Санкт-Петербургский государственный политехнический
университет Петра Великого.
**Высшая школа интеллектуальных сисем и
суперкомпьютерных технологий**

Лабораторная работа

Фильтрация и свертка.

Работу выполнила студентка:
_____ А. И.Луцкевич
«____» _____ 2021 г.

Преподаватель лабораторных
работ:
_____ Н. В.Богач
«___» _____ 2021 г.

Санкт-Петербург, 2021 г.

Суть работы 8.1:

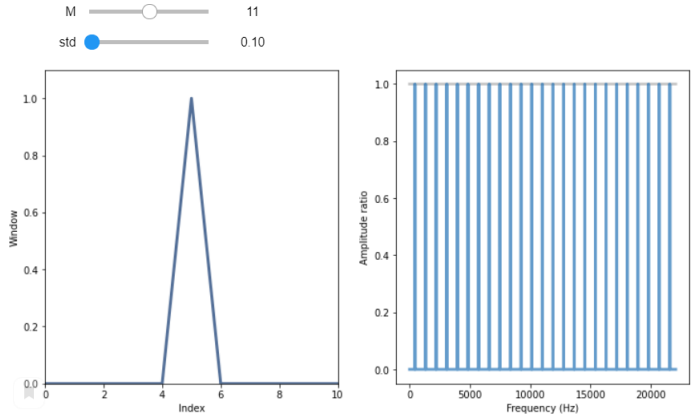
Необходимо запустить весь код из блокнота chap08.ipynb и проверить, что будет при увеличении ширины гауссова окна std, не увеличивая число элементов в окне m.

Скопируем функцию plot_filter из chap08.ipynb и напишем интерактивный виджет для использования.

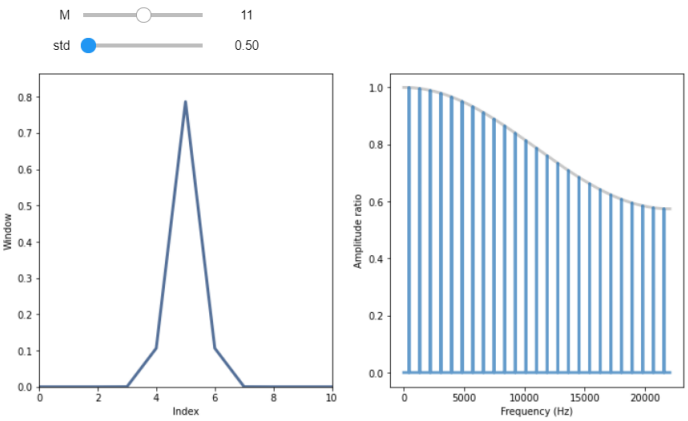
In [3]:
```
def plot_filter(M=11, std=2):
    signal = SquareSignal(freq=440)
    wave = signal.make_wave(duration=1, framerate=44100)
    spectrum = wave.make_spectrum()

    gaussian = scipy.signal.gaussian(M=M, std=std)
    gaussian /= sum(gaussian)
    high = gaussian.max()
    thinkplot.preplot(cols=2)
    thinkplot.plot(gaussian)
    thinkplot.config(xlabel='Index', ylabel='Window',
    xlim=[0, len(gaussian)-1], ylim=[0, 1.1*high])

    ys = np.convolve(wave.ys, gaussian, mode='same')
    smooth =  Wave(ys, framerate=wave.framerate)
    spectrum2 = smooth.make_spectrum()

    amps = spectrum.amps
    amps2 = spectrum2.amps
    ratio = amps2 / amps
    ratio[amps<560] = 0

    padded =  zero_pad(gaussian, len(wave))
    dft_gaussian = np.fft.rfft(padded)

    thinkplot.subplot(2)
    thinkplot.plot(abs(dft_gaussian), color='0.7', label='Gaussian filter')
    thinkplot.plot(ratio, label='amplitude ratio')
    thinkplot.show(xlabel='Frequency (Hz)', ylabel='Amplitude ratio')
```

In [4]:
```
slider = widgets.IntSlider(min=2, max=100, value=11)
slider2 = widgets.FloatSlider(min=0, max=20, value=2)
interact(plot_filter, M=11, std=slider2);
```

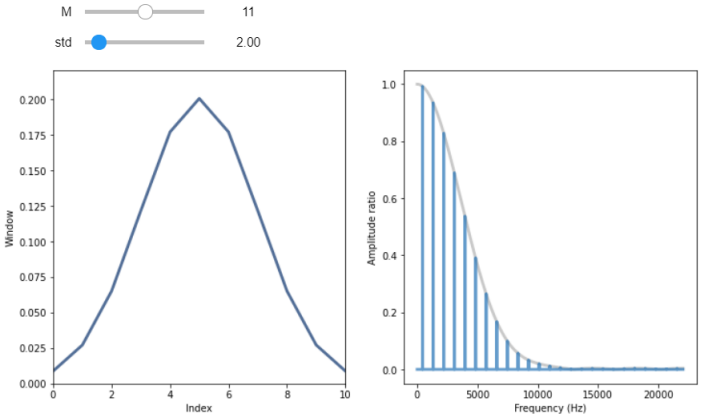
Теперь по заданию будет менять только std, значение m будет по умолчанию всегда равно 11.
Пойдем на увеличение, начнем с std равное 0.1.



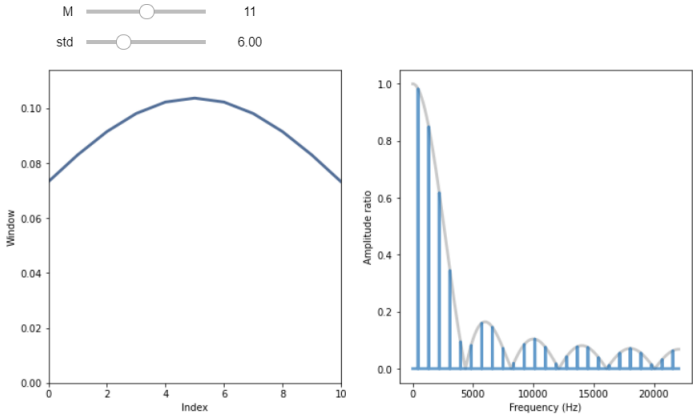
Теперь std будет равно 0.5.



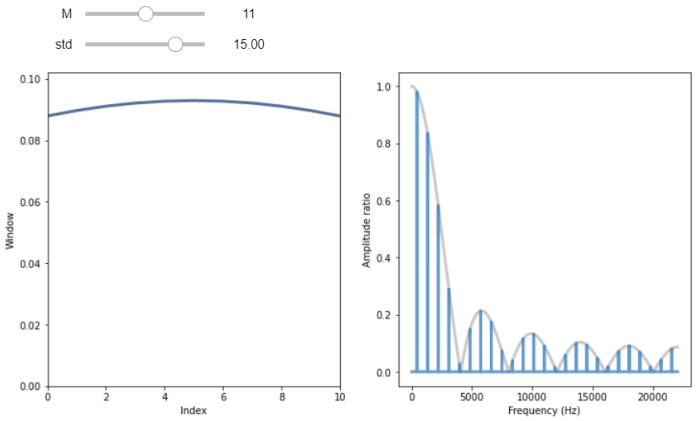
Поставим std равным 2.



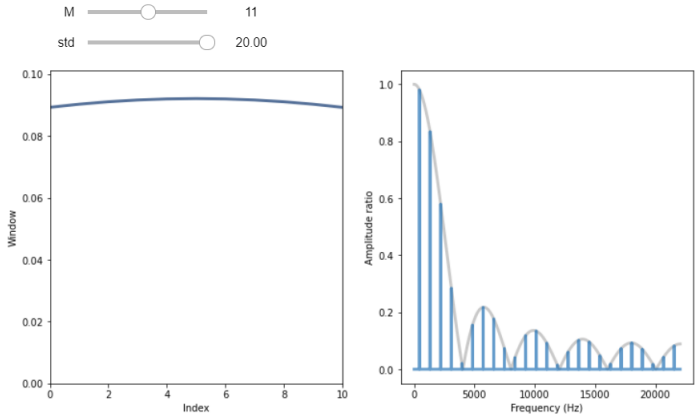
Поставим std равным 6.



Поставим std равным 15.



И в конце поставим максимальное значение std равным 20.



Из этого можно сделать вывод, что при увеличении std происходит "выпрямление"гауссовой прямой.

Суть работы 8.2:

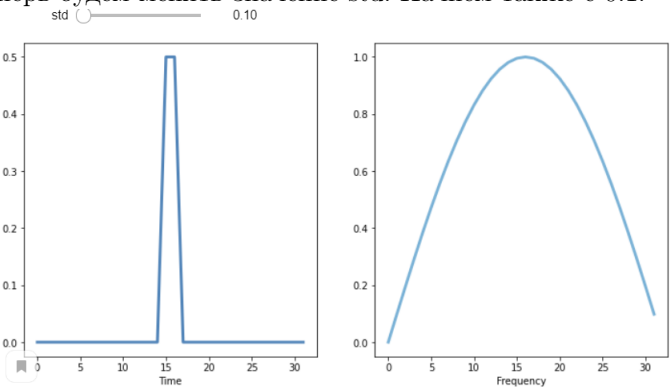
Необходимо попробовать ДПФ на нескольких примерах и понять, что происходит при изменении std.

Реализуем функцию plot_gaissian. Она будет отображать окно Гаусса и БСП. Также напишем интерактивный виджет.

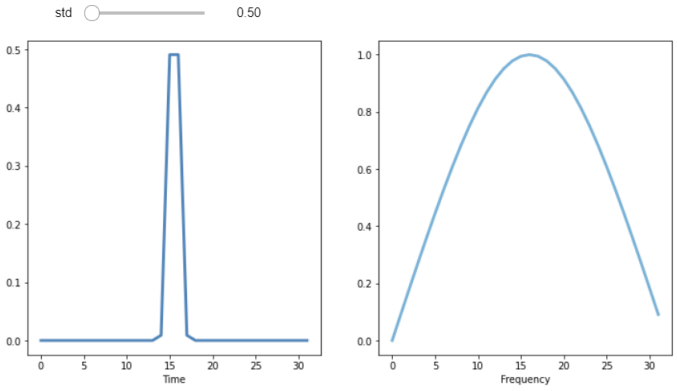
In [5]:
```
def plot_gaussian(std):
    gaussian = scipy.signal.gaussian(M=32, std=std)
    gaussian /= sum(gaussian)

    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, 32//2)

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

In [6]:
```
slider = widgets.FloatSlider(min=0.1, max=10, value=2)
interact(plot_gaussian, std=slider);
```

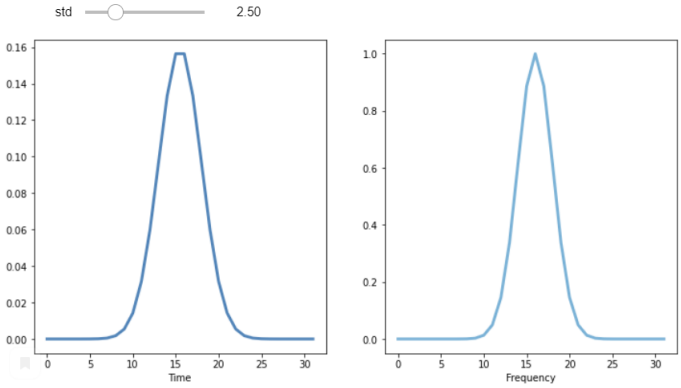
Теперь будем менять значение std. Начнем также с 0.1.



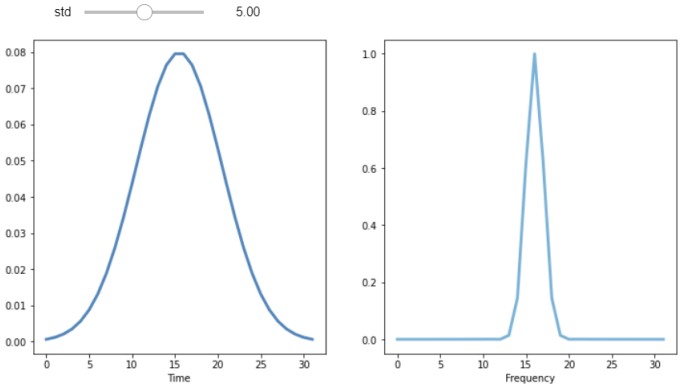
Поставим std равным 0.5.



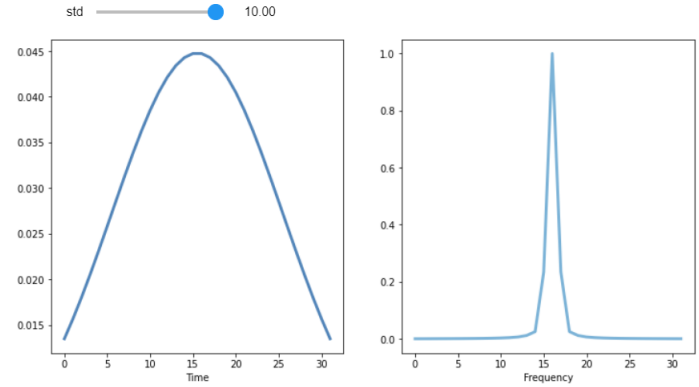
Поставим std равным 2.5.



Поставим std равным 5.



И в конце поставим максимальное значение std равным 10.



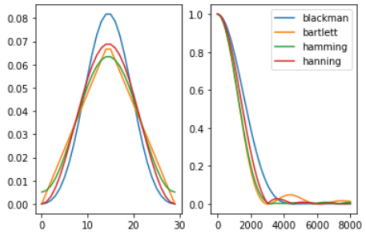
При увеличении ширины сигнала уменьшается ширина преобразования Фурье, а аналогично наоборот.

Суть работы 8.3:

необходимо в дополнение к Гауссову окну, использованному ранее создать окно Хэмминга тех же размеров. Также нужно дополнить окно нулями и напечатать его ДПФ. Определить, какое окно больше подходит для фильтра НЧ.

Напишем функцию plot_window, чтобы решить данную задачу. И запустим данную фукцию.

In [7]:
```
def plot_window(ax, window_fun, M=30):
    signal = SquareSignal(freq=440)
    wave = signal.make_wave(duration=1.0, framerate=44100)

    window = window_fun(M)
    window /= sum(window)

    padded = zero_pad(window, len(wave))
    fft = np.fft.rfft(padded)

    ax[0].plot(window, label=window_fun.__name__)
    ax[1].plot(abs(fft)[:8000], label=window_fun.__name__)
    plt.legend()

_, ax = plt.subplots(1, 2)
for w in [np.blackman, np.bartlett, np.hamming, np.hanning]:
    plot_window(ax, w)
```



Исходя из рисунка, можно увидеть, что для фильтрации НЧ надо использовать окно Хэмминга, потому что у него меньше выпуклостей.

Заключение:

В данной лабороторной работы были изучена зависимость ширины гауссова окна std на гауссовую кривую. Также были написаны две фукнции для отображения окна Гаусса (с выводом БПФ) и окна Хэмминга. Также был сделан вывод, что для фильтрации НЧ лучше использовать окно Хэмминга.