442	49	138.5913154884	61	277.1826309769
D	38	73.4161919794	50	146.8323839587	62	293.6647679174
Eb	39	77.7817459305	51	155.5634918610	63	311.1269837221
E	40	82.4068892282	52	164.8137784564	64	329.6275569129
F	41	87.3070578583	53	174.6141157165	65	349.2282314330
Gb	42	92.4986056779	54	184.9972113558	66	369.9944227116
G	43	97.9988589954	55	195.9977179909	67	391.9954359817
Ab	44	103.8261743950	56	207.6523487900	68	415.3046975799
A	45	110.0000000000	57	220.0000000000	69	440.0000000000
Bb	46	116.5409403795	58	233.0818807590	70	466.1637615181
B	47	123.4708253140	59	246.9416506281	71	493.8833012561
C7	72	523.2511306012	84	1046.5022612024	96	2093.0045224048
Db	73	554.3652619537	85	1108.7305239075	97	2217.4610478150
D	74	587.3295358348	86	1174.6590716696	98	2349.3181433393
Eb	75	622.2539674442	87	1244.5079348883	99	2489.0158697766
E	76	659.2551138257	88	1318.5102276515	100	2637.0204553030
F	77	698.4564628660	89	1396.9129257320	101	2793.8258514640
Gb	78	739.9888454233	90	1479.9776908465	102	2959.9553816931
G	79	783.9908719635	91	1567.9817439270	103	3135.9634878540
Ab	80	830.6093951599	92	1661.2187903198	104	3322.4375806396
A	81	880.0000000000	93	1760.0000000000	105	3520.0000000000
Bb	82	932.3275230362	94	1864.6550460724	106	3729.3100921447
B	83	987.7666025122	95	1975.5332050245	107	3951.0664100490

NOTE PLAYED BY A DIFFERENT INSTRUMENT

PYFLUIDSYNTH

SOUND FONT

CONCLUSIONS

PYTHON FOR RAPID PROTOTYPING

NUMERICAL LIBRARIES

SIMPLE I/O OPERATIONS

GOOD API OF USED WRAPPERS