INSTITUTO INFNET

ENGENHARIA DE COMPUTAÇÃO

ALEXANDER VIEIRA DA SILVA

MATRÍCULA 041.380.007-55

PROJETO DE BLOCO
PROCESSAMENTO DIGITAL DE SINAIS
APRESENTAÇÃO

RIO DE JANEIRO-RJ

2020

APRESENTAÇÃO

```
[ ] pip install thinkx
```

```
[10] from __future__ import print_function, division

     import thinkdsp
     import thinkplot
     import thinkstats2
     import numpy as np

     import warnings
     warnings.filterwarnings('ignore')

     from ipywidgets import interact, interactive, fixed
     import ipywidgets as widgets

     PI2 = np.pi * 2

     %matplotlib inline

     from google.colab import files
```

```
[12] uploaded = files.upload()

     for fn in uploaded.keys():
       print('User uploaded file "{name}" with length {length} bytes'.format(name=fn, length=len(uploaded[fn])))
```

```
Escolher arquivos | Trombone_Do.wav
• Trombone_Do.wav(audio/wav) - 176444 bytes, last modified: 10/02/2020 - 100% done
Saving Trombone_Do.wav to Trombone_Do.wav
User uploaded file "Trombone_Do.wav" with length 176444 bytes
```

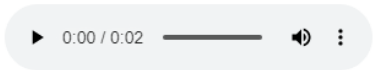
1. Tocar o áudio dos instrumentos escolhidos, separadamente, em um determinado tom (frequência);

BERIMBAU

```
[13] wave_berimbau = thinkdsp.read_wave('berimbau_sem_pedra.wav')
     wave_berimbau.normalize()
     wave_berimbau.make_audio()
```

TROMBONE

```
[14] wave_trombone = thinkdsp.read_wave('Trombone_Do.wav')
     wave_trombone.normalize()
     wave_trombone.make_audio()
```



2. Representar graficamente, no domínio do tempo, o sinal de áudio, para a representação do timbre (curva específica do instrumento).

BERIMBAU

```
[13] wave_berimbau = thinkdsp.read_wave('berimbau_sem_pedra.wav')
     wave_berimbau.normalize()
     wave_berimbau.make_audio()
```

0:00 / 0:03

TROMBONE

```
[14] wave_trombone = thinkdsp.read_wave('Trombone_Do.wav')
     wave_trombone.normalize()
     wave_trombone.make_audio()
```

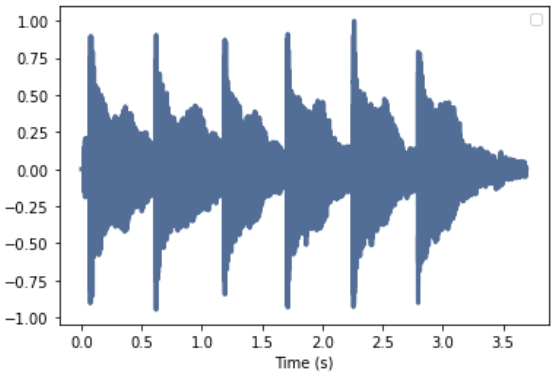
0:00 / 0:02

2. Representar graficamente, no domínio do tempo, o sinal de áudio, para a representação do timbre (curva específica do instrumento).

BERIMBAU

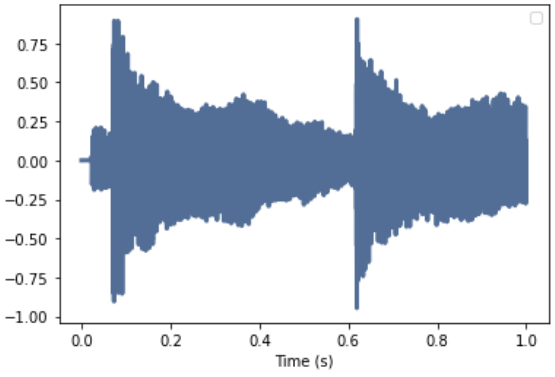
```
[ ] wave_berimbau.plot()
    thinkdsp.thinkplot.Config(xlabel='Time (s)')
```

```
No handles with labels found to put in legend.
```




```
[ ] segment_berimbau = wave_berimbau.segment(start=0, duration=1)
    segment_berimbau.plot()
    thinkdsp.thinkplot.Config(xlabel='Time (s)')
```

```
No handles with labels found to put in legend.
```

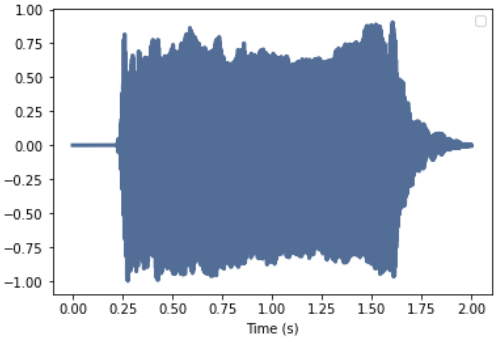


```
[ ]
segment_berimbau.make_audio()
```

TROMBONE

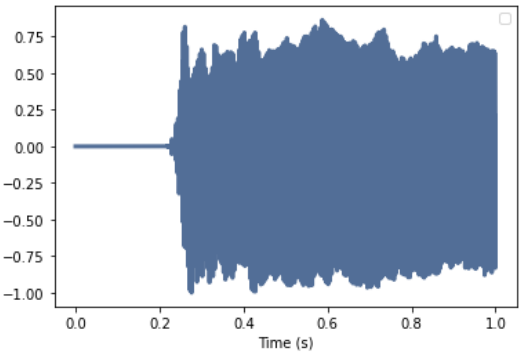
```
[ ] wave_trombone.plot()
thinkdsp.thinkplot.Config(xlabel='Time (s)')
```

```
No handles with labels found to put in legend.
```




```
[ ] segment_trombone = wave_trombone.segment(start=0, duration=1)
segment_trombone.plot()
thinkdsp.thinkplot.Config(xlabel='Time (s)')
```

```
No handles with labels found to put in legend.
```



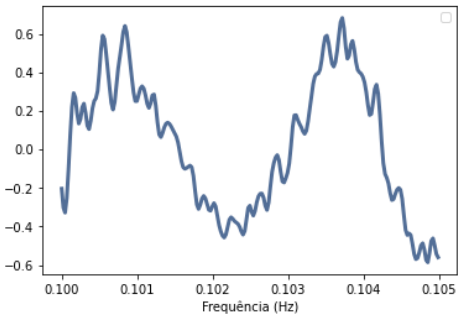

```
[ ]
segment_trombone.make_audio()
```

3. Realizar a análise espectral de cada instrumento e verificar a frequência fundamental e os seus harmônicos;

BERIMBAU

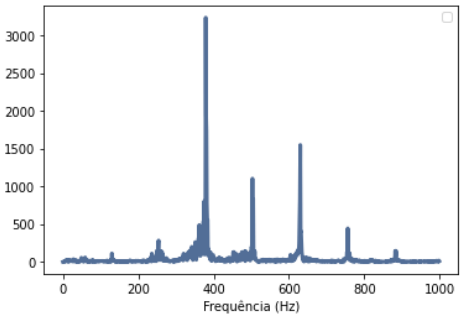
```
[39]
segment_berimbau.segment(start=0.1, duration=0.005).plot()
thinkdsp.thinkplot.Config(xlabel='Frequência (Hz)')
```

```
No handles with labels found to put in legend.
```




```
[40] spectrum_berimbau = segment_berimbau.make_spectrum()
spectrum_berimbau.plot(1000)
thinkdsp.thinkplot.Config(xlabel='Frequência (Hz)')
```

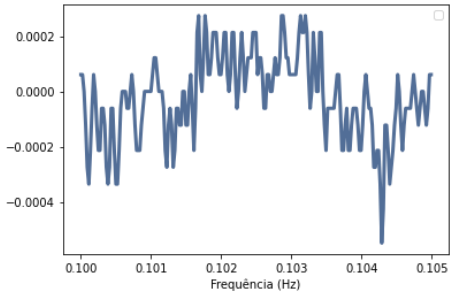
```
No handles with labels found to put in legend.
```



TROMBONE

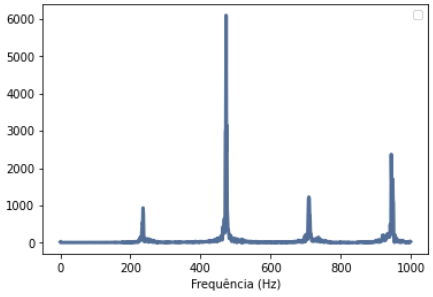
```
[41] segment_trombone.segment(start=0.1, duration=0.005).plot()
thinkdsp.thinkplot.Config(xlabel='Frequência (Hz)')
```

```
No handles with labels found to put in legend.
```




```
[42] spectrum_trombone = segment_trombone.make_spectrum()
spectrum_trombone.plot(1000)
thinkdsp.thinkplot.Config(xlabel='Frequência (Hz)')
```

```
No handles with labels found to put in legend.
```



3.1. Frequência fundamental

BERIMBAU

```
[ ]  spectrum_berimbau.peaks()[:1]
```

```
[(3239.945056097351, 379.0)]
```

TROMBONE

```
[ ]  spectrum_trombone.peaks()[:1]
```

```
[(6095.543520644917, 473.0)]
```

3.1.2. Lista com a frequência fundamental e seus harmônicos.

BERIMBAU

```
[ ]  spectrum_berimbau.peaks()[:30]
```

```
[(3239.945056097351, 379.0),
 (1952.116238241911, 380.0),
 (1731.5643303039024, 381.0),
 (1552.8340402919616, 630.0),
 (1107.3873225820032, 503.0),
 (839.4683095498765, 505.0),
 (831.6771384427108, 629.0),
 (808.6167099658628, 375.0),
 (776.23441584193, 382.0),
 (616.5496671377128, 377.0),
 (575.3059943477497, 383.0),
 (563.3641749460816, 384.0),
 (541.4622328146928, 373.0),
 (517.595421136577, 628.0),
 (492.58901914303124, 362.0),
 (471.31823115347936, 1517.0),
 (452.5951072296196, 370.0),
 (448.4074581735523, 756.0),
 (434.9153338947309, 378.0),
 (432.17787951853103, 364.0),
 (423.7380170362356, 366.0),
 (409.51371297392126, 632.0),
 (399.5992181994902, 631.0),
 (392.9711629683038, 371.0),
 (385.50042820904343, 368.0),
 (344.4685249265738, 372.0),
 (341.6948311687631, 361.0),
 (332.9355055581315, 1135.0),
 (316.0615250729609, 360.0),
 (293.953426448522, 359.0)]
```

TROMBONE

```
[ ] spectrum_trombone.peaks()[:30]
```

```
[(6095.543520644917, 473.0),
 (3144.133397262977, 475.0),
 (3044.6938772157887, 472.0),
 (2973.2157722029783, 471.0),
 (2464.236321006656, 474.0),
 (2370.580535153125, 944.0),
 (1726.3764669362968, 946.0),
 (1703.4523316993705, 948.0),
 (1691.947430039529, 945.0),
 (1455.5750806144447, 1654.0),
 (1226.999585567578, 709.0),
 (1183.6807216821733, 710.0),
 (1179.8086613692449, 708.0),
 (1166.0028304616158, 947.0),
 (1159.3447772751808, 949.0),
 (1067.8375796382932, 1653.0),
 (1058.4665688308585, 950.0),
 (1038.2769970003624, 1418.0),
 (974.2304894287721, 1181.0),
 (936.1484209988115, 236.0),
 (914.559573834853, 1417.0),
 (896.7359902997986, 1652.0),
 (887.1075412072023, 476.0),
 (876.1951898809913, 1179.0),
 (793.299212886557, 711.0),
 (764.3808298364106, 707.0),
 (761.0841839141418, 237.0),
 (739.0231073053474, 712.0),
 (713.3966107027278, 1178.0),
 (707.1151445452455, 470.0)]
```

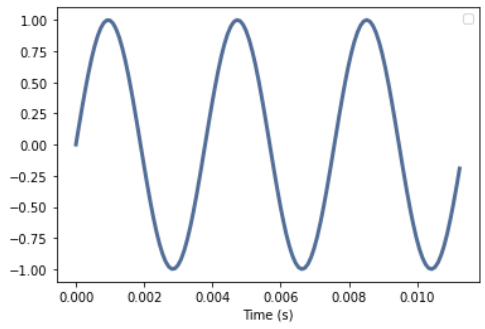
4. Simular, por soma de senos e cossenos, o sinal de áudio, simulando o timbre, ou variar a frequência dos áudios apresentados no primeiro item;

BERIMBAU

4.1. Sinal da nota musical DÓ.

```
[ ] signal_nota_do = thinkdsp.SinSignal(freq=264, amp=1, offset=0)
    signal_nota_do.plot()
    thinkdsp.thinkplot.Config(xlabel='Time (s)')
```

```
No handles with labels found to put in legend.
```



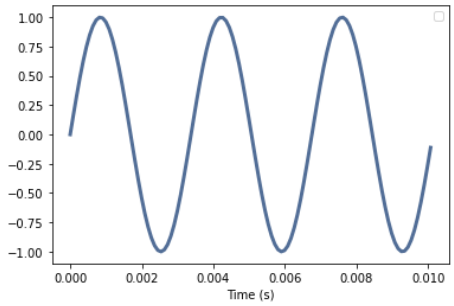
4.1.2. Áudio nota DÓ.

```
[ ] wave_signal_nota_do = signal_nota_do.make_wave(duration=1, start=0, framerate=11025)
    wave_signal_nota_do.make_audio()
```

4.2. Sinal da nota musical RÉ.

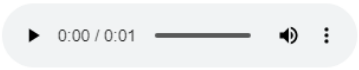
```
[ ] signal_nota_re = thinkdsp.SinSignal(freq=296.2, amp=1, offset=0)
    signal_nota_re.plot()
    thinkdsp.thinkplot.Config(xlabel='Time (s)')
```

```
No handles with labels found to put in legend.
```



4.2.1. Áudio nota RÉ.

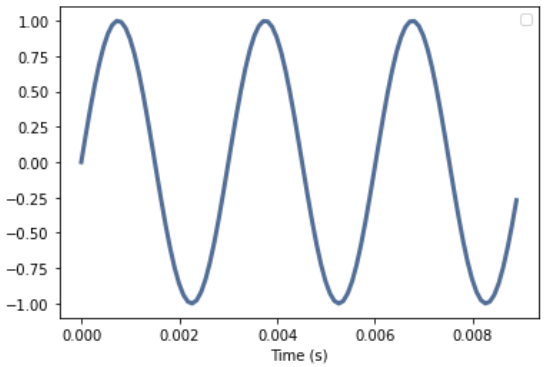
```
[ ] wave_signal_nota_re = signal_nota_re.make_wave(duration=1, start=0, framerate=11025)
    wave_signal_nota_re.make_audio()
```



4.3. Sinal da nota musical MI.

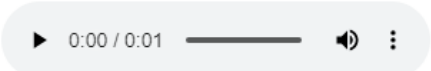
```
[ ] signal_nota_mi = thinkdsp.SinSignal(freq=332.6, amp=1, offset=0)
    signal_nota_mi.plot()
    thinkdsp.thinkplot.Config(xlabel='Time (s)')
```

```
No handles with labels found to put in legend.
```



4.3.1. Áudio nota MI.

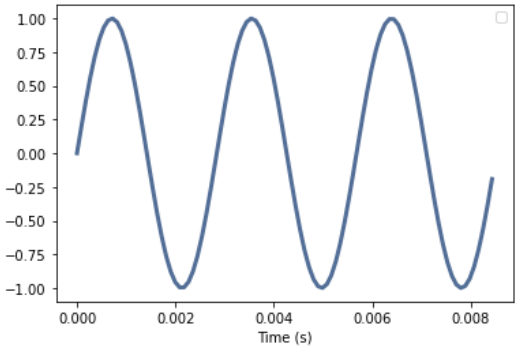
```
[ ] wave_signal_nota_mi = signal_nota_mi.make_wave(duration=1, start=0, framerate=11025)
    wave_signal_nota_mi.make_audio()
```



4.4. Sinal da nota musical FA.

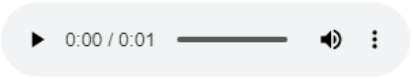
```
[ ] signal_nota_fa = thinkdsp.SinSignal(freq=352, amp=1, offset=0)
    signal_nota_fa.plot()
    thinkdsp.thinkplot.Config(xlabel='Time (s)')
```

```
No handles with labels found to put in legend.
```



4.4.1. Áudio nota FA.

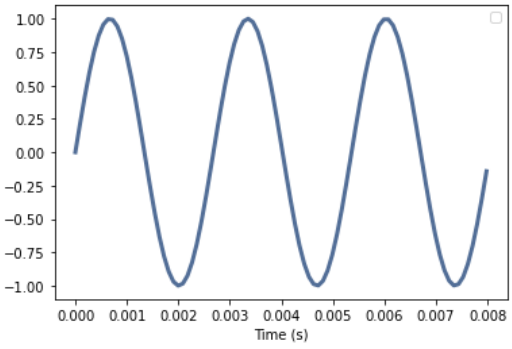
```
[ ] wave_signal_nota_fa = signal_nota_fa.make_wave(duration=1, start=0, framerate=11025)
    wave_signal_nota_fa.make_audio()
```



4.5. Sinal da nota musical SOL.

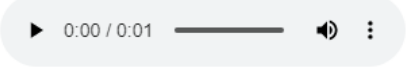
```
[ ] signal_nota_sol = thinkdsp.SinSignal(freq=373, amp=1, offset=0)
    signal_nota_sol.plot()
    thinkdsp.thinkplot.Config(xlabel='Time (s)')
```

```
No handles with labels found to put in legend.
```



4.5.1. Áudio nota SOL.

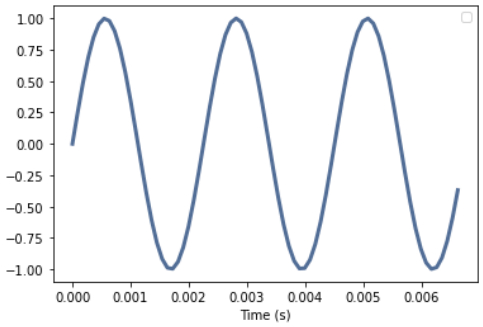
```
[ ] wave_signal_nota_sol = signal_nota_sol.make_wave(duration=1, start=0, framerate=11025)
    wave_signal_nota_sol.make_audio()
```



4.6. Sinal da nota musical LÁ.

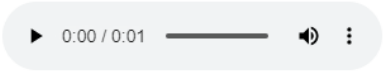
```
[ ] signal_nota_la = thinkdsp.SinSignal(freq=444, amp=1, offset=0)
    signal_nota_la.plot()
    thinkdsp.thinkplot.Config(xlabel='Time (s)')
```

```
No handles with labels found to put in legend.
```



4.6.1 Áudio nota LA.

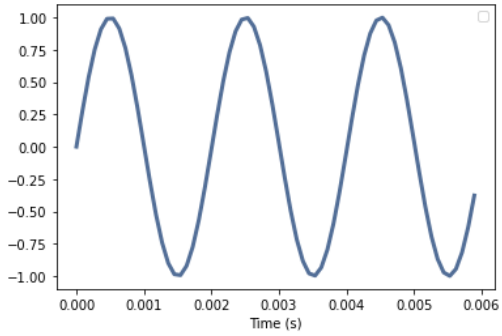
```
[ ] wave_signal_nota_la = signal_nota_la.make_wave(duration=1, start=0, framerate=11025)
    wave_signal_nota_la.make_audio()
```



4.7. Sinal da nota musical SÍ.

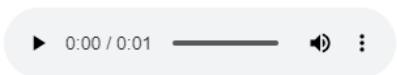
```
[ ] signal_nota_si = thinkdsp.SinSignal(freq=498.4, amp=1, offset=0)
    signal_nota_si.plot()
    thinkdsp.thinkplot.Config(xlabel='Time (s)')
```

```
No handles with labels found to put in legend.
```



4.7.1. Áudio nota SI.

```
[ ] wave_signal_nota_si = signal_nota_si.make_wave(duration=1, start=0, framerate=11025)
    wave_signal_nota_si.make_audio()
```

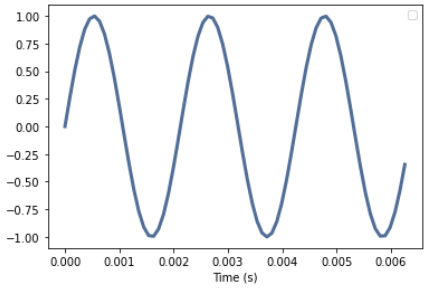


TROMBONE

4.8. Sinal da nota musical LA#.

```
[20] signal_nota_la_trombone = thinkdsp.SinSignal(freq=470.4, amp=1, offset=0)
     signal_nota_la_trombone.plot()
     thinkdsp.thinkplot.Config(xlabel='Time (s)')
```

```
No handles with labels found to put in legend.
```



4.8.1. Áudio nota LA#.

```
[22] wave_signal_nota_la_trombone = signal_nota_la_trombone.make_wave(duration=1, start=0, framerate=11025)
     wave_signal_nota_la_trombone.make_audio()
```

5. Correlação

5.1. Sinal do berimbau

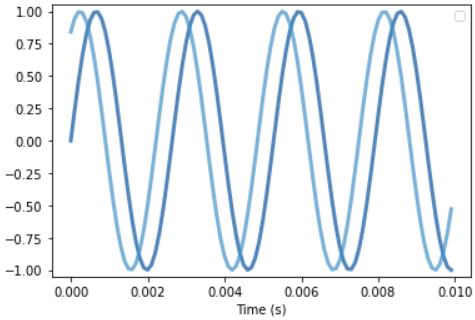
```
[24] def make_signal_nota(signal):
         wave = signal.make_wave(duration=0.5, framerate=10000)
         return wave
```

```
[25] def make_sine(offset, freq):
         signalGenered = thinkdsp.SinSignal(freq=freq, offset=offset)
         wave = signalGenered.make_wave(duration=0.5, framerate=10000)
         return wave
```

```
[ ]  wave1 = make_signal_nota(signal_nota_sol)
     wave2 = make_sine(offset=1, freq=379)

     thinkplot.preplot(2)
     wave1.segment(duration=0.01).plot()
     wave2.segment(duration=0.01).plot()
     thinkplot.config(xlabel='Time (s)', ylim=[-1.05, 1.05])
```

```
No handles with labels found to put in legend.
```



Resultado da correlação

```
[ ]  wave1.corr(wave2)
```

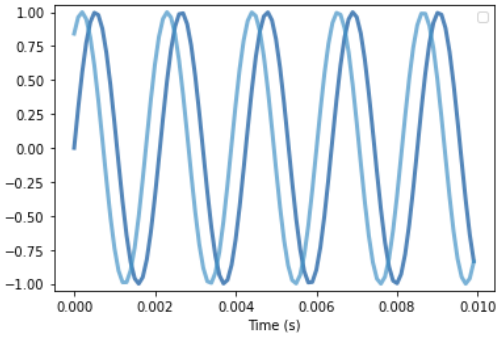
```
0.5403008534808665
```

5.2. Sinal do trombone

```
[26] wave3 = make_signal_nota(signal_nota_la_trombone)
     wave4 = make_sine(offset=1, freq=473)

     thinkplot.preplot(2)
     wave3.segment(duration=0.01).plot()
     wave4.segment(duration=0.01).plot()
     thinkplot.config(xlabel='Time (s)', ylim=[-1.05, 1.05])
```

```
No handles with labels found to put in legend.
```



Resultado da correlação

```
[27] wave3.corr(wave4)
```

```
-0.07126814653468377
```