Signalų filtravimas su FFT

Signalų analizė ir apdorojimas

Rokas Mačiulaitis 2019-05-31

Analizuotų signalų sąrašas

1. Sintetinis signalas
2. Minimali paros temperatūra Melburne - https://raw.githubusercontent.com/jbrownlee/Datasets/master/shampoo.csv

Algoritmų palyginimas

Realizavus algoritmus - Diskrečioji Furje transformacija (DFT) bei Greitoji Furje transformacija (FFT) buvo atliekami gautų rezultatų palyginimai norint įsitikti, jog algoritmas veikia teisingai.

Testavimui buvo naudojamas atsitiktinai sugeneruotas 1024 dydžio masyvas (1 pav.).

```
In [99]: x = np.random.random(1024)
```

1 pav. Sugeneruojamas 1024 dydžio atsitiktinis masyvas.

Realizuotų algoritmų rezultatai buvo palyginti su NumPy bei SciPy bibliotekų atitinkamų procedūrų rezultatais. Kaip matome (2 pav.) implementuotų DFT bei FFT algortimų rezultatai sutampa.

```
In [20]: # Let's compare the results of DFT function with numpy FFT
         np.allclose(DFT(x), np.fft.fft(x))

Out[20]: True

In [21]: # Let's compare the results of FFT function with sc FFT
         np.allclose(FFT(x), sc.fft(x))

Out[21]: True

In [22]: # Let's compare the results of FFT function with numpy FFT
         np.allclose(FFT(x), np.fft.fft(x))

Out[22]: True
```

2 pav. Algoritmų testavimas

DFT ir FFT greičių palyginimas

DFT yra 9 kartus (3 pav.) lėtesnis nei FFT skaičiuojant Furje transformaciją masyvui su 1024 elementais.

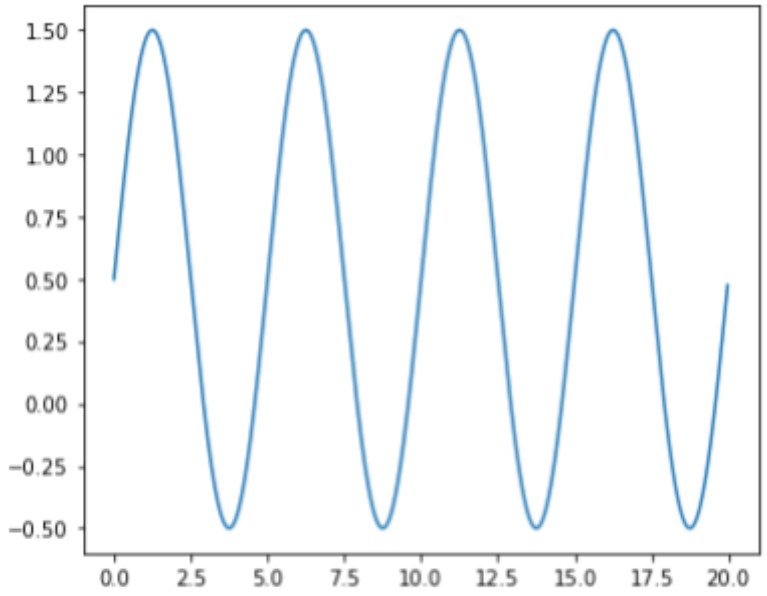
```
In [24]: # compare execution time of DFT and FFt
         %timeit DFT(x)
         %timeit FFT(x)

         45.4 ms ± 2.57 ms per loop (mean ± std. dev. of 7 runs, 10 loops each)
         5.07 ms ± 986 µs per loop (mean ± std. dev. of 7 runs, 100 loops each)
```

3 pav. DFT ir FFT greičio palyginimas

Sintetinis užtriukšmintas signalas

Pirmas analizuojamas signalas (4 pav.) yra gaunamas naudojant žemiau pateiktą kodą.

```
sin(2 * np.pi / period * time) + 0.5
```

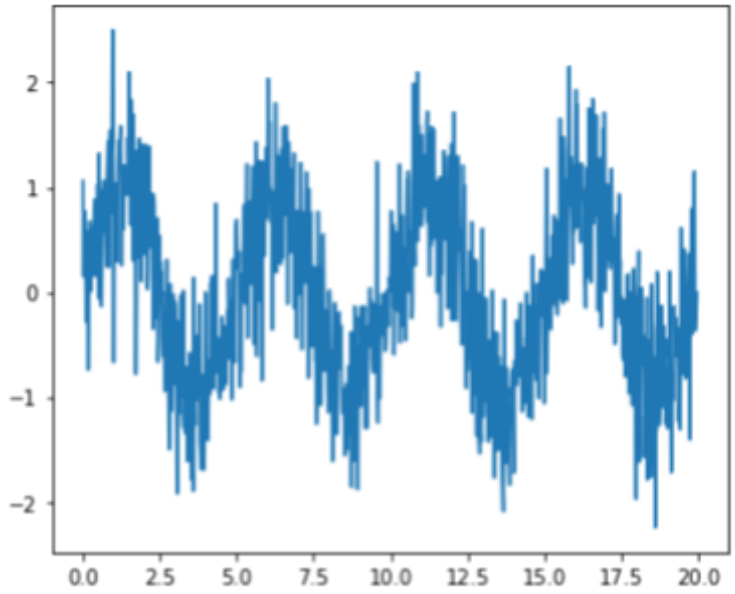


4 pav. Orginalus signalas

Orginalus signalas yra dirbtinai užtriukšminamas sudauginant signalo reikšmes su atsitiktinai gaunamais skaitmenimis intervale [0,N].

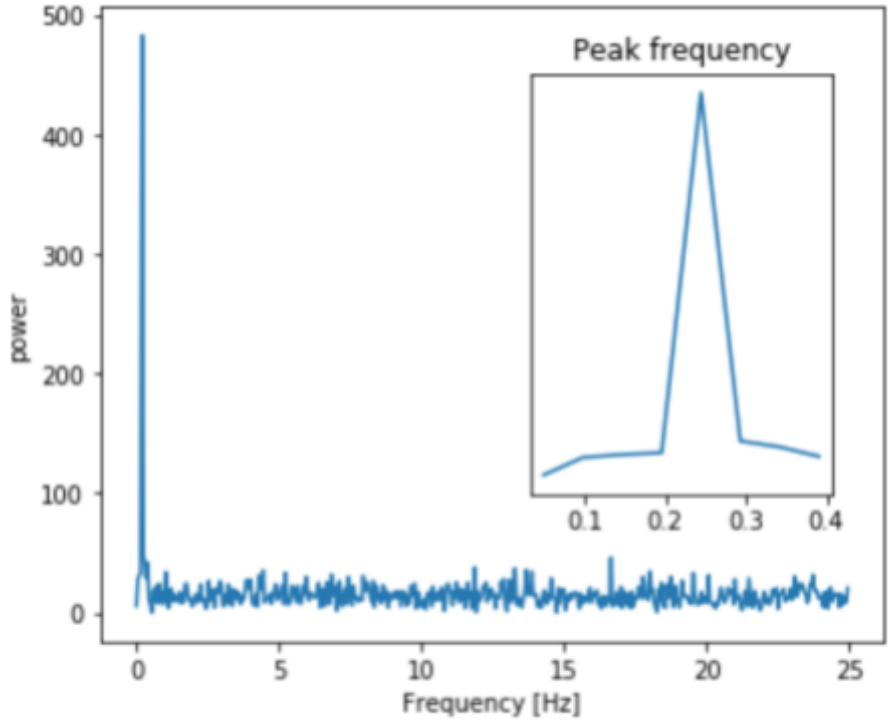
```
sin(2 * np.pi / period * time) + 0.5 * random(time.size))
```

Gaunamas užtriukšmintas signalas (5 pav.)



5 pav. Užtriukšmintas signalas

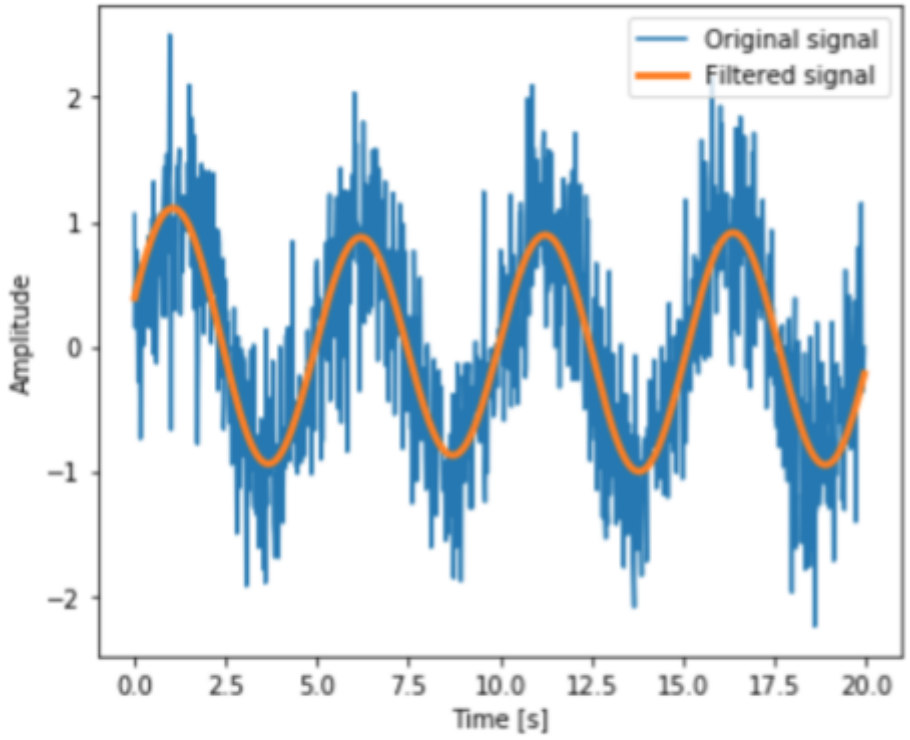
Norint pašalinti triukšmą iš signalo, reikia žinoti kokius dažnius prilyginti nuliui. Dažnių priklausomybė nuo FFT koficientų (6 pav.).



6 pav. FFT koeficientų priklausomybė nuo dažnio

Matome staigų šuolį dažnių intervale nuo 0.2 iki 0.3, tad nuliui prilyginsime FFT koeficientus, atitinkačius šių dažnių rėžį. Naujai gauti FFT koeficientai yra nusiunčiami į atvirkštinį FFT

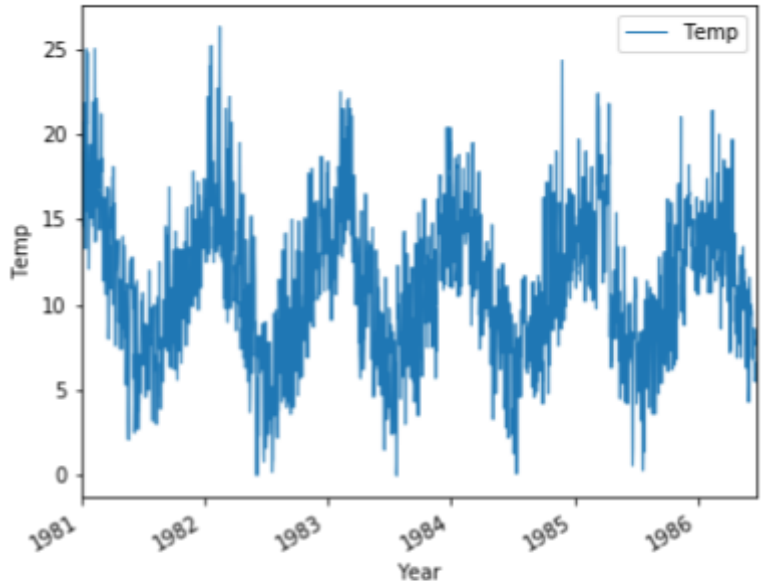
algoritmą ir gaunamas nutriukšmintas, pradinis sinalas (7 pav.).



7 pav. Orginalus pritriukšmintas signalas ir jo nutriukšminta versija

Minimali paros temperatūra Melburne

Šis signalas nusako minimalią paros temperatūrą Melburne, Australijoje. Analizuosime 2000 dienų rėžį. 8 pav. Matomi tam tikri sezoniniai temperatūros svyravimai, tačiau dėl per dažnų temperatūros matavimų negalime matyti ar egzistuoja tam tikros sezoninių kitimų tendencijos. Pvz. minimali temperatūra vasaros sezono metu kyla bėgant metams.

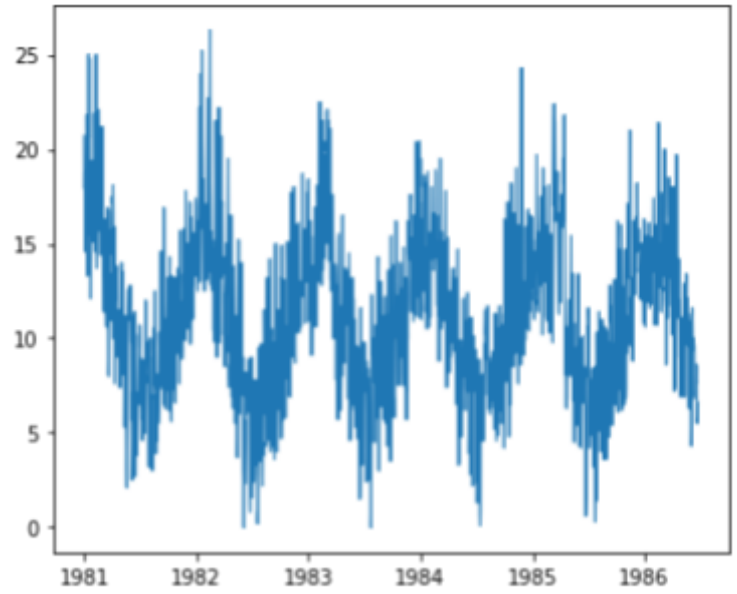


8 pav. Minimali dienos temperatūra Melburne

Norint įžvelgti anksčiau minėtas tendencijas reikėtų susilpninti šio signalo dažnį. Jeigu laiko matavimo vienetas yra metai, šis signalas turi 365 signalo reikšmes (365 Hz). Tad apskaičiavus FFT koeficientus ir filtruoti signalus nustačius maksimalu 356 Hz dažnį turėtume gauti pradinį grafiką kaip ir 8 pav.

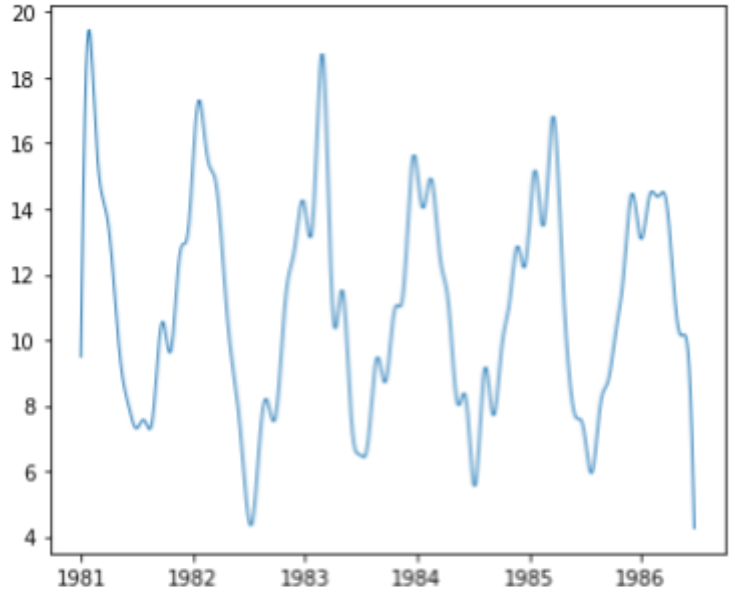
```
temp_signal_filtered = nulls_freq_in_signal(
    signal=temp_signal,
    max_freq=365,
    min_freq=None,
    timestep=1)
```

Atlikus šią operaciją gavome įdentišką grafiką 9 pav.



9 pav. Išfiltruotas signalas su 365 Hz dažniu

Atlikus tokias pat operacijas, tačiau su mažesniu maksimaliu dažniu - 1, turėtume įžvelgti anksčiau minėtas tendencijas (10 pav.).



10 pav. Išfiltruotas signalas

Tam tikrų, pasirinktų FFT koeficientų prilyginimas nuliams (aštrus filtravimas) nėra pats tiksliausias būdas norint įžvelgti anksčiau minėtas tendencijas. Kur kas tikslesni rezultatai būtų panaudojus taip vadinamą “glodųjį” filtravimą.