Лабораторная работа №8
Фильтрация и свёртка

Кобыжев Александр

11 апреля 2021 г.

Оглавление

Список иллюстраций

Листинги

Глава 1

Упражнение 8.1

В данном упражнении нас просят открыть `chap08.ipynb`, прочитать пояснения, а также запустить примеры.

Если увеличивать ширину гауссова окна STD без увеличения количества элементов в окне M, это окно становится ближе к прямоугольному, более высокие частоты подавляются хуже, и следующие параметры проявляются боковым лепестком.

Глава 2

Упражнение 8.2

Начнём с гауссовского аналога:

```python
gaussian = scipy.signal.gaussian(M=64, std=2)
gaussian /= sum(gaussian)
thinkplot.plot(gaussian)
thinkplot.config(xlabel='Index')
```

Листинг 2.1: Визуализация гауссовского сигнала

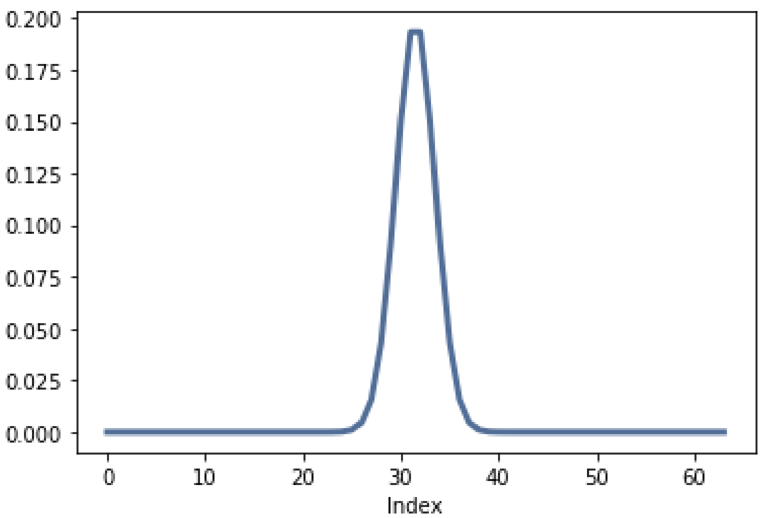

Рис. 2.1: Визуализация гауссовского сигнала

Вот как выглядит БПФ:

```python
fft_gaussian = np.fft.fft(gaussian)
thinkplot.plot(abs(fft_gaussian))
```

```python
thinkplot.config(xlabel='Frequency (Hz)', ylabel='Amplitude')
```

Листинг 2.2: Визуализация БПФ

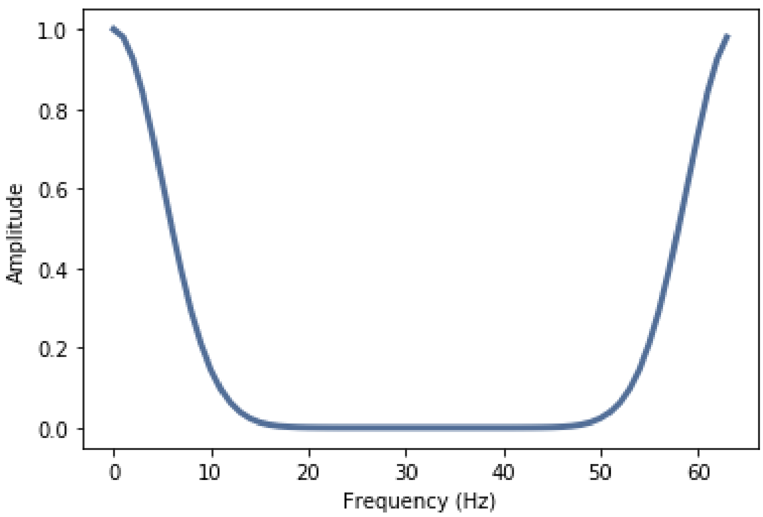

Рис. 2.2: Визуализация БПФ

Если мы повернём отрицательные частоты влево, то сможем яснее увидеть, что это гауссово, по крайней мере приблизительно.

```python
N = len(gaussian)
fft_rolled = np.roll(fft_gaussian, N//2)
thinkplot.plot(abs(fft_rolled))
thinkplot.config(xlabel='Frequency (Hz)', ylabel='Amplitude')
```

Листинг 2.3: Визуализация гауссовского сигнала

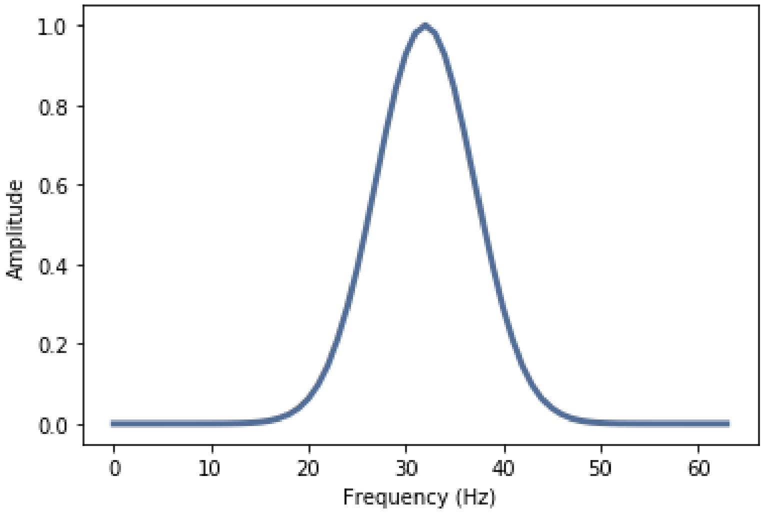

Рис. 2.3: Визуализация гауссовского сигнала

Эта функция отображает окно Гаусса и его БПФ друг с другом.

```
def plot_gaussian(std):
    M = 64
    gaussian = scipy.signal.gaussian(M=M, std=std)
    gaussian /= sum(gaussian)

    thinkplot.preplot(num=2, cols=2)
    thinkplot.plot(gaussian)
    thinkplot.config(xlabel='Time', legend=False)

    fft_gaussian = np.fft.fft(gaussian)
    fft_rolled = np.roll(fft_gaussian, M//2)

    thinkplot.subplot(2)
    thinkplot.plot(abs(fft_rolled))
    thinkplot.config(xlabel='Frequency')


plot_gaussian(2)
```

Листинг 2.4: Функция plot_gaussian

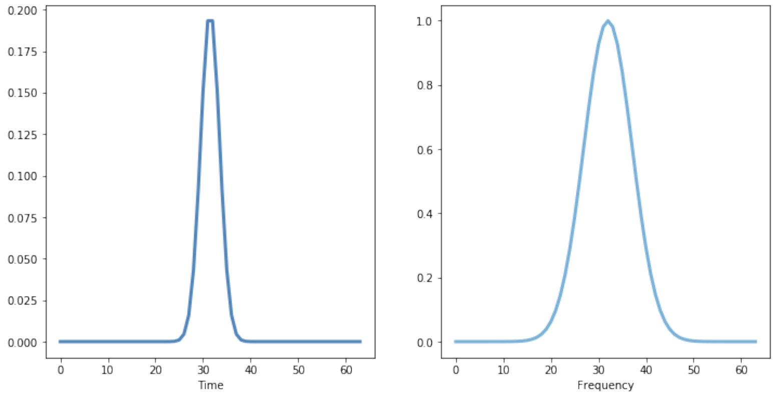

Рис. 2.4: Визуализация окна Гаусса и его БПФ

Теперь мы можем сделать взаимодействие, которое показывает, что происходит при изменении `std`.

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

slider = widgets.FloatSlider(min=0.1, max=10, value=2)
interact(plot_gaussian, std=slider);
```

Листинг 2.5: Изменение `std`

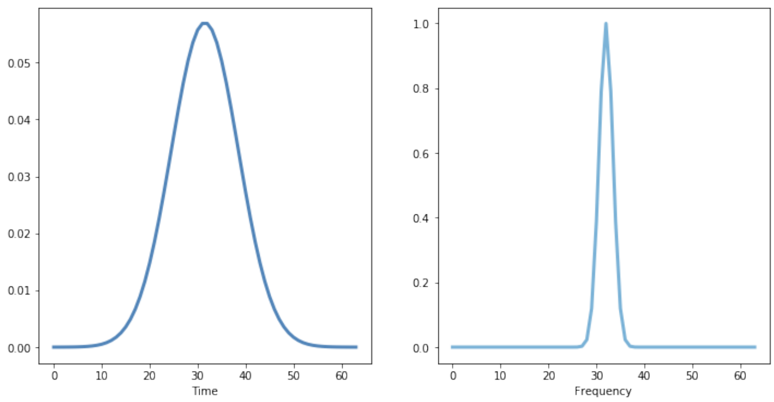

Рис. 2.5: Изменение `std`

По мере увеличения `std` Гауссовский становится шире, а его БПФ сужается.

С точки зрения непрерывной математики, если
$f(x) = e^{-ax^2}$

который является гауссовским со средним 0 и стандартным отклонением $1/a$, его преобразование Фурье имеет вид

$F(k) = \sqrt{\frac{\pi}{a}} e^{-\pi^2 k^2/a}$

который является гауссовским со стандартным отклонением a/π^2. Таким образом, существует обратная зависимость между стандартными отклонениями f и F.

Глава 3

Упражнение 8.3

Создадим 1-секундную волну с частотой дискретизации 44 кГц.

```
signal = thinkdsp.SquareSignal(freq=440)
wave = signal.make_wave(duration=1.0, framerate=44000)
```

Листинг 3.1: Создание сигнала

Затем создадим несколько окон. Выберем стандартное отклонение окна Гаусса, чтобы сделать его похожим на другие.

```
M = 17
std = 2.5

gaussian = scipy.signal.gaussian(M=M, std=std)
bartlett = np.bartlett(M)
blackman = np.blackman(M)
hamming = np.hamming(M)
hanning = np.hanning(M)

windows = [gaussian, blackman, hamming, hanning]
names = ['gaussian', 'blackman', 'hamming', 'hanning']

for window in windows:
    window /= sum(window)
```

Листинг 3.2: Создание различных окон

Теперь посмотрим, как выглядят эти окна.

```
thinkplot.preplot(4)
for window, name in zip(windows, names):
    thinkplot.plot(window, label=name)

```

```
thinkplot.config(xlabel='Index', legend=True, loc='center bottom')
```

Листинг 3.3: Визуализация окон

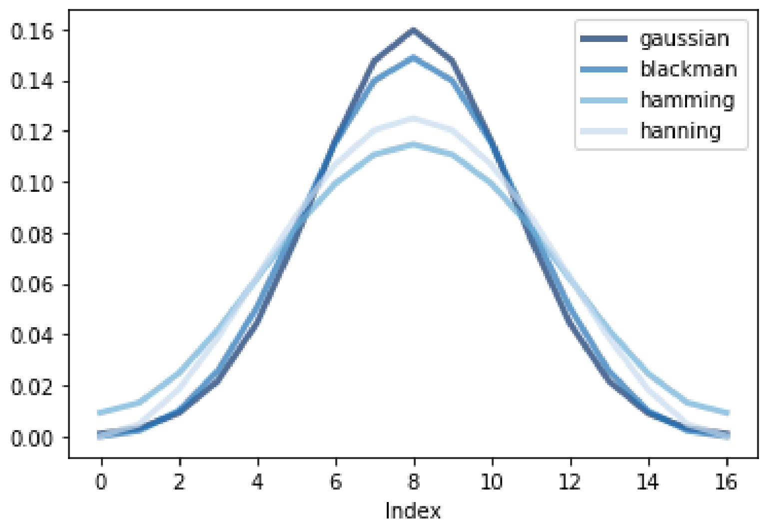

Рис. 3.1: Визуализация окон

Они выглядят довольно похоже, но по Гауссу и Блэкману немного выше. Посмотрим, как выглядят их ДПФ:

```
def plot_window_dfts(windows, names):
    thinkplot.preplot(5)

    for window, name in zip(windows, names):
        padded = thinkdsp.zero_pad(window, len(wave))
        dft_window = np.fft.rfft(padded)
        thinkplot.plot(abs(dft_window), label=name)
```

Листинг 3.4: Функция `plot_window_dfts`

```
plot_window_dfts(windows, names)
thinkplot.config(xlabel='Frequency (Hz)', loc='upper right')
```

Листинг 3.5: Визуализация ДПФ

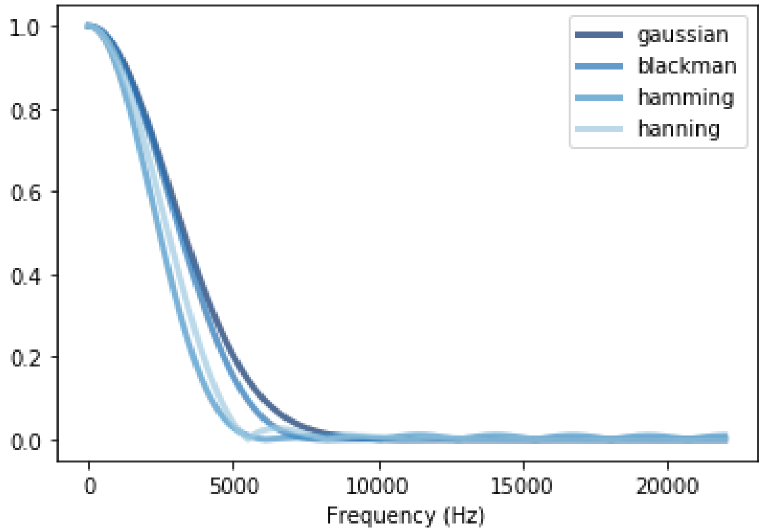

Рис. 3.2: Визуализация ДПФ

Тоже очень похоже, но похоже, что Гауссово падает быстрее всех, Блэкман - самым медленным, а у Ханнинга самые заметные боковые лепестки.

```
plot_window_dfts(windows, names)
thinkplot.config(xlabel='Frequency (Hz)', yscale='log',
                 loc='lower left')
```

Листинг 3.6: Визуализация ДПФ

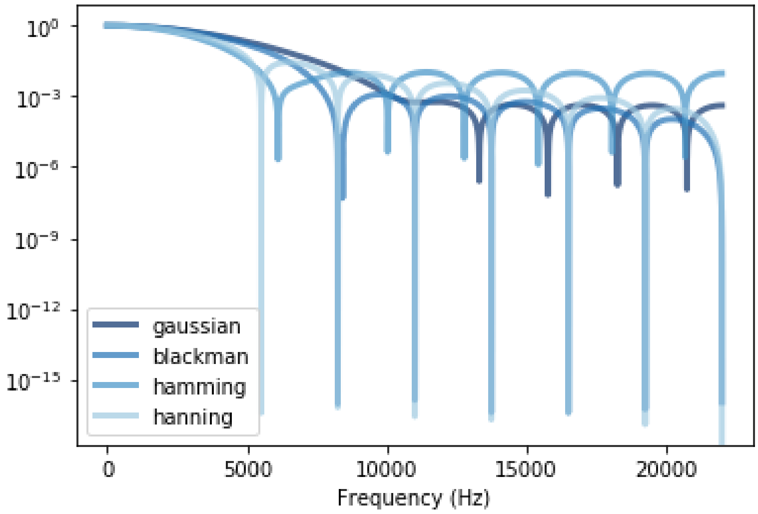

Рис. 3.3: Визуализация ДПФ

В логарифмической шкале мы видим, что сначала значения Хэмминга и Хеннинга падают быстрее, чем два других. И окна Хэмминга и Гаусса, кажется, имеют самые стойкие боковые лепестки. Окно Ханнинга, кажется, имеет наилучшее сочетание быстрого спада и минимальных боковых лепестков.

Глава 4

Выводы

Во время выполнения лабораторной работы получены навыки работы с концепцией свёртки и теоремой свёртки, а также научился применять эти знания на практике.