РАБОТА №7. СВЕРТКА И ДЕКОНВОЛЮЦИЯ

Цель работы: изучение свертки и обратной ей операции на примере простых сигналов и их суммы.

Планируемая продолжительность: от 2 до 4 академических часов.

Тип работы:  с использованием компьютерных средств.

Теоретические основы

Сверткой сигналов называют интеграл произведения двух сигналов, причем один из сигналов сдвигается с опережением относительно другого. То есть, если взаимная корреляция сигналов $f(t)$ и $g(t)$ описывается выражением:

$$(f \times g)(\tau) = \int_{-\infty}^{\infty} \bar{f}(t) g(t-\tau)\, dt,$$

то свертка может быть описана похожим выражением:

$$(f \times g)(t) = \int_{-\infty}^{\infty} \bar{f}(\tau) g(t-\tau)\, d\tau.$$

исходя из чего, можно выявить связь взаимной корреляции и свертки:

$$(f \times g)(t) = f(t) \times g(t) = f(-t) * g(t),$$

где $\bar{f}(t)$ и $\bar{f}(\tau)$ - комплексно сопряженные, соответственно, $f(t)$ и $f(\tau)$ сигналы, и в случае используемых нами действительных сигналов можно заменить самими сигналами их комплексно сопряженные копии.

Графически связь ВКФ и свертки представлена ниже. Если оба сигнала являются четными, то их свертка равна их же ВКФ. Упрощенно график свертки двух функций можно интерпретировать также, как и ВКФ – локальная «похожесть» двух функций при соответствующем сдвиге, но, в отличие от ВКФ, при свертке второй сигнал «переворачивается», то есть, для дискретных сигналов, первый элемент свертки определяется как$f_1 \cdot g_1$, (при максимальном отрицательном сдвиге), второй – как $f_1 \cdot g_2 + f_2 \cdot g_1$, третий – как $f_1 \cdot g_3 + f_2 \cdot g_2 + f_3 \cdot g_1$ и т.д.

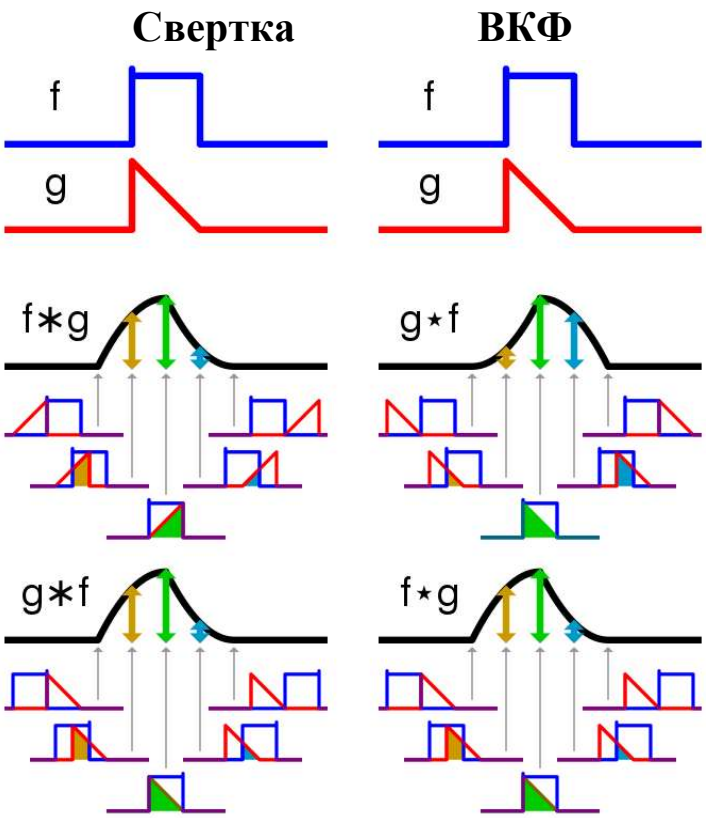


Рис. 7.1. Связь свертки и ВКФ

Часто исследуется случай, когда свертке с одним сигналом фиксированной формы и длительности подвергаются различные по форме и длительности сигналы, при этом фиксированный сигнал, как правило, значительно короче произвольных. В этом случае фиксированный сигнал называют ядром свертки. Также ядром свертки часто называют сдвигаемый сигнал (в выражении свертки), то есть, в текущих обозначениях, $g(t-\tau)$.

Также $g(t-\tau)$ в практических задачах измерительной техники характеризует свойства системы, через которую пропускается сигнал $f(t)$, в результате чего получается искаженная его копия (результат свертки). В этом случае $g(t-\tau)$ называют **импульсной характеристикой** данной системы (причем это может быть как некоторая измерительная система или блок, так и сама среда или канал связи, искажающие воздействия которых и задаются импульсной характеристикой).

Если f и g – дискретные сигналы, свертку для них можно, в общем случае записать выражением:

$$(f * g)_k = \sum_{m=1}^{M} f_m \cdot g_{k-m}, \quad (9.1)$$

где m – сдвиг;

k – номер элемента в массиве сигнала f и в массиве свертки;

M – число элементов в сигнале M.

Очевидно, что в сигнале g по выражению (6.1) можно ссылаться на несуществующие элементы. Для исключения вызванных этим ошибок следует изменить (6.1):

$$(f * g)_k = \sum_{m=k-N}^{k-1} f_m \cdot g_{k-m}, \quad (9.2)$$

а также проверять выход за границы f:

$$f_m = 0, \text{если } (m < 1) \vee (m > M),$$

где N – число элементов в массиве g.

Результат свертки будет иметь размер K=M+N-1, так как g будет пересекаться с f левым краем при k, близких к M, поэтому (9.2) может не быть равным нулю при k>M.

Свертка в Python

Импортируем следующие зависимости.

```
1  import numpy as np
2  import matplotlib.pyplot as plt
3  from scipy import signal
4  from random import randint
5  %matplotlib widget
6  # from google.colab import output
7  # output.enable_custom_widget_manager()
```

Рис. 7.2. Импортирование зависимостей

Если в работе применяется Google Colab в качестве редактора, то для работы интерактивных графиков нужно раскоментировать последние две строчки из ячейки импортов.

Также из предыдущей работы импортируем функцию plot.

```
def plot(*args):
    ax = plt.figure()
    for y in args:
        x = np.arange(y.size)
        plt.plot(x, y)
    plt.grid(True)
    plt.ylabel('y')
    plt.xlabel('x')
    plt.show()
```

Рис. 7.3. Функция отрисовки графиков

Чтобы задать выражение для свертки двух дискретных сигналов, можно прибегнуть к алгоритму, похожему на задание ВКФ.

```
def convolve_element(
        f_array: np.ndarray,
        g_array: np.ndarray,
        k: int) -> np.ndarray:
    """ Свёртка элемента """

    def get_f(idx: int):
        """
        Получаем значение f
        Если индекс выходит за пределы массива, то возвращаем 0
        """
        if idx >= 0 and idx < f_array.size:
            return f_array[idx]
        return 0

    result = np.zeros(g_array.size)
    for idx in range(k - g_array.size, k):
        result[g_array.size - (k - idx)] = get_f(idx) * g_array[k - idx - 1]
    return np.sum(result)
```

```
def convolve(
    array_1: np.ndarray,
    array_2: np.ndarray
) -> np.ndarray:
    """ Свёртка двух массивов """
    result = np.zeros(array_1.size + array_2.size)
    for idx in range(1, result.size):
        result[idx] = convolve_element(array_1, array_2, idx)
    return result
```

Рис. 7.4. Алгоритм свёртки двух массивов

При этом в данной работе отдельно задавать массив сдвигов не требуется, так как все значения сдвига – положительные целые числа, отдельный массив является избыточной величиной, достаточно просто верно задать итератор в цикле for.

Зададим два сигнала, построим их графики и графики свертки (скобку на подписи графика для вставки двух массивов можно найти на панели Функции, в самом конце). Стоит обратить внимание, что сигналы могут отличаться по размеру (что чаще всего наблюдается в рамках практических задач).

```
y_1 = np.array([1., 1.2, 0., 1.1, 1.3, 0, 1.1, 0])
x_1 = np.arange(stop=y_1.size)
y_2 = np.array([0.5, 2, 0.5])
x_2 = np.arange(stop=y_2.size)
plot(y_1, y_2)
```

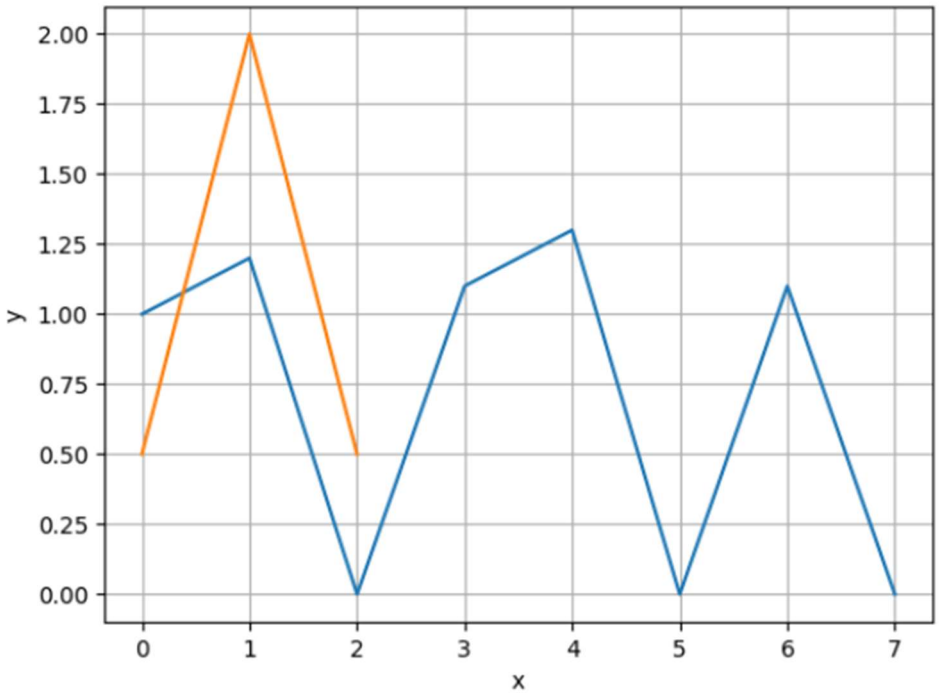


Рис. 7.5. Сигналы

```
convolved = convolve(y_1, y_2)
expected = np.zeros(convolved.size)
expected[1:] = signal.convolve(y_1, y_2, mode='full')
np.testing.assert_array_equal(convolved, expected)
plot(convolved)
```

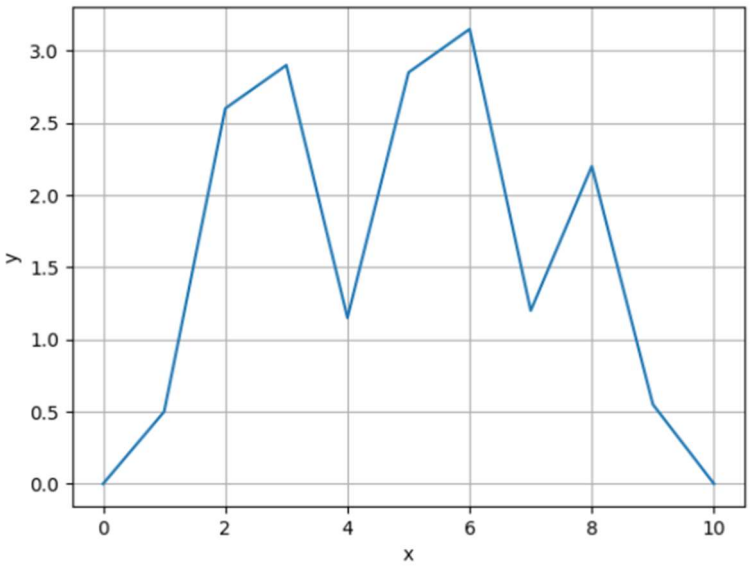


Рис. 7.6. Результат свёртки

Слева синим цветом показан исходный сигнал, жёлтым – функция, с которой он сворачивается. Такая процедура часто имеет место в измерительной технике – многие приборы обладают собственной импульсной характеристикой (например, фильтры), и при прохождении через них сигналов происходит свертка сигналов с импульсной характеристикой. В данном случае исходный сигнал задан тремя простыми импульсами, пример, который может быть использован в качестве импульсной характеристики – треугольным одиночным импульсом.

Деконволюция в частотной области

Если продолжить рассматривать свертку в данной работе, как результат взаимодействия некоторого сигнала и импульсной характеристики прибора, следует считать, что сдвигу подвергается импульсная характеристика, поскольку она на практике практически во всех случаях меньше по длительности, чем сам сигнал. А из выражений выше видно, что размер массива свертки определяется как 2(rows(ar1)-1), поэтому для сигналов разной длины использование в качестве ar1 самого короткого из них даст неполный результат.

Следовательно, считая неподвижным исходный сигнал, получаем график, на котором еще различимы (но очень слабо) три исходных импульса.

Задача деконволюции – обратная свертке задача. В сущности, в русскоязычной литературе данный термин часто встречается именно как обратная свертка (реже развертка). Задача деконволюции в измерительной технике – восстановить исходный сигнал, искаженный прибором, а именно его импульсной характеристикой, которая должна быть известна (бывают и иные случаи, которые здесь не рассматриваются).

Восстановить исходный сигнал можно, если прибегнуть к преобразованию Фурье, поскольку известно следующее свойство свертки:

$$F[f(t) * g(t)] = F[f(t)] \cdot F[g(t)],$$

где F[] – преобразование Фурье.

То есть, получить исходный сигнал f(t) можно, выразив его из выражения выше:

$$f(t) = F^{-1}\left[\frac{F[f(t) * g(t)]}{F[g(t)]}\right],$$

где F^{-1}[] – обратное преобразование Фурье.

Реализация деконволюции в Python

Воспользуемся библиотекой SciPy для свёртки сигнала, так как данный пакет учитывает особенности работы NumPy. При реализации напрямую выражений из вышеописанных функций, результатом данных функций будут числа выходящий за пределы разрядности чисел, что естественно будет ошибкой.

Создадим массив для свёртки и импульсную характеристику. Функцией signal.convolve произведём свёртку, и выведем полученные данные на графике.

```
original = np.array([0, 1, 0, 0, 1, 1, 0, 0])
impulse_response = np.array([2, 1])
convolved = signal.convolve(impulse_response, original)
plot(original)
plot(convolved)
```

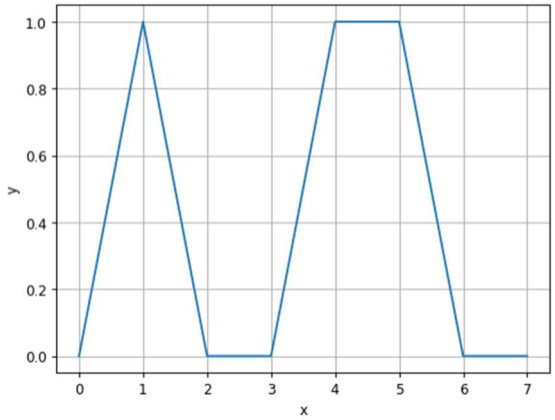


Figure 27

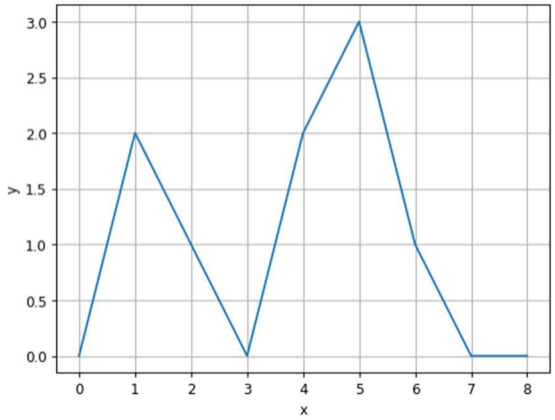


Рис. 7.7. Свёртка сигнала

Важно учитывать, что импульсная характеристика не должна быть существенно более длительная, чем отдельные компоненты сигнала – иначе детали теряются полностью и данный способ восстановить исходный сигнал не позволяет (то есть если имеются проблемы в работе описанного алгоритма – в первую очередь смотрите на длительность второго сигнала; значения равные нулям справа и слева от отличной от нуля области здесь не учитываются).

Деконволюция происходит также функцией deconvolve из того же модуля.

```
1  recovered, _ = signal.deconvolve(convolved, impulse_response)
2  plot(recovered)
```

Рис. 7.8. Деконволюция сигнала

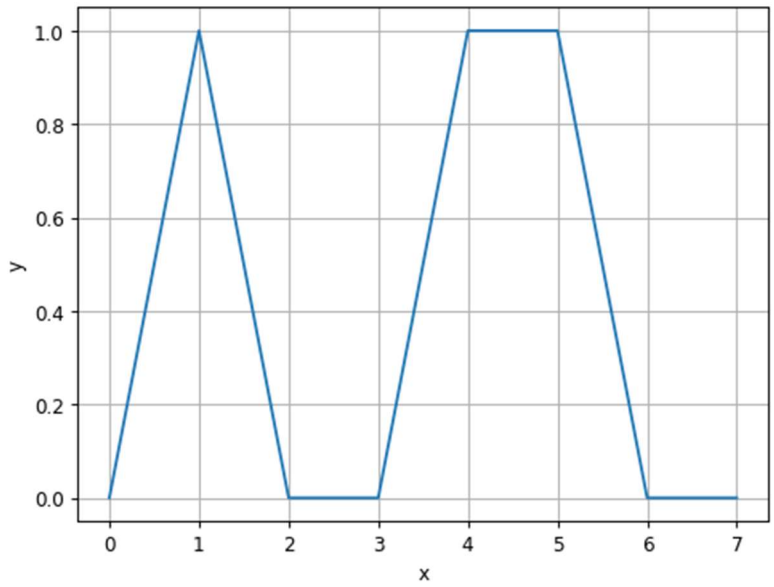


Рис. 7.9. График деконволированного сигнала

Свертка сигнала с функцией Гаусса

Теоретические основы

Перейдем к примеру, который куда ближе к практике. Имеется прибор с импульсной характеристикой, заданной функцией y(x).

Эта функция, y(x), называется кривой Гаусса и в данном контексте может говорить о том, что прибор понижает разрешение выходного сигнала, по сравнению с входным, «смазывает» его. Такое можно наблюдать, например, при фотографировании с дефокусом – расфокусировка может быть устранена деконволюцией с использованием функции Гаусса в качестве импульсной характеристики. Также фильтры, размывающие фон на фото в режиме портрета в камере смартфона, а также фильтр Размытие по Гауссу в Adobe Photoshop выполняют именно данную процедуру – свертку исходного изображения или его частей с функцией Гаусса.

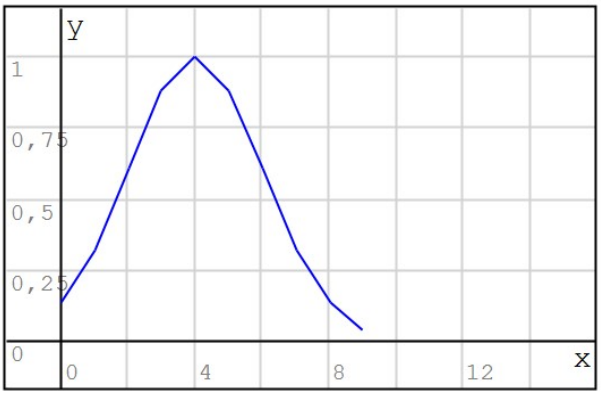


Рис. 7.10. Функция Гаусса

В измерительной технике такая импульсная характеристика свойственна многим приборам и связана с их инерционностью. К примеру, в спектрометрии можно встретить спектр вида, представленного на рисунке.

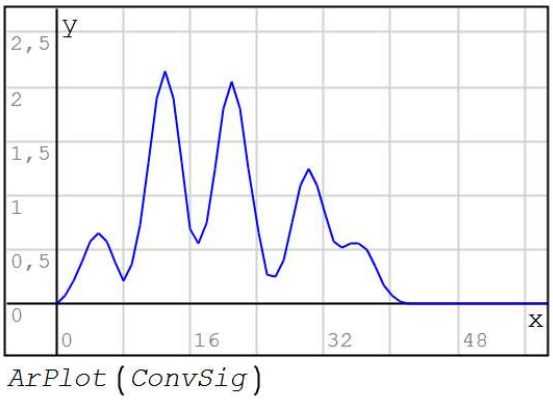


Рис. 7.11. Свернутый спектр

Тогда как действительный (или истинный) спектр имеет вид, показанный ниже.

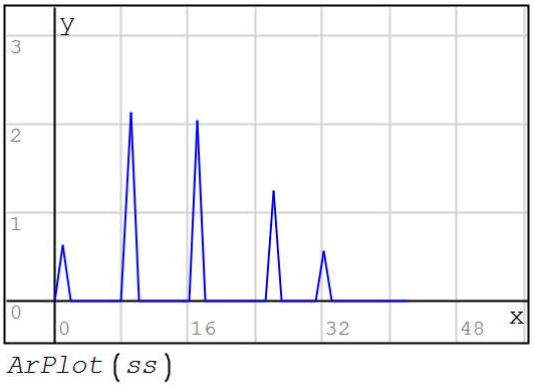


Рис. 7.12. Действительный спектр

Можно заметить, что помимо растяжения и сглаживания четких одиночных пиков, свертка с импульсной характеристикой вызывает смещение максимальных значений пиков вправо. Если представленные выше графики – спектры частот, то смещение положения пиков по оси абсцисс вызывает погрешность в определении частоты исследуемого процесса, что куда серьезнее погрешности в определении амплитуды.

Пример реализации

В качестве примера зададим сигнал, подобный приведенному выше, для чего сперва спроектируем функцию, генерирующую массив, в котором содержится заранее заданное число пиков со случайными параметрами. Зададим сигнал по этой функции и выведем его.

```
def form_signal(n: int) -> np.ndarray:
    tmp = [0]
    for _ in range(n):
        tmp.append(0.25 + randint(0, 100) / 50)
        for _ in range(10 - randint(0, 8)):
            tmp.append(0)
    return np.array(tmp)
```

```
signal_s = form_signal(5)
plot(signal_s)
```

Рис. 7.13. Задание сигнала

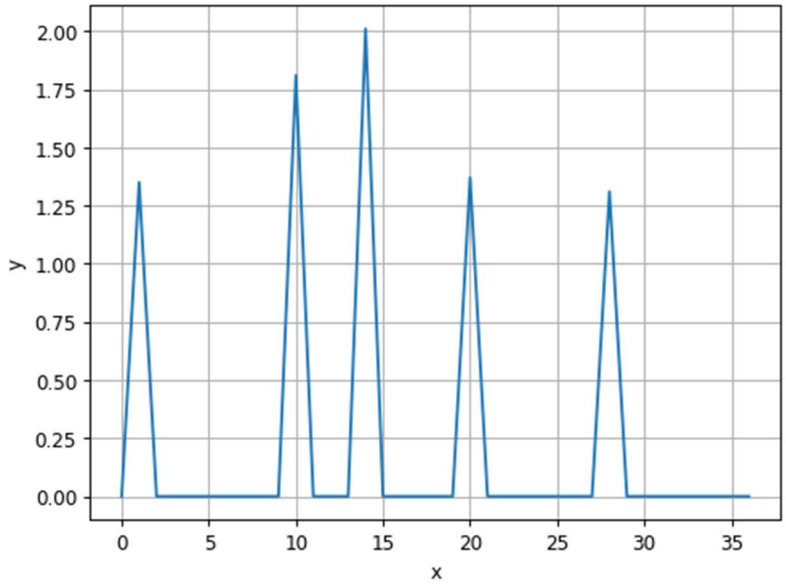


Рис. 7.14. График сигнала

Для ss вычислим его свертку с импульсной характеристикой по функции Гаусса. Построим график свертки (при пересчете будет новый график из-за случайности параметров пиков).

```
def y(x, w):

    def numerator():
        degree = (-1) * (x ** 2) / (2 * (w ** 2))
        return np.power(np.e, degree)

    def denominator():
        return np.sqrt(2 * np.pi) * w

    return numerator() / denominator()
```

```
N = 10
DX = 1 / N
x = -0.5 + DX * np.arange(start=1, stop=N + 1)
IR = y(x, 1 / 5)
convolved_signal = signal.convolve(signal_s, IR, mode='full', method='direct')
plot(convolved_signal)
```

Рис. 7.15. Алгоритм свёртки функцией Гаусса

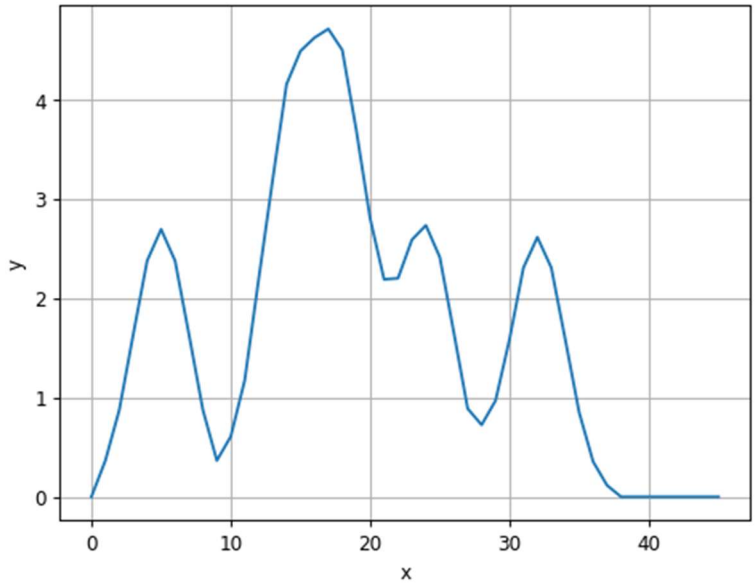


Рис. 7.16. График свёрнутого сигнала

Теперь, поскольку параметры импульсной характеристики заранее известны, вычислить деконволированный сигнал по приведенному ранее алгоритму не составит трудностей. Здесь мы вместо IR вызываем функцию y() с известным заранее параметром w (1/5) – сейчас процедура выполняется для примера, в дальнейшем же понадобится для реализации последнего раздела и выполнения соответствующего задания.

```
1  deconvolved_signal, _ = signal.deconvolve(convolved_signal, y(x, 1 / 5))
2  plot(np.array(deconvolved_signal))
```

Рис. 7.17. Деконволюция и вывод графика результата

Построим графики результата деконволюции и подобранной импульсной характеристики. Как видим, график сигнала больше не имеет «сглаженностей», содержит только короткие одиночные импульсы.

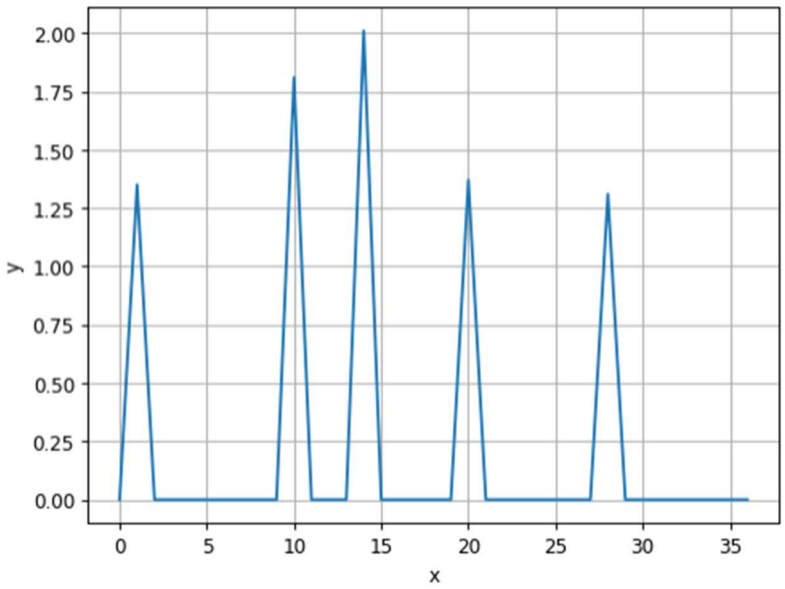


Рис. 7.18. График деконволированного сигнала

Деконволюция сигнала с использованием априорной информации о форме импульсной характеристики и последовательным ручным уточнением ее параметров

Теоретические основы

Для случаев, когда импульсная характеристика заранее неизвестна, ее пытаются оценить, обычно статистическими методами. Для некоторых случаев, например, когда известна функция, задающая импульсную характеристику, но не известны ее параметры.

Чем ближе ширина пика к требуемому значению, тем ближе деконволированный сигнал к исходной форме, в случае, если сигнал состоит из отдельных резких, преимущественно положительных пиков, на графике деконволюции будет все меньше искажений, особенно в отрицательной области.

Такой результат может быть получен, если ширина импульсной характеристики выбрана больше требуемой (справа – деконволюция, слева - исходный).

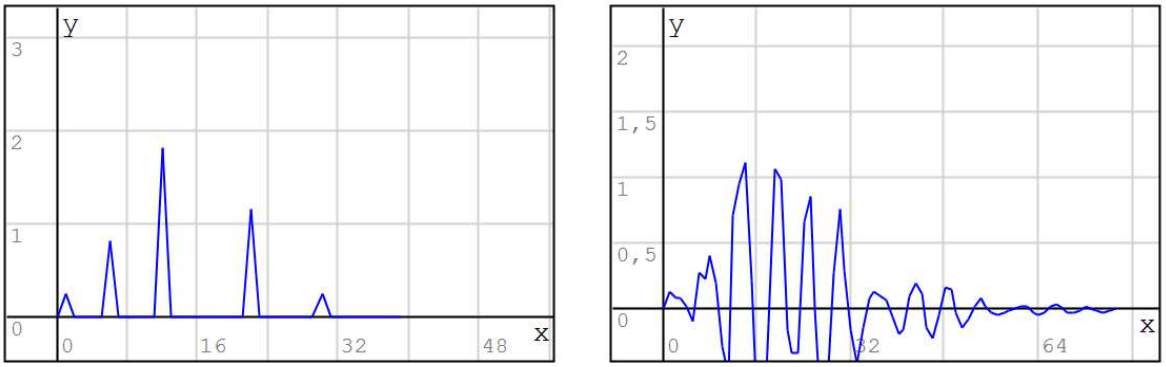


Рис. 7.19. Результат при превышении максимально возможной ширины

Результат ниже – если ширина меньше требуемой.

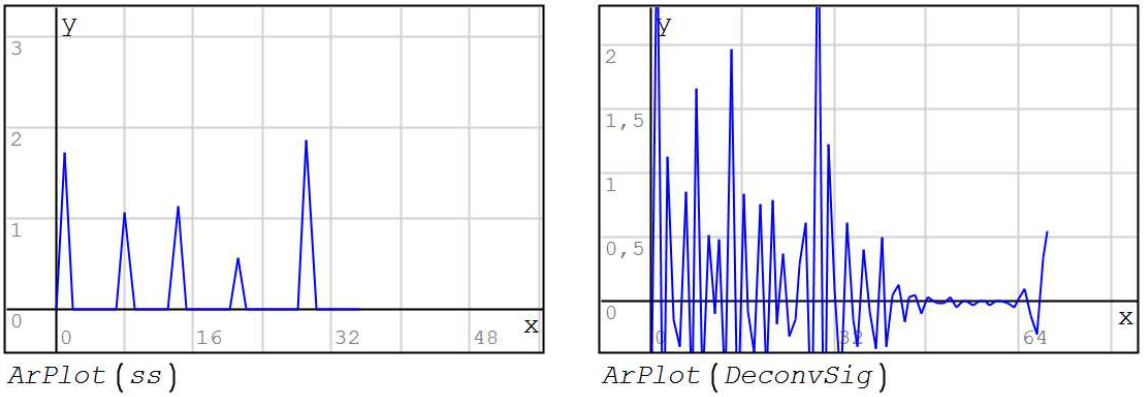


Рис. 7.20. Результат при занижении минимальной ширины

Графики ниже характеризуют близкое к требуемому значение, при его небольшом превышении.

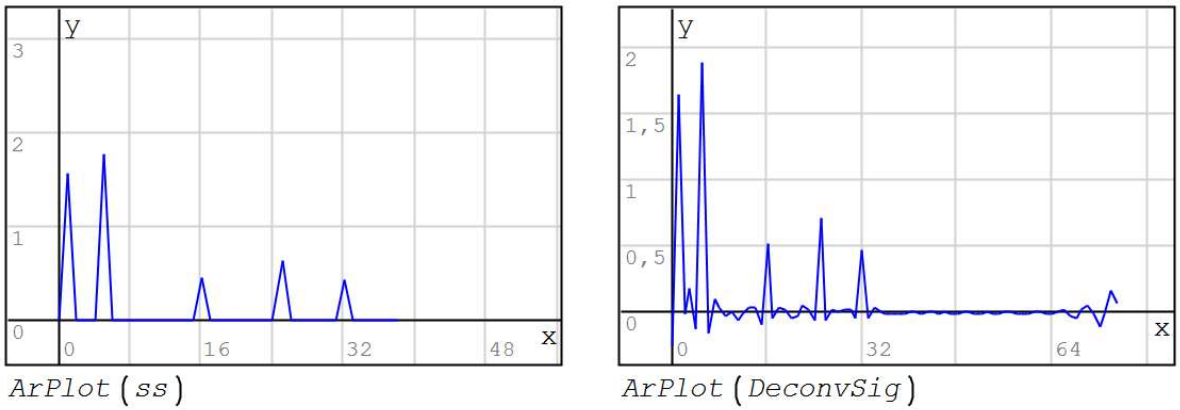


Рис. 7.21. Приближение с большей стороны

Графики ниже характеризуют близкое к требуемому значение, при его небольшом занижении.

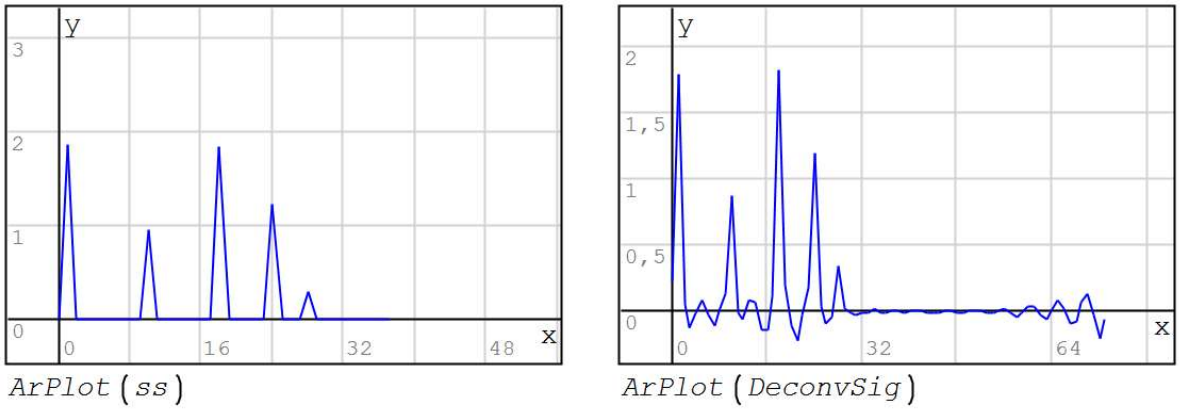


Рис. 7.22. Приближение с меньшей стороны

Таким образом, деконволировать некоторые простые сигналы можно даже вручную – если известна форма импульсной характеристики прибора, но неизвестны ее параметры.

Пример реализации

Рассмотрим нередкий случай, когда известна в общем виде форма импульсная характеристики, но не ее параметры. Допустим, импульсная характеристика задана функцией Гаусса неизвестной ширины. При этом сигнал, являющийся результатом конволюции исходного сигнала с этой функцией, представлен ниже.

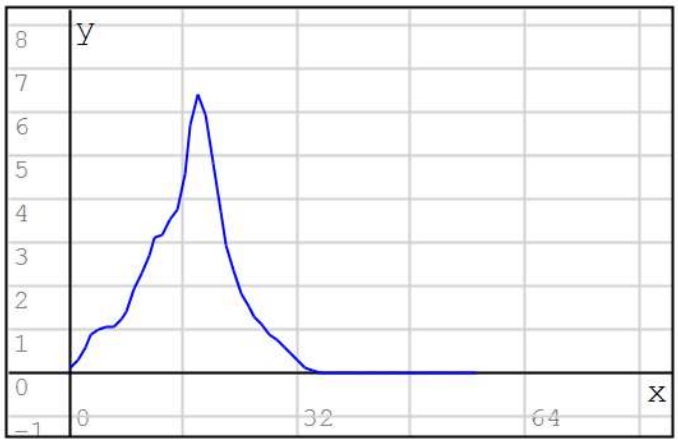


Рис. 7.23. Результат свертки

Для решения задачи воспользуемся уже заданными в разделе «Свертка сигнала с функцией Гаусса». В resp заменим параметр ширины функции y() с 1/5 на 1. Применим спроектированный ранее алгоритм деконволюции к исследуемому сигналу.

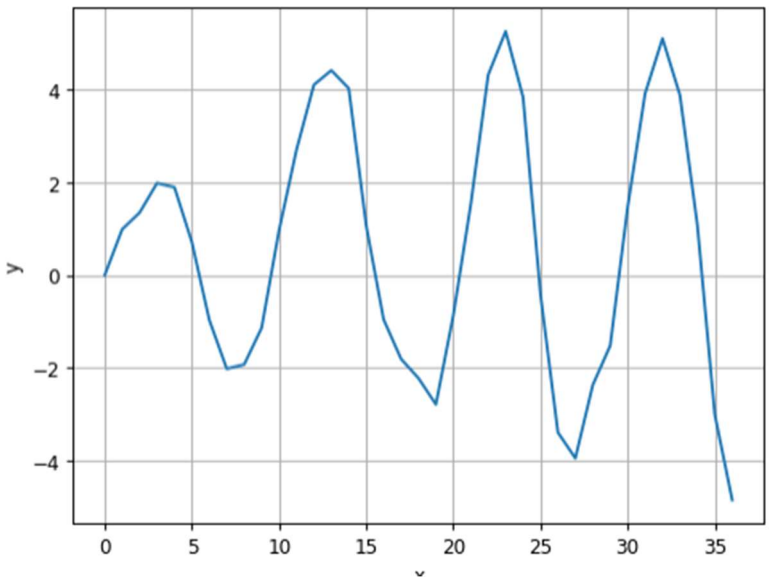


Рис. 7.24. Первичная деконволюция

Видим практически синусоиду на выходе. Учитывая, что входной сигнал был строго положительным после свертки, можно быть уверенным, что до свертки с функцией Гаусса он также был исключительно положительным, а, значит, полученный результат является следствием неверно выбранного параметра ширины импульсной характеристики.

Уменьшим ширину импульсной характеристики вдвое, для чего в выражении resp сменим внутри функции y() параметр w с 1 на 1/2. Полученная импульсная характеристика и результат деконволюции ниже.

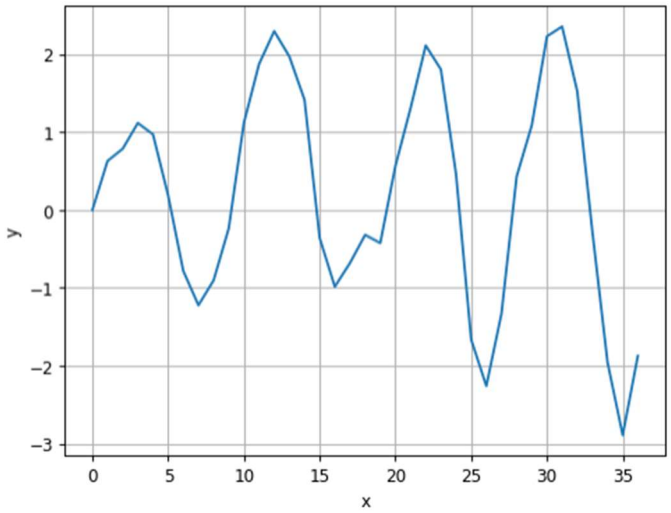


Рис. 7.25. Уточнение деконволюции

Видно, что на графике деконволюции появляются непериодические составляющие, что может указывать на приближение к истинной форме исходного сигнала.

Изменим параметр w на 1/4.

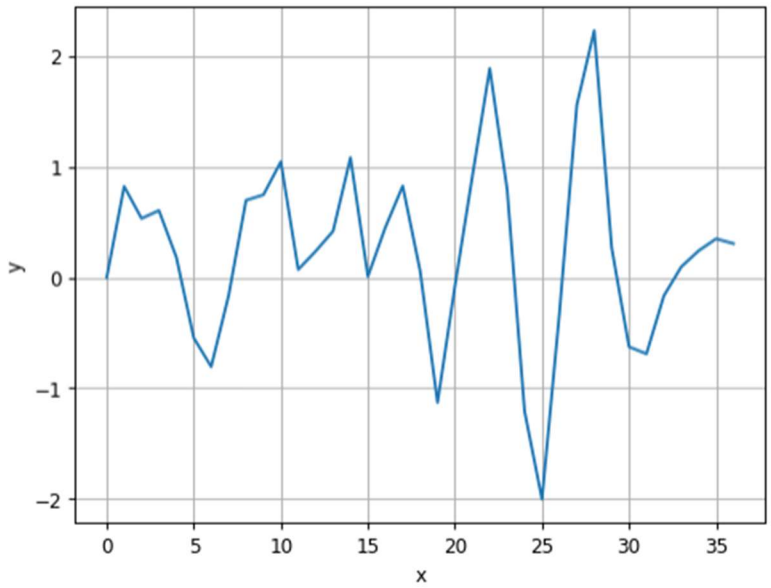


Рис. 7.26. Уточнение деконволюции

Как видим, непериодические детали проявляются все сильнее. Попробуем принять параметр w равный 1/5,5.

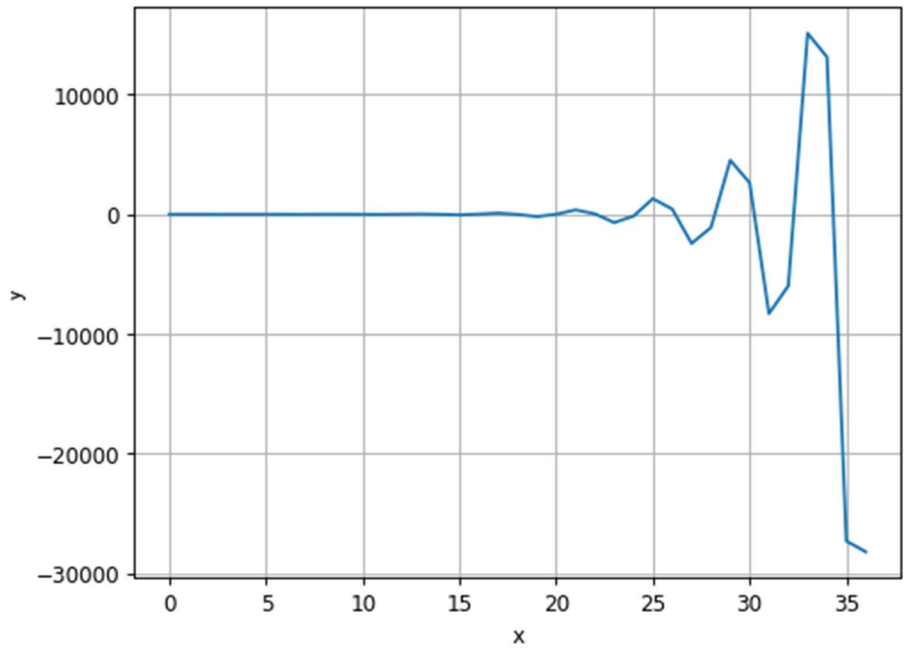


Рис. 7.27. Уточнение деконволюции

В данном случае всплеск в конце указывает, на то, что происходит отдаление от истинной формы сигнала. Этот всплеск является высокочастотной составляющей.

Попробуем уменьшить параметр w до 1/6, чтобы уточнить результаты.

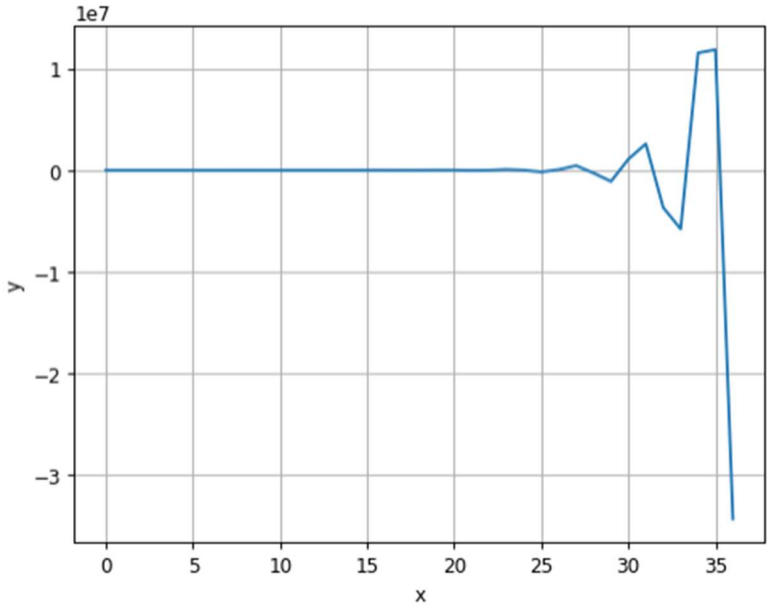
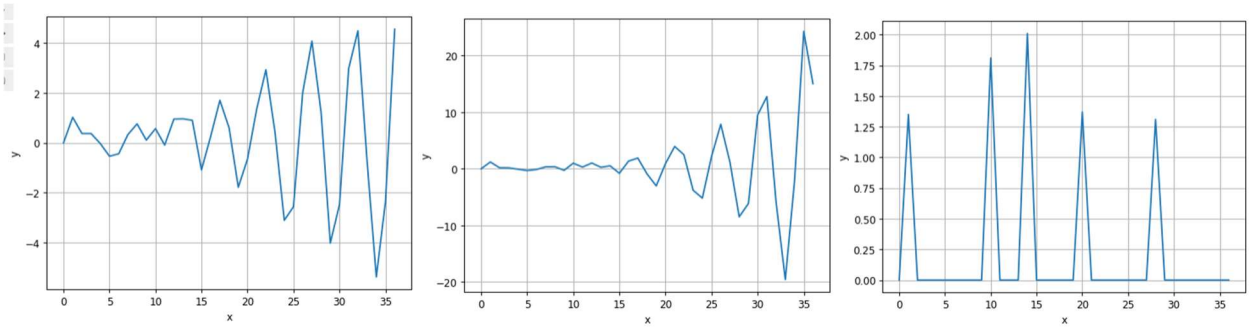


Рис. 7.28. Уточнение деконволюции

Высокочастотная переменная составляющая нарастает, значит, определенно требуемый множитель ширины был пропущен. Исходя из анализа приведенных графиков, можно видеть, что исходный сигнал был подвергнут свертке с функцией Гаусса с параметром w в диапазоне 1/5,5…1/4.

Дальнейшее уточнение результата представлено ниже.



w=1/4,5 w=1/4,8 w=1/5

Рис. 7.29. Последовательное уточнение

Таким образом, исходный сигнал представлен на последнем графике, а подобранная импульсная характеристика описывается выражением $\frac{e^{-\frac{x^2}{2\left(\frac{1}{5}\right)^2}}}{\sqrt{2\pi}\left(\frac{1}{5}\right)}$ или, упрощенно $\frac{5e^{-\frac{25x^2}{2}}}{\sqrt{2\pi}}$.

Задание

1. Реализовать пример для вычисления свертки и деконволюции (второй и четвертый разделы).
2. Опробовать свертку и деконволюцию для заданных сигнала и импульсной характеристики (таблица 9.1), заменив указанным сигналом тот, что использовался в задании 1.
3. Реализовать пример свертки сигнала с функцией Гаусса и его деконволюции по пятому разделу.
4. Задать вместо сигнала ss в задании 3 тот, что определен вариантом (таблица 9.2).
5. Решить задачу определения параметров импульсной характеристики (функции Гаусса) по примеру в последнем разделе описания данной работы. При описании приводить все промежуточные результаты, обосновывать выбор того или иного множителя от этапа к этапу. Начальное приближение w брать равным 1. Менять только w (все по примеру в последнем разделе), причем:
6. искомое значение w не может быть менее 1/10 и более 1;
7. искомое значение w во всех вариантах представляет собой дробь вида $\frac{1}{c}$, где c – некоторое целое или дробное число;
8. c может иметь не более одного знака после запятой (к примеру, допускаются $c = 5{,}1$ и $w = \frac{1}{5{,}1}$, $c = 7$ и $w = \frac{1}{7}$, $c = 4{,}4$ и $w = \frac{1}{4{,}4}$ и т.п.).

Таблица 9.1

Массивы для второго задания

Вариант	**Сигнал**	**Импульсная характеристика**
1,6,11, 16,21,26	0.3,1.2,4,2.1,3.3,1.2,1.7,0	2,0,1
2,7,12, 17,22,27	0,1,0,2.5,2.6,1.1,0.7,0	1,0,0
3,8,13, 18,23,28	3,2.2,4,2.7,0.3,4.2,2.6,1.4	0,1,0
4,9,14, 19,24,29	1,1,1,3,1.7,2.1,3.7,1	0.5,1,0
5,10,15, 20,25,30	1.4,0.2,1,1.5,1.6,1.7,0.4,1	0.5,1,0

Таблица 9.2

Массивы для четвертого задания

Вариант	**Сигнал**
1,6,11, 16,21,26	0,1.4678,2.1005,2.7133,3.1638,3.7576,3.7757,3.5037,3.0221,2.4378,1.847 5,1.6234,1.804,1.9676,2.0654,1.89,1.7957,3.3781,3.8225,4.2308,4.4874, 4.17,5.1476,5.0946,4.8222,4.3938,3.8494,2.5557,2.1918,1.6968,1.1857, 0.7479
2,7,12, 17,22,27	0.1435,0.3109,0.54,0.7522,0.84,0.7522,0.54,0.6269,0.8281,1.2424,1.6565, 1.85,1.6565,1.1893,0.6846,0.316,0.1169,0.0427,0.0925,0.1607,0.2239, 0.25,0.4852,0.7269,1.0761,1.4127,1.5458,1.37,0.9836,0.5662,0.4749, 0.5593,0.8036,1.1193,1.25,1.1193,0.8036,0.4626,0.591,0.8969,1.4208, 1.9789,2.21,1.9789,1.4208,0.8179,0.3775,0.1397
3,8,13, 18,23,28	0.6302,0.8292,1.0088,1.1346,1.18,1.1346,2.3173,2.5509,2.7247,2.7987, 2.45,2.3558,2.0944,1.7217,1.3085,2.3989,1.9465,2.368,2.6635,2.77, 2.6635,2.368,1.9465,1.4794,1.0396,0.9667,1.2719,1.5473,1.7404,1.81, 2.3653,2.3695,2.2721,2.0917,1.8493,2.2413,2.4689,2.6089,2.6345,2.5291, 2.0097,1.7867,1.4687,1.1162,0.7844
4,9,14, 19,24,29	0.0358,0.1073,0.235,0.3761,0.44,0.3761,0.235,0.1073,0.0358,0.0584, 0.1487,0.3258,0.5215,0.61,0.5215,0.3258,0.1487,0.0496,0.2049,0.5779, 1.2658,2.0261,2.37,2.0261,1.2658,0.5779,0.1928,0.047,0,0.3002,0.8998, 1.9708,3.1545,3.69,3.1545,1.9708,0.8998,0.3002,0.2823,0.6267,1.3726, 2.197,2.57,2.197,1.3726,0.7935,0.709,1.1459,1.7525,2.05,1.7525,1.0949, 0.4999,0.1668,0.0407
5,10,15, 20,25,30	0.0632,0.2324,0.5891,1.0295,1.24,1.0295,0.5891,0.2324,0.0632,0.1836, 0.6316,1.6011,2.7979,3.37,2.7979,1.6011,0.6316,0.1717,0.0322,0.1901, 0.699,1.7722,3.0968,3.73,3.1278,1.8865,0.9889,0.6965,0.6456,0.5064, 0.2898,0.1143,0.1233,0.345,0.86,1.5027,1.81,1.5786,1.1392,1.0471, 1.3293,1.5073,1.237,0.7079,0.2792,0.0759,0.0142