# fourier_transforms

July 26, 2019

## 0.1 FFT 1

**The arapaly.txt file contains data about the fluctuations of the sea level at the nuclear power plant in Haysham, UK.**

**Is there a dominant frequency? If yes, what kind of dominant frequency do we expect?**

**Plot the Fourier-transformed signal! Is the dominant frequency visible in the frequency spectrum?**

```python
[15]: # Importing required environments
      %pylab inline
      from numpy.fft import *

      ds = loadtxt("arapaly.txt")
      plot(ds[:288,0], ds[:288,1])
      xlabel("Days", size=12)
      ylabel("Height difference from average sea level [m]", size=10)
      title("Height difference from average sea level with respect to days", size=18,
        ↪y=1.05) # Title and labels added to plot
```

Populating the interactive namespace from numpy and matplotlib

```
[15]: Text(0.5, 1.05, 'Height difference from average sea level with respect to days')
```

1

# Height difference from average sea level with respect to days
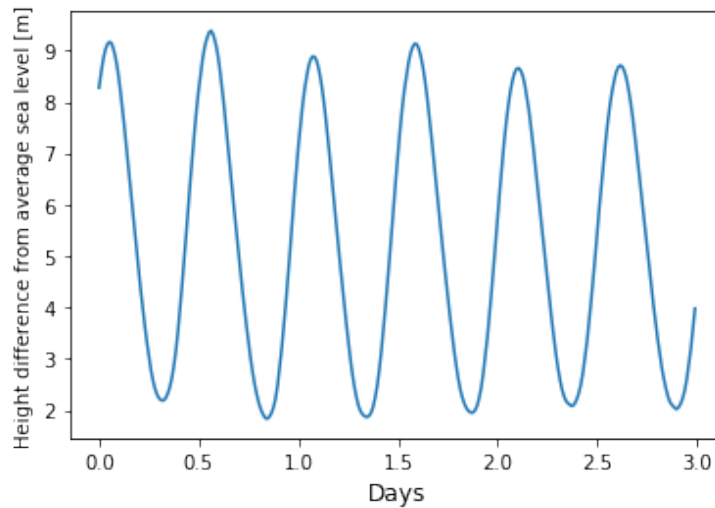


[14]:
```python
# FFT signal
Fsignal = fft(ds[:,1] - mean(ds[:, 1])) # The mean is subtracted, so the 0 Hz
 ↪ffrequency does not disturb
freq = fftfreq(len(ds[:,0]), d=ds[1,0] - ds[0,0])

print("Dominant frequency: {} 1/day".format(freq[abs(Fsignal).argmax()])) #
 ↪Maxima is extracted from transformed dataset
plot(abs(freq), abs(Fsignal), label="Transformed signal")
plot(freq[abs(Fsignal).argmax()], max(abs(Fsignal)), "ro", label="The maximum
 ↪is at 1,93 1/day")
xlabel("Frequency", size=12)
ylabel("Transformed signal strength", size=12)
title("FFTed signal", size=18, y=1.05)
legend()
```

Dominant frequency: 1.932420091324201 1/day

[14]: <matplotlib.legend.Legend at 0x7f10d3724198>

## FFTed signal
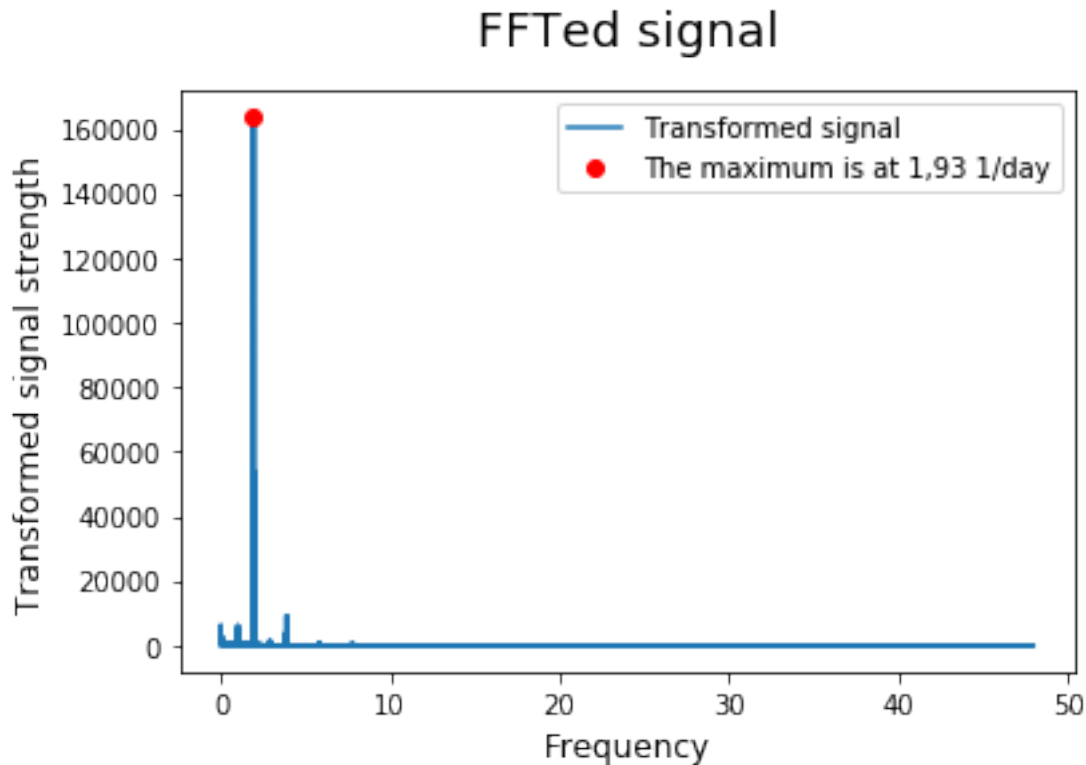


Due to the gravitational attraction of celestial bodies and the orbit of the Earth, the equipotential surfaces in the vicinity of the Earth are not perfectly spherical (minor perturbation). The large water bodies attempt to stay on these equipotential surfaces, but since the Earth is rotating an observer on Earth will see the sea level going up and down periodically, so we expect a dominant frequency.

This is the basic mechanism of high and low tide observed at different times in the day. Therefore, according to this the dominant frequency is around 12 hours (most dominant frequency spectrum).

The dominant frequency from FFT is $1,93 \frac{1}{day}$ , which corresponds to a periodicity of 12 hours and 25 minutes. The main reason for high/low tides is the rotation of the Earth, however the Moon perturbes this process and contributes to the difference from exact 12 hours. .

### 0.2 FFT 2

#### 0.2.1 Using the ifft (inverse FFT) function we can obtain the inverse FT of a function. By using this, filter the components higher than 20Hz from the audio file

```python
[9]: ds = loadtxt("zenebona.txt")
plot(ds[:,0], ds[:,1])
xlabel("Time", size=12)
ylabel("Signal strength", size=12)
title("Signal strength with respect to time", size=18, y=1.05)
```

3

[9]: Text(0.5, 1.05, 'Signal strength with respect to time')

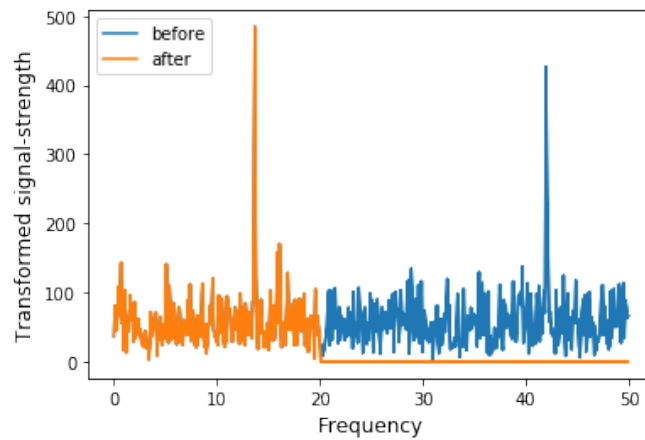## Signal strength with respect to time
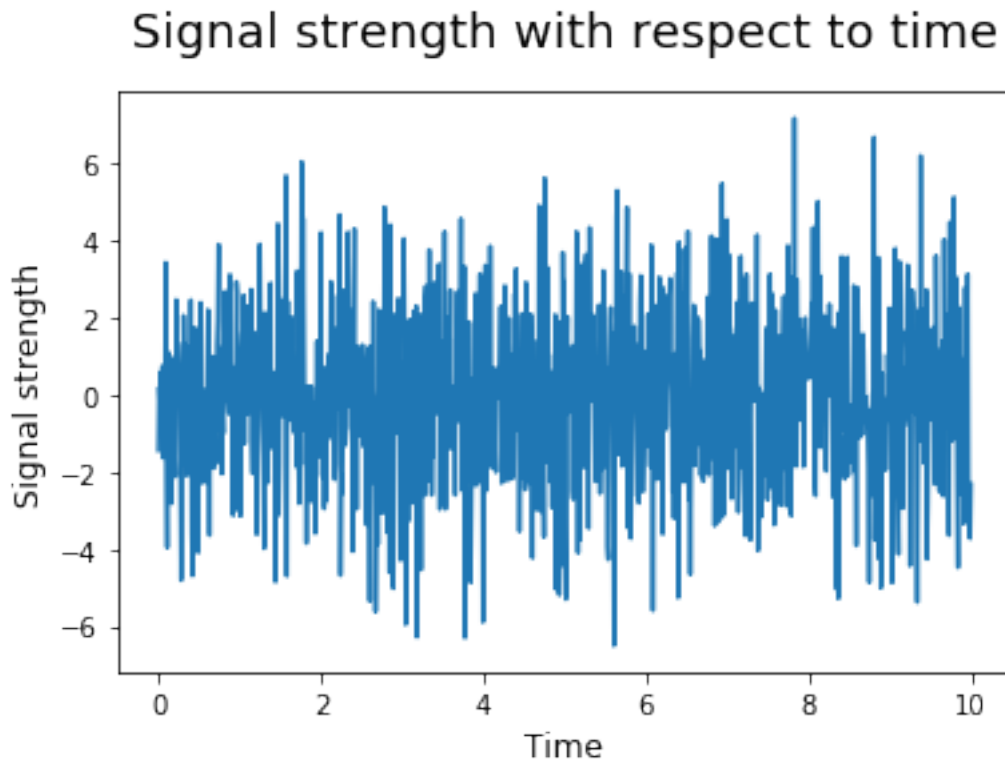


```
[11]: Fsignal = fft(ds[:,1]) # Applied FFT to dataset
      freq = fftfreq(len(ds[:,0]), d=ds[1,0] - ds[0,0])
      plot(abs(freq), abs(Fsignal), label="before")


      for i in range(len(Fsignal)):
          if abs(freq[i]) > 20:
              Fsignal[i] = 0 # Creating our virtual filter


      plot(abs(freq), abs(Fsignal), label="after")
      title("The Fourier-transformed signal before and after applying the 20Hz␣
       ↪filter", size=18, y=1.05)
      xlabel("Frequency", size=12)
      ylabel("Transformed signal-strength", size=12)
      legend()
```

[11]: <matplotlib.legend.Legend at 0x7f10d3407320>

## The Fourier-transformed signal before and after applying the 20Hz filter



```
[12]: filtered = ifft(Fsignal)
      plot(ds[:,0], filtered)
      xlabel("Time", size=12)
      ylabel("Signal Strenght", size=12)
      title("Signal strength with respect to time in filtered signal")
```

/home/alexbocc/anaconda3/lib/python3.7/site-packages/numpy/core/numeric.py:538:
ComplexWarning: Casting complex values to real discards the imaginary part
  return array(a, dtype, copy=False, order=order)

```
[12]: Text(0.5, 1.0, 'Signal strength with respect to time in filtered signal')
```

Signal strength with respect to time in filtered signal