Adam Kala xkalaa00

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ FAKULTA INFORMAČNÍCH TECHNOLOGIÍ

Protokol s řešením – vygerenovaný Python notebook

ISS Projekt 2022/2023

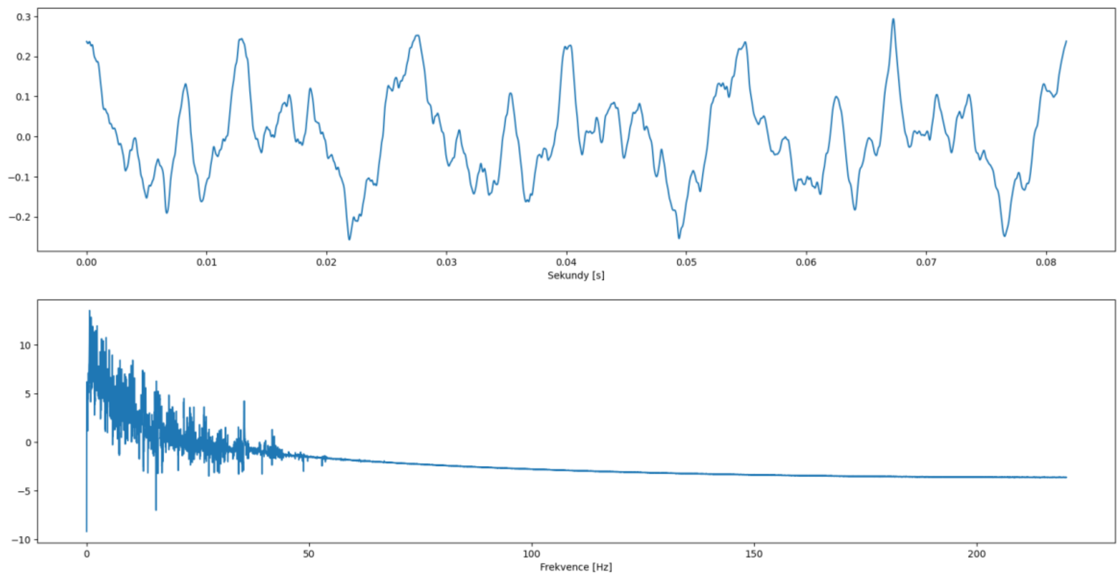
Adam Kala xkalaa00

```python
#imports
import numpy as np
import matplotlib.pyplot as plt
import soundfile as sf
import scipy.signal as ss

#hlavni inicializace
klavir, Fs = sf.read("klavir.wav")

#26     36.71       1/36.71     0.02724053
#51     155.56      1/155.56    0.00642839
#104    3322.44     1/3322.44   0.00030098
MIDIFROM = 24
MIDITO = 108
SKIP_SEC = 0.25
HOWMUCH_SEC = 0.5
WHOLETONE_SEC = 2
howmanytones = MIDITO - MIDIFROM + 1
tones = np.arange(MIDIFROM, MIDITO+1)
N = int(Fs * HOWMUCH_SEC)
Nwholetone = int(Fs * WHOLETONE_SEC)
xall = np.zeros((MIDITO+1, N)) # matrix with all tones - first signals empty,
# but we have plenty of memory ...
samplefrom = int(SKIP_SEC * Fs)
sampleto = samplefrom + N
for tone in tones:
    x = klavir[samplefrom:sampleto]
    x = x - np.mean(x) # safer to center ...
    xall[tone,:] = x
    samplefrom += Nwholetone
    sampleto += Nwholetone

perioda1 = int(0.02724053*Fs*3)
perioda2 = int(0.00642839*Fs*3)
perioda3 = int(0.00030098*Fs*3)
start1 = xall[26]
start2 = xall[51]
start3 = xall[104]
second = np.arange(0, 0.1, 1/Fs)
```

Adam Kala xkalaa00

```
#ukol 4.1.1
_, tony = plt.subplots(2,1,figsize=(20, 10))
tony[0].plot(second[:3922], start1[:perioda1])
tony[0].set_xlabel("Sekundy [s]")
FFT = np.fft.fft(start1)
log = np.log(10e-5+np.abs(FFT)**2) #zlogaritmuji, abych vytvoril spektrum
moduleHalf = log[:log.size // 2]
F = np.arange(moduleHalf.size)*(Fs /xall.size)
sf.write("../a_orig.wav", xall[26], Fs)
tony[1].plot(F, moduleHalf)
tony[1].set_xlabel("Frekvence [Hz]")
```

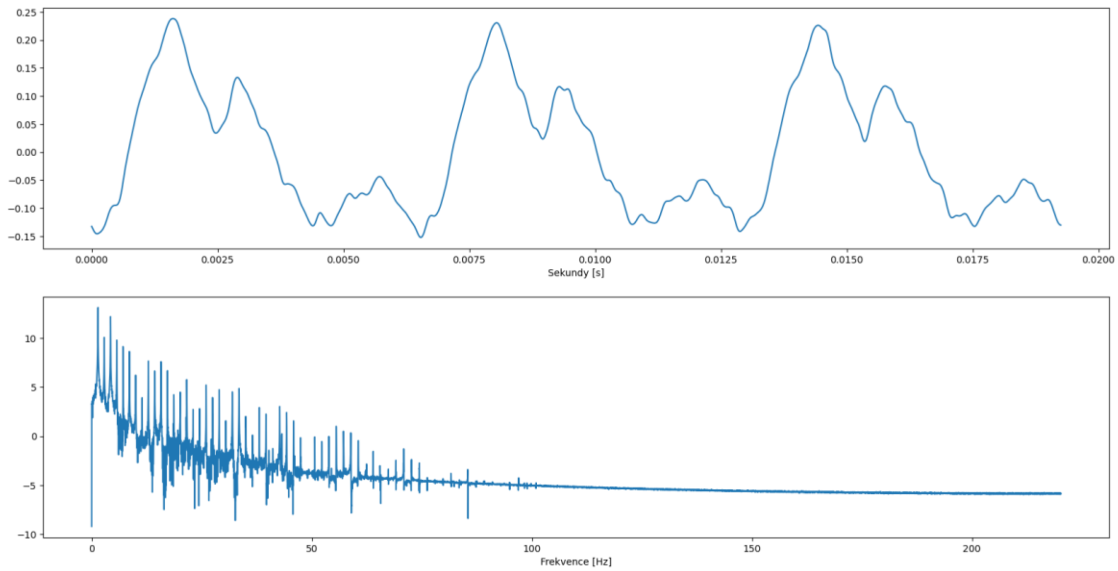
```
Text(0.5, 0, 'Frekvence [Hz]')
```



Adam Kala xkalaa00

```
#ukol 4.1.2
_, tony = plt.subplots(2,1,figsize=(20, 10))
tony[0].plot(second[:925],start2[:perioda2])
tony[0].set_xlabel("Sekundy [s]")
FFT2 = np.fft.fft(start2)
log2 = np.log(10e-5+np.abs(FFT2)**2)
moduleHalf2 = log2[:log2.size // 2]
F2 = np.arange(moduleHalf2.size)*(Fs /xall.size)
sf.write("../b_orig.wav", xall[51], Fs)
tony[1].plot(F2, moduleHalf2)
tony[1].set_xlabel("Frekvence [Hz]")
```

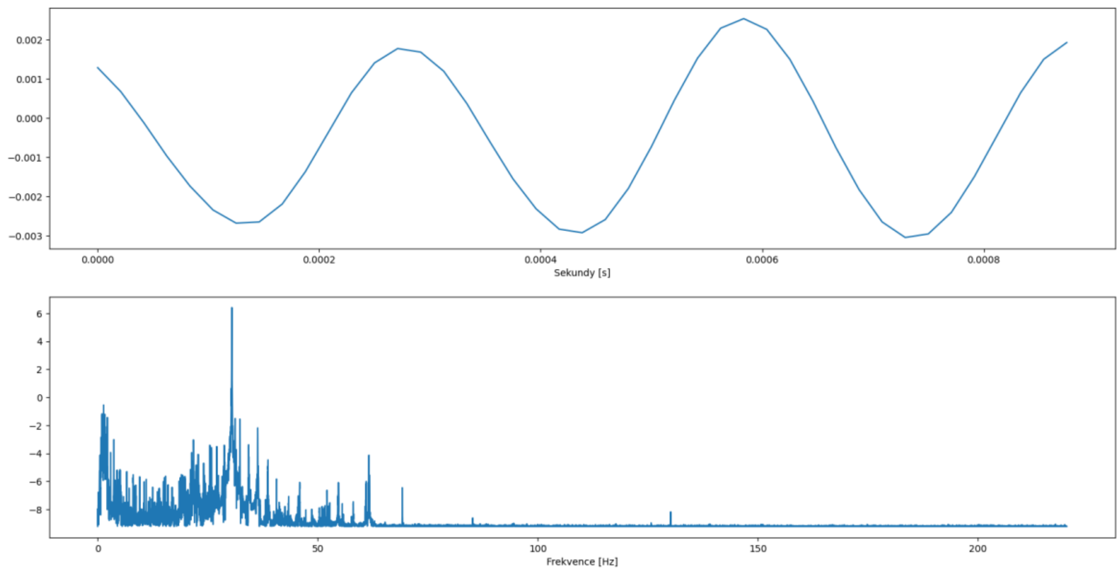
```
Text(0.5, 0, 'Frekvence [Hz]')
```



Adam Kala xkalaa00

```
#ukol 4.1.3
_, tony = plt.subplots(2,1,figsize=(20, 10))
tony[0].plot(second[:43],start3[:perioda3])
tony[0].set_xlabel("Sekundy [s]")
FFT3 = np.fft.fft(start3)
log3 = np.log(10e-5+np.abs(FFT3)**2)
moduleHalf3 = log3[:log3.size // 2]
F3 = np.arange(moduleHalf3.size)*(Fs /xall.size)
sf.write("../c_orig.wav", xall[104], Fs)
tony[1].plot(F3, moduleHalf3)
tony[1].set_xlabel("Frekvence [Hz]")
```

```
Text(0.5, 0, 'Frekvence [Hz]')
```



Adam Kala xkalaa00

```
#ukol 4.2. zakladni frekvence vsech tonu
```

Pro určení základní frekvence jsem si vybral metodu autokorelace, která je sice horší pro tóny s vyšší frekvencí, ale lépe zvládne ty nízké. Pokud nějaký tón neseděl, vynásobil jsem jeho hodnotu podle potřeby. Podle porovnání s MIDI frekvencemi jsem zjistil, že se hodnoty s menší odchylkou nemění, čili jsou správné. Hodnotu jsem vypočítal podle nalezení první maximální hodnoty (kromě té na x=0), jeji hodnotu na x ose jsem dal do vzorečku (1/hodnota)*Fs.

```
for i in np.arange(24,109):
    vypocetCorrelate = (ss.correlate(xall[i],xall[i],"full"))
    velikost = int(vypocetCorrelate.size//2)
    korelace = vypocetCorrelate[velikost:]
    vrcholky = ss.find_peaks(korelace, height=0.0001)
    maxVrch = vrcholky[1]['peak_heights']
    hodnota = np.argmax(maxVrch)
    odpoved = vrcholky[0][hodnota]
    pomocna = (1/odpoved)*Fs
    print(i, pomocna)
```

```
24 32.80929596719071
25 34.757422157856624
26 36.83806600153492
27 39.02439024390244
28 41.343669250645995
29 43.7956204379562
30 46.42166344294004
31 49.18032786885246
32 52.11726384364821
33 55.172413793103445
34 58.465286236297196
35 61.935483870967744
36 65.57377049180327
37 69.46454413892909
38 73.61963190184049
39 77.92207792207793
40 82.61617900172118
41 87.75137111517367
42 92.84332688588007
43 98.36065573770492
44 104.34782608695652
45 110.59907834101382
46 117.07317073170732
47 123.71134020618557
48 131.14754098360655
49 138.72832369942196
50 147.23926380368098
51 155.84415584415586
```

Adam Kala xkalaa00

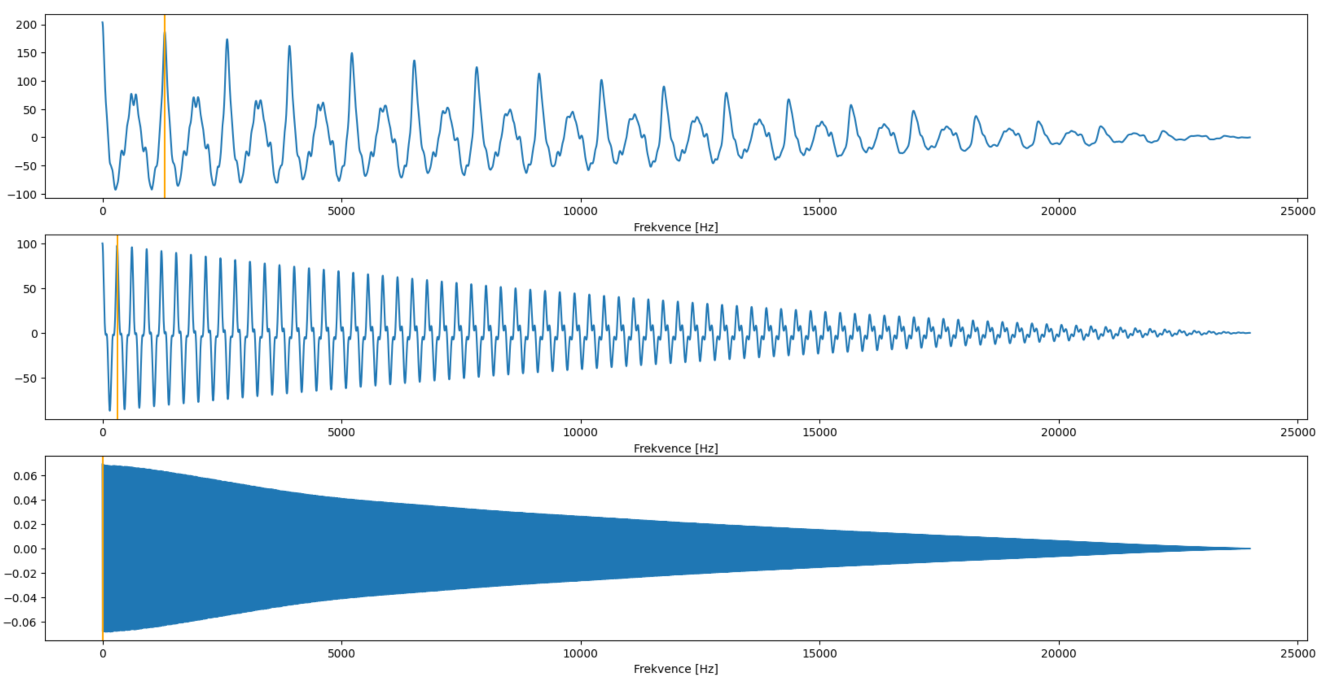
```
52 164.94845360824743
53 175.1824817518248
54 185.32818532818533
55 196.72131147540983
56 207.7922077922078
57 220.1834862385321
58 234.14634146341464
59 247.42268041237114
60 262.2950819672131
61 277.4566473988439
62 294.47852760736197
63 311.6883116883117
64 328.7671232876712
65 350.3649635036496
66 369.2307692307692
67 393.44262295081967
68 417.39130434782606
69 440.3669724770642
70 466.0194174757281
71 494.8453608247423
72 521.7391304347826
73 551.7241379310344
74 585.3658536585366
75 623.3766233766235
76 657.5342465753424
77 695.6521739130435
78 738.4615384615385
79 786.8852459016393
80 827.5862068965517
81 888.8888888888888
82 941.1764705882352
83 1000.0
84 1043.4782608695652
85 1116.2790697674418
86 1170.7317073170732
87 1230.7692307692307
88 1333.3333333333333
89 1411.764705882353
90 1500.0
91 1548.3870967741934
92 1655.1724137931035
93 1777.7777777777776
94 1846.1538461538462
95 2000.0
96 2086.9565217391305
97 2181.818181818182
98 1170.7317073170732 #tuto hodnotu vynasobime 2, aby se rovnala pozadovane
99 2526.315789473684
100 2666.6666666666665
101 2823.529411764706
102 3000.0
103 1043.4782608695652 #tuto hodnotu vynasobime 3, aby se rovnala pozadovane
104 1655.1724137931035 #tuto hodnotu vynasobime 2, aby se rovnala pozadovane
105 1170.7317073170732 #tuto hodnotu vynasobime 3, aby se rovnala pozadovane
```

Adam Kala xkalaa00

```
106 3692.3076923076924
107 4000.0
108 2086.9565217391305 #tuto hodnotu vynasobime 2, aby se rovnala pozadovane

#hodnoty jsou spatne z duvodu autokorelace, ktera je horsi pro vetsi hodnoty
```

```
#ukol 4.2 zobrazeni grafu
_, tony = plt.subplots(3,1,figsize=(20, 10))
correlate0 = ss.correlate(xall[26],xall[26],"full")
correlate = correlate0[int(correlate0.size//2):]
tony[0].plot(correlate)
tony[0].set_xlabel("Frekvence [Hz]")
tony[0].axvline(x=1303, ls='-', color="orange")

correlate02 = ss.correlate(xall[51],xall[51],"full")
correlate2 = correlate02[int(correlate02.size//2):]
tony[1].plot(correlate2)
tony[1].set_xlabel("Frekvence [Hz]")
tony[1].axvline(x=308, ls='-', color="orange")

correlate03 = ss.correlate(xall[104],xall[104],"full")
correlate3 = correlate03[int(correlate03.size//2):]
tony[2].plot(correlate3)
tony[2].set_xlabel("Frekvence [Hz]")
tony[2].axvline(x=14, ls='-', color="orange")
```

```
<matplotlib.lines.Line2D at 0x7f3dfa07a3e0>
```



Adam Kala xkalaa00

#ukol 4.3 zpresneni odhadu

V úkolu 3 jsem si opět vybral metodu autokorelace spojenou s DTFT metodou. V první polovině kódu dělám to stejné, co v předchozím cvičení. V druhé polovině přijde na scénu DTFT. Je to kód z python notebooků z materiálů na stránkách ISS s mírnou změnou ve for cyklu, kde jsem místo -1j zadal -2j, který mi poskytoval lepší a přesnější výsledky. U této metody mi pro změnu vycházela špatně frekvence s vyšší hodnotou, které jsem musel opět podle potřeby vynásobit.

```
# >>>>>>>>>>>>>>>>> #dtft <<<<<<<<<<<<<<<<<<<<<<<<<
for i in np.arange(24,109):
    vypocetCorrelate = (ss.correlate(xall[i],xall[i],"full"))
    velikost = int(vypocetCorrelate.size//2)
    korelace = vypocetCorrelate[velikost:]

    vrcholky = ss.find_peaks(korelace, height=0.0001)
    maxVrch = vrcholky[1]['peak_heights']
    hodnota = np.argmax(maxVrch)

    odpoved = vrcholky[0][hodnota]
    pomocna = ((1/odpoved)*Fs)

    FREQRANGE = 10
    FREQPOINTS = 500
    n = np.arange(0, N)

    # finding the max and showing where we'll compute ...
    fsweep = np.linspace(pomocna-FREQRANGE, pomocna+FREQRANGE,FREQPOINTS)

    # do the DTFT
    A = np.zeros([FREQPOINTS, N],dtype=complex)
    for k in np.arange(0,FREQPOINTS):
        A[k,:] = np.exp(-2j * 2 * np.pi * fsweep[k] / Fs * n)  # norm. omega = 2 * pi * f
/ Fs ...
    Xdtft = np.matmul(A,xall[i].T)
    precisefmax = fsweep[np.argmax(np.abs(Xdtft))]
    #i je ton, pomocna je vypocitane pomoci korelace, precisefmax je pomoci korelace a
pote dtft
    print(i, pomocna, precisefmax)
```

```
24 32.80929596719071 32.66901540606846
25 34.757422157856624 34.61714159673438
26 36.83806600153492 36.69778544041267
27 39.02439024390244 38.88410968278019
28 41.343669250645995 41.16330852920311
29 43.7956204379562 43.615259716513314
30 46.42166344294004 46.24130272149715
31 49.18032786885246 48.95988698708893
32 52.11726384364821 51.85674280156404
33 55.172413793103445 54.951972911339915
34 58.465286236297196 58.24484535453367
```

Adam Kala xkalaa00

```
35 61.935483870967744 61.63488266856294
36 65.57377049180327 65.31324944971911
37 69.46454413892909 69.20402309684492
38 73.61963190184049 73.43927118039761
39 77.92207792207793 77.82187752127632
40 82.61617900172118 82.4358182802783
41 87.75137111517367 87.33052943180694
42 92.84332688588007 92.54272568347527
43 98.36065573770492 98.01997437497947
44 104.34782608695652 103.88690424326914
45 110.59907834101382 110.01791601636452
46 117.07317073170732 116.5721687276993
47 123.71134020618557 123.17025804185691
48 131.14754098360655 130.44613817799532
49 138.72832369942196 138.1872415350933
50 147.23926380368098 147.09898324255875
51 155.84415584415586 155.90427608463682
52 164.94845360824743 165.16889449001096
53 175.1824817518248 174.76164006845806
54 185.32818532818533 185.14782460674243
55 196.72131147540983 196.18022931108118
56 207.7922077922078 208.13288915493325
57 220.1834862385321 220.48408744093692
58 234.14634146341464 233.605259299086
59 247.42268041237114 247.48280065285212
60 262.2950819672131 262.1548014060908
61 277.4566473988439 277.75724860124876
62 294.47852760736197 293.41640335886495
63 311.6883116883117 310.82658824141794
64 328.7671232876712 329.30820545199987
65 350.3649635036496 349.262759094832
66 369.2307692307692 370.05241251734236
67 393.44262295081967 392.0197772594369
68 417.39130434782606 415.5275768929163
69 440.3669724770642 440.18661175562136
70 466.0194174757281 466.36009883845355
71 494.8453608247423 494.14395801913105
72 521.7391304347826 523.4826174087306
73 551.7241379310344 554.6299495542809
74 585.3658536585366 587.4700620753703
75 623.3766233766235 622.3144991281265
76 657.5342465753424 659.2376533889696
77 695.6521739130435 699.2393482617409
78 738.4615384615385 740.806227840296
79 786.8852459016393 784.861197805447
80 827.5862068965517 831.7345034897381
81 888.8888888888888 881.2536183478067
82 941.1764705882352 933.7416008487563
83 1000.0 990.2004008016032
84 1043.4782608695652 1049.0694432342948
85 1116.2790697674418 1111.4894906091251
86 1170.7317073170732 1177.6054548120633
87 1230.7692307692307 1240.2481886850624
88 1333.3333333333333 1324.1349365397461
```

Adam Kala xkalaa00

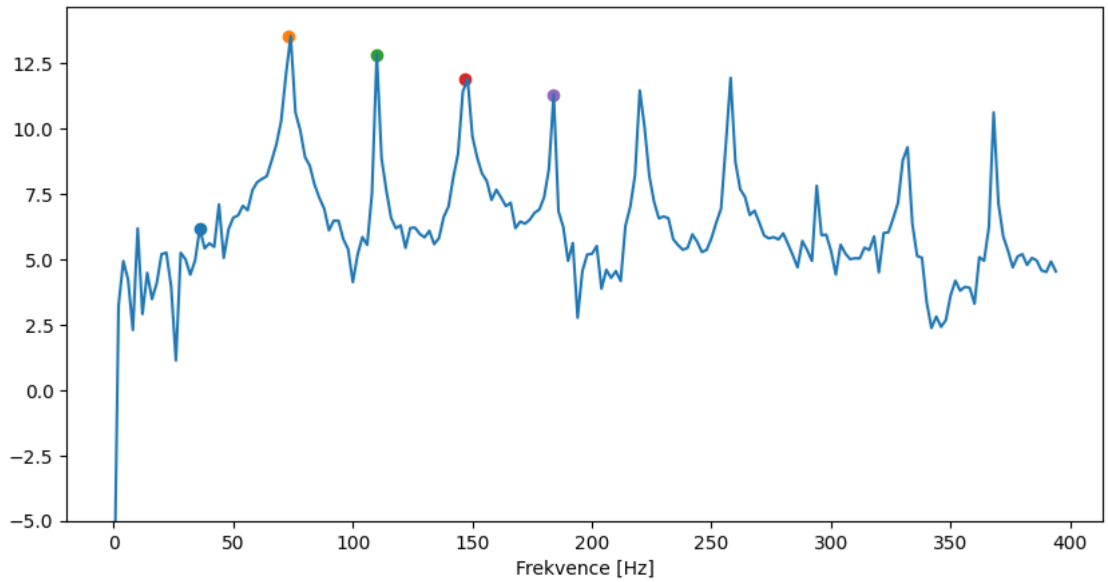
89 1411.764705882353 1402.20558764588
90 1500.0 1490.801603206413
91 1548.3870967741934 1556.5033292391233
92 1655.1724137931035 1665.1724137931035
93 1777.7777777777776 1768.499220663549
94 1846.1538461538462 1854.0696778171728
95 2000.0 1992.004008016032
96 2086.9565217391305 2095.794197089832
97 2181.818181818182 2174.1829112770997
98 1170.7317073170732 1175.3208856737865 #tuto hodnotu vynasobime 2, aby se rovnala pozadovane
99 2526.315789473684 2517.3177934817004
100 2666.6666666666665 2659.311957247829
101 2823.529411764706 2823.3891312035835
102 3000.0 2991.202404809619
103 1043.4782608695652 1033.9993029537336 #tuto hodnotu vynasobime 3, aby se rovnala pozadovane
104 1655.1724137931035 1662.0060811277729 #tuto hodnotu vynasobime 2, aby se rovnala pozadovane
105 1170.7317073170732 1178.2467373771933 #tuto hodnotu vynasobime 3, aby se rovnala pozadovane
106 3692.3076923076924 3699.983043009095
107 4000.0 3993.0060120240482
108 2086.9565217391305 2094.110830356365 #tuto hodnotu vynasobime 2, aby se rovnala pozadovane

Adam Kala xkalaa00

```
#ukol 4.4 pro 26
# >>>>>>>>>>>>>>>>> #dtft <<<<<<<<<<<<<<<<<<<<<<<<<
X = np.fft.fft(xall[26])
FREQRANGE = 10
FREQPOINTS = 200
kall = np.arange(0,int(N/2) +1)
Xmag = np.abs(X[kall])
Xphase = np.angle(X[kall])
f = kall / N * Fs

# finding the max and showing where we'll compute ...
fmax = f[np.argmax(Xmag)]
Xmax = np.max(Xmag)
ffrom = fmax-FREQRANGE
fto = fmax+FREQRANGE
fsweep = np.linspace(fmax-FREQRANGE, fmax+FREQRANGE,FREQPOINTS)

modul1 = np.abs(X[int(precisefmax)])
faze1 = np.angle(X[int(precisefmax)])

# do the DTFT
A = np.zeros([FREQPOINTS, N],dtype=complex)
for k in np.arange(0,FREQPOINTS):
    A[k,:] = np.exp(-1j * 2 * np.pi * fsweep[k] / Fs * n)     # norm. omega = 2 * pi * f
/ Fs ...
Xdtft = np.matmul(A,xall[26].T)
precisefmax = fsweep[np.argmax(np.abs(Xdtft))]
precisefmax = precisefmax//2 #vydelime dvema, protoze dft selhava na nizsich frekvenci
(je dvojnasobna)
prechodna = fmax #ulozime pocatecni f0 do prechodna, aby se nam s ni lepe pracovalo

preciseXmag = np.zeros(Xmag.size)

for i in np.arange(0, int(11*precisefmax//2)):
  preciseXmag[i] = np.log(10e-5+Xmag[i]**2) #vypocet logaritmu pro 11*f0

plt.figure(figsize=(10,5))
plt.plot(f[:int(11*precisefmax//2)],preciseXmag[:int(11*precisefmax//2)])
plt.ylim(ymin = -5) #aby graf vypadal lepe

for i in np.arange(1,6): #for cyklus od 1 do 5, kde nasobime hodnotu f0 a provadime DTFT
    fmax = prechodna * i

    ffrom = fmax-FREQRANGE
    fto = fmax+FREQRANGE
    fsweep = np.linspace(fmax-FREQRANGE, fmax+FREQRANGE,FREQPOINTS)

    # do the DTFT
    A = np.zeros([FREQPOINTS, N],dtype=complex)
    for k in np.arange(0,FREQPOINTS):
      A[k,:] = np.exp(-1j * 2 * np.pi * fsweep[k] / Fs * n)
    Xdtft = np.matmul(A,xall[26].T)
    precisefmax = fsweep[np.argmax(np.abs(Xdtft))]
```

Adam Kala xkalaa00

```
precisefmax = precisefmax//2

modul1 = np.abs(X[int(precisefmax)])
faze1 = np.angle(X[int(precisefmax)])
ymax = np.max(preciseXmag[int((precisefmax/2) - 2) : int((precisefmax/2) + 2)])
#-2 a +2 magicka konstanta
plt.xlabel("Frekvence [Hz]")
plt.scatter(precisefmax,ymax) #pro vykresleni “tecek”
```

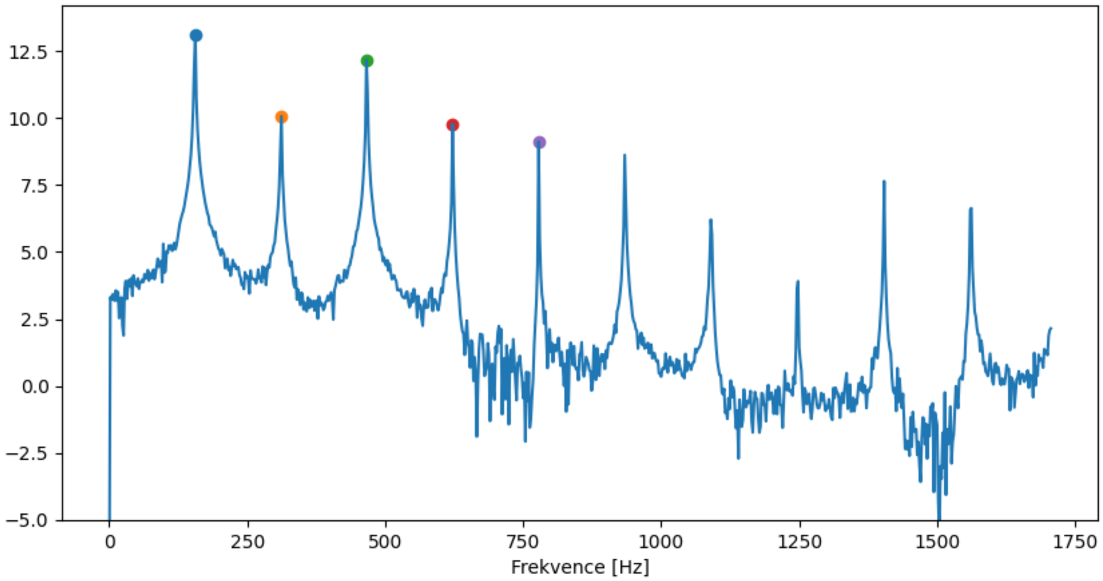


Adam Kala xkalaa00

```
#ukol 4.4 pro 51
# >>>>>>>>>>>>>>>>> #dtft <<<<<<<<<<<<<<<<<<<<<<<<<
X = np.fft.fft(xall[51])
FREQRANGE = 10
FREQPOINTS = 200
kall = np.arange(0,int(N/2) +1)
Xmag = np.abs(X[kall])
Xphase = np.angle(X[kall])
f = kall / N * Fs

# finding the max and showing where we'll compute ...
fmax = f[np.argmax(Xmag)]
Xmax = np.max(Xmag)
ffrom = fmax-FREQRANGE
fto = fmax+FREQRANGE
fsweep = np.linspace(fmax-FREQRANGE, fmax+FREQRANGE,FREQPOINTS)

modul2 = np.abs(X[int(precisefmax)])
faze2 = np.angle(X[int(precisefmax)])

A = np.zeros([FREQPOINTS, N],dtype=complex)
for k in np.arange(0,FREQPOINTS):
    A[k,:] = np.exp(-1j * 2 * np.pi * fsweep[k] / Fs * n)
Xdtft = np.matmul(A,xall[51].T)
precisefmax = fsweep[np.argmax(np.abs(Xdtft))]

prechodna = fmax

preciseXmag = np.zeros(Xmag.size)

for i in np.arange(0, int(11*precisefmax//2)):
  preciseXmag[i] = np.log(10e-5+Xmag[i]**2)

plt.figure(figsize=(10,5))
plt.plot(f[:int(11*precisefmax//2)], preciseXmag[:int(11*precisefmax//2)])
plt.ylim(ymin = -5)

for i in np.arange(1,6):
    fmax = prechodna * i
    rotate = 0

    ffrom = fmax-FREQRANGE
    fto = fmax+FREQRANGE
    fsweep = np.linspace(fmax-FREQRANGE, fmax+FREQRANGE,FREQPOINTS)

    # do the DTFT
    A = np.zeros([FREQPOINTS, N],dtype=complex)
    for k in np.arange(0,FREQPOINTS):
      A[k,:] = np.exp(-1j * 2 * np.pi * fsweep[k] / Fs * n)
    Xdtft = np.matmul(A,xall[51].T)
    precisefmax = fsweep[np.argmax(np.abs(Xdtft))]

    modul2 = np.abs(X[int(precisefmax)])
```

Adam Kala xkalaa00

```
faze2 = np.angle(X[int(precisefmax)])
ymax = np.max(preciseXmag[int((precisefmax/2) - 2) : int((precisefmax/2) + 2)])
plt.xlabel("Frekvence [Hz]")
plt.scatter(precisefmax,ymax)
```

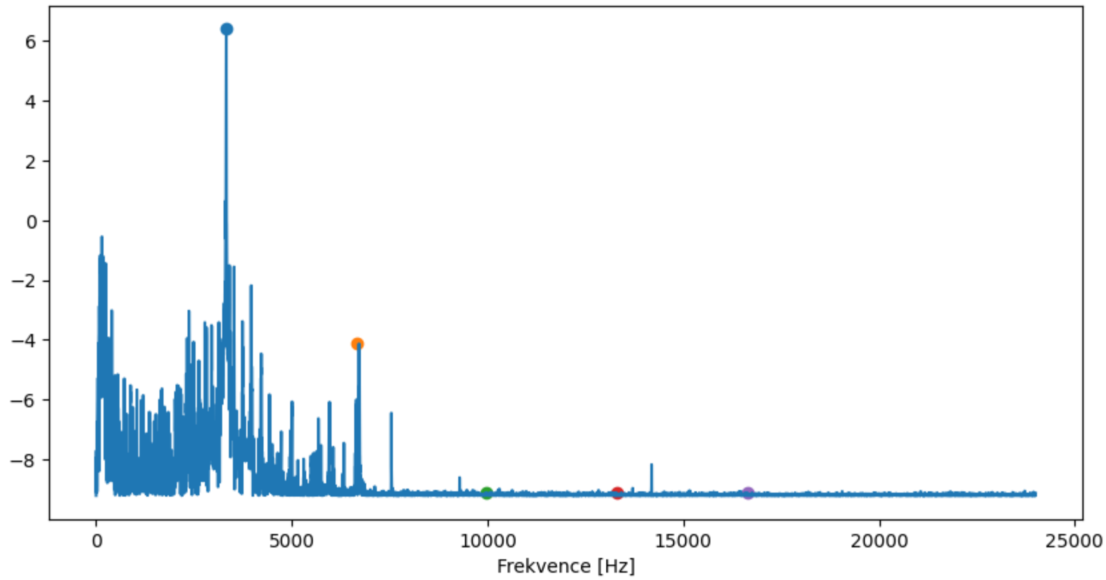


Adam Kala xkalaa00

```python
#ukol 4.4 pro 104
# >>>>>>>>>>>>>>>>> #dtft <<<<<<<<<<<<<<<<<<<<<<<<<
X = np.fft.fft(xall[104])
FREQRANGE = 10
FREQPOINTS = 200
kall = np.arange(0,int(N/2) +1)
Xmag = np.abs(X[kall])
Xphase = np.angle(X[kall])
f = kall / N * Fs

# finding the max and showing where we'll compute ...
fmax = f[np.argmax(Xmag)]
Xmax = np.max(Xmag)
ffrom = fmax-FREQRANGE
fto = fmax+FREQRANGE
fsweep = np.linspace(fmax-FREQRANGE, fmax+FREQRANGE,FREQPOINTS)

modul3 = np.abs(X[int(precisefmax)])
faze3 = np.angle(X[int(precisefmax)])

# do the DTFT
A = np.zeros([FREQPOINTS, N],dtype=complex)
for k in np.arange(0,FREQPOINTS):
    A[k,:] = np.exp(-1j * 2 * np.pi * fsweep[k] / Fs * n)
Xdtft = np.matmul(A,xall[104].T)
precisefmax = fsweep[np.argmax(np.abs(Xdtft))]

prechodna = fmax

preciseXmag = np.zeros(Xmag.size)

for i in np.arange(0, int(11*precisefmax//2)):
  if i > 12000:
    i = 12000 - i
  preciseXmag[i] = np.log(10e-5+Xmag[i]**2)

plt.figure(figsize=(10,5))
plt.plot(f[:int(11*precisefmax//2)], preciseXmag[:int(11*precisefmax//2)])

for i in np.arange(1,6):
    fmax = prechodna * i
    rotate = 0

    ffrom = fmax-FREQRANGE
    fto = fmax+FREQRANGE

    fsweep = np.linspace(fmax-FREQRANGE, fmax+FREQRANGE,FREQPOINTS)

    # do the DTFT
    A = np.zeros([FREQPOINTS, N],dtype=complex)
    for k in np.arange(0,FREQPOINTS):
      A[k,:] = np.exp(-1j * 2 * np.pi * fsweep[k] / Fs * n)
    Xdtft = np.matmul(A,xall[104].T)
```

Adam Kala xkalaa00

```
precisefmax = fsweep[np.argmax(np.abs(Xdtft))]

modul3 = np.abs(X[int(precisefmax)])
faze3 = np.angle(X[int(precisefmax)])

ymax = np.max(preciseXmag[int((precisefmax / 2) - 50) : int((precisefmax / 2) + 50)])
#-50 a +50 magicka konstanta
plt.xlabel("Frekvence [Hz]")
plt.scatter(precisefmax,ymax)
```



Adam Kala xkalaa00

```
#ukol 4.5 pro 26
```

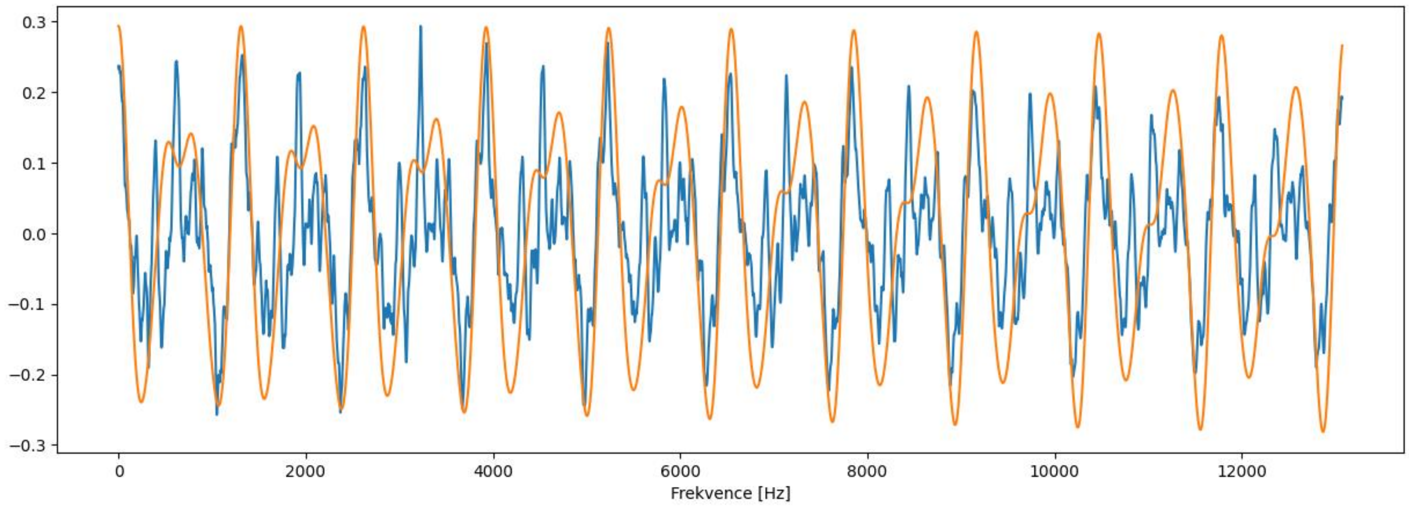
Pro tento ukol jsem si vzal DFT hodnotu, kterou jsem opět projížděl DTFT pro 5xf0. Hodnoty jsem ukládal do pole, který jsem potom celý syntetizoval. Ton jsem pote normalizoval, vykreslil a uložil.

```
# >>>>>>>>>>>>>>>>> #dtft <<<<<<<<<<<<<<<<<<<<<<<<<
X = np.fft.fft(xall[26])
FREQRANGE = 10
FREQPOINTS = 200
kall = np.arange(0,int(N/2) +1)
Xmag = np.abs(X[kall])
Xphase = np.angle(X[kall])
f = kall / N * Fs

# finding the max and showing where we'll compute ...
fmax = f[np.argmax(Xmag)]
Xmax = np.max(Xmag)
ffrom = fmax-FREQRANGE
fto = fmax+FREQRANGE
fsweep = np.linspace(fmax-FREQRANGE, fmax+FREQRANGE,FREQPOINTS)

prechodna = fmax
ton26 = np.zeros(N)

for i in np.arange(1,6):
    fmax = prechodna * i

    ffrom = fmax-FREQRANGE
    fto = fmax+FREQRANGE
    fsweep = np.linspace(fmax-FREQRANGE, fmax+FREQRANGE,FREQPOINTS)

    # do the DTFT
    A = np.zeros([FREQPOINTS, N],dtype=complex)
    for k in np.arange(0,FREQPOINTS):
      A[k,:] = np.exp(-1j * 2 * np.pi * fsweep[k] / Fs * n)
    Xdtft = np.matmul(A,xall[26].T)

    precisefmax = fsweep[np.argmax(np.abs(Xdtft))]
    precisefmax = precisefmax//2
    index = int(precisefmax)
    ton26[index] = Xdtft[index] #ukladame hodnotu z DTFT do ton26, ktery pak
syntetizujeme

perioda1_2 = int(0.02724053*Fs*10) #10 period

ton26_0 = np.fft.ifft(ton26, n=Fs) #pouzijeme ifft pro syntetizaci
ton26_0 = np.real(ton26_0/max(ton26_0)*max(xall[26])) #ton, ktery vysel z ifft, vydelim
jeho nejvetsi hodnotou a vynasobim amplitudou z xall[26]
sf.write("../a.wav", ton26_0, Fs) #ulozime ton
plt.figure(figsize=(15,5))
plt.plot(xall[26][:perioda1_2])
plt.plot(ton26_0[N:perioda1_2+N]) #pro hezci graf pridame konstantu
plt.xlabel("Frekvence [Hz]")
```

Adam Kala xkalaa00

```
/tmp/ipykernel_2823/750745762.py:39: ComplexWarning: Casting complex values to real
discards the imaginary part
  ton26[index] = Xdtft[index]
```

```
Text(0.5, 0, 'Frekvence [Hz]')
```



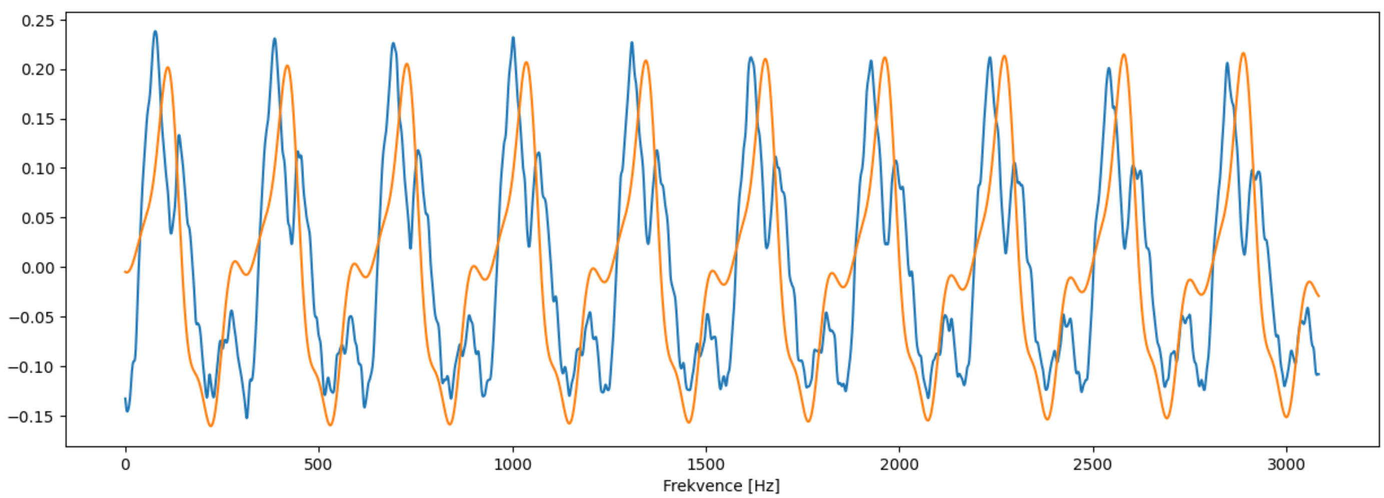

```
#ukol 4.5 pro 51
# >>>>>>>>>>>>>>>>> #dtft <<<<<<<<<<<<<<<<<<<<<<<<<
X = np.fft.fft(xall[51])
FREQRANGE = 10
FREQPOINTS = 1000
kall = np.arange(0,int(N/2) +1)
Xmag = np.abs(X[kall])
Xphase = np.angle(X[kall])
f = kall / N * Fs

# finding the max and showing where we'll compute ...
fmax = f[np.argmax(Xmag)]
Xmax = np.max(Xmag)
ffrom = fmax-FREQRANGE
fto = fmax+FREQRANGE
fsweep = np.linspace(fmax-FREQRANGE, fmax+FREQRANGE,FREQPOINTS)

prechodna = fmax
ton51 = np.zeros(N)


for i in np.arange(1,6):
    fmax = prechodna * i

    ffrom = fmax-FREQRANGE
    fto = fmax+FREQRANGE
    fsweep = np.linspace(fmax-FREQRANGE, fmax+FREQRANGE,FREQPOINTS)

    # do the DTFT
    A = np.zeros([FREQPOINTS, N],dtype=complex)
    for k in np.arange(0,FREQPOINTS):
      A[k,:] = np.exp(-1j * 2 * np.pi * fsweep[k] / Fs * n)
```

Adam Kala xkalaa00

```
    Xdtft = np.matmul(A,xall[51].T)

    precisefmax = fsweep[np.argmax(np.abs(Xdtft))]
    index = int(precisefmax)
    ton51[index] = Xdtft[index]

perioda2_2 = int(0.00642839*Fs*10)

ton51_0 = np.fft.ifft(ton51, n=Fs)
ton51_0 = np.real(ton51_0/max(ton51_0)*(max(xall[51])))
sf.write("../b.wav", ton51_0, Fs)
plt.figure(figsize=(15,5))
plt.plot(xall[51][:perioda2_2])
plt.plot(ton51_0[12000:perioda2_2+12000]) #konstanta pro posunuti tonu, aby odpovidal
zadanemu tonu
plt.xlabel("Frekvence [Hz]")
```

```
/tmp/ipykernel_2823/2236565542.py:37: ComplexWarning: Casting complex values to real
discards the imaginary part
  ton51[index] = Xdtft[index]
```

```
Text(0.5, 0, 'Frekvence [Hz]')
```




```
#ukol 4.5 pro 104
# >>>>>>>>>>>>>>>>> #dtft <<<<<<<<<<<<<<<<<<<<<<<<<
X = np.fft.fft(xall[104])
FREQRANGE = 10
FREQPOINTS = 16650
```

Adam Kala xkalaa00

```
kall = np.arange(0,int(N/2) +1)
Xmag = np.abs(X[kall])
Xphase = np.angle(X[kall])
f = kall / N * Fs

# finding the max and showing where we'll compute ...
fmax = f[np.argmax(Xmag)]
Xmax = np.max(Xmag)
ffrom = fmax-FREQRANGE
fto = fmax+FREQRANGE
fsweep = np.linspace(fmax-FREQRANGE, fmax+FREQRANGE,FREQPOINTS)
n = np.arange(0, N)
prechodna = fmax
ton104 = np.zeros(N)


for i in np.arange(1,6):
    fmax = prechodna * i

    ffrom = fmax-FREQRANGE
    fto = fmax+FREQRANGE
    fsweep = np.linspace(fmax-FREQRANGE, fmax+FREQRANGE,FREQPOINTS)

    # do the DTFT
    A = np.zeros([FREQPOINTS, N],dtype=complex)
    for k in np.arange(0,FREQPOINTS):
      A[k,:] = np.exp(-1j * 2 * np.pi * fsweep[k] / Fs * n)
    Xdtft = np.matmul(A,xall[104].T)

    precisefmax = fsweep[np.argmax(np.abs(Xdtft))]
    index = int(precisefmax)
    ton104[index] = Xdtft[index]

perioda3_2 = int(0.00030098*Fs*10)

ton104_0 = np.fft.ifft(ton104, n=Fs)
ton104_0 = np.real((ton104_0)/max(ton104_0)*max(xall[104]))
sf.write("../c.wav", ton104_0, Fs)
plt.figure(figsize=(15,5))
plt.plot(xall[104][:perioda3_2])
plt.plot(ton104_0[:perioda3_2])
plt.xlabel("Frekvence [Hz]")
```

```
/tmp/ipykernel_2823/2585288865.py:37: ComplexWarning: Casting complex values to real
discards the imaginary part
  ton104[index] = Xdtft[index]

Text(0.5, 0, 'Frekvence [Hz]')
```

Adam Kala xkalaa00