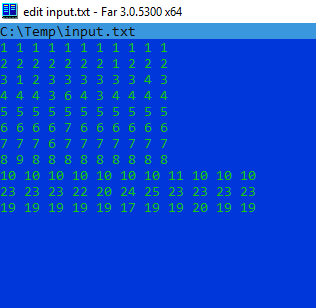
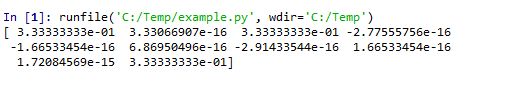
Лабораторная работа № 3

В рамках данной лабораторной работы студентам предлагается ознакомиться с возможностью реального применения на практике линейных моделей. В частности, предлагается рассмотреть модели, имеющиеся в библиотеке scikit learn. По ссылке <https://scikit-learn.org/stable/modules/classes.html#module-sklearn.linear_model> находится описание всех линейных моделей, доступных для использования. Для того, чтобы установить библиотеку, можно воспользоваться инструкцией по ссылке <https://scikit-learn.org/stable/install.html>

Рассмотрим на примере работу с данной библиотекой. Пусть имеется набор объектов и ответов из 11 объектов. Каждый объект описывается 10 параметрами – вещественными числами. На рисунке представлено содержимое файла с описанными данными, в которых каждая строка состоит из 10 чисел, описывающих объект и одного числа-ответа.

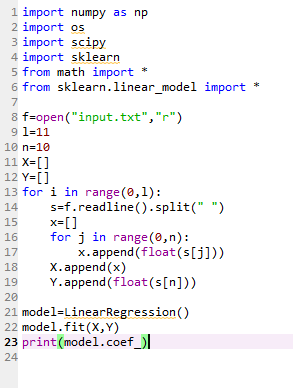


Разработав программу, которая строит модель линейной регрессии и выводит значения коэффициентов, получили следующие значения:



Как можно увидеть, наша модель предположила, что наиболее оптимальным вариантом будет рассмотреть ответ, как среднее значение 1-го, 3-го и 10-го параметров. Для остальных параметров значение коэффициентов настолько малы, что ими можно пренебречь. Таким образом, . Если посмотреть на данные, то можно заметить, что в каждой строке ответ примерно равен каждому из значений параметров, однако, в некоторых из параметров допущены ошибки и их значение отклоняется от ответа. Наша модель нашла подобные отклонения и по предложенным данным выделила три параметра, в которых ошибок не было допущено ни в одном из примеров, совершенно логичным будет предположить, что оптимальный ответ находится при помощи вычисления среднего значения этих параметров.

Ниже представлен код указанной программы, чтобы вы могли ознакомиться и увидеть, насколько просто строятся подобные модели на практике.



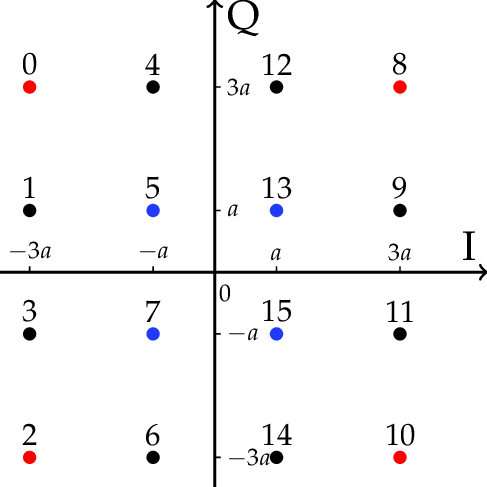
В рамках выполнения лабораторной работы предлагается решить реальную практическую задачу, над решением которой работают крупные телекоммуникационные компании и коллективы учёных. Речь идёт о задаче компенсации нелинейных искажений, возникающих при передаче данных по волоконно-оптическим каналам связи.

Для начала опишем параметры исследуемого канала связи: трёхканальная передача импульсов с использованием двух поляризаций в каждом канале с форматом модуляции 16-QAM (quadature amplitude modulation) на 2000 км. В качестве формы импульсов используется форма RRC (root raised cosine) с коэффициентом сглаживания 0,1 на символьной скорости 32 ГБод и межканальным расстоянием 37,5 ГГц. Чтобы не вдаваться в тонкости работы волоконно-оптических каналов связи, выделим только те аспекты, которые для нас важны:

1. каждый передаваемый импульс задаётся комплексным числом (варианты чисел определяются модуляцией 16-QAM);
2. всего передаётся одновременно 6 потоков импульсов: 3 канала, в каждом канале 2 поляризации.

Все остальные параметры важны исключительно для численного эксперимента, благодаря которому были сформированы файлы с данными. Кроме того, в рамках лабораторной работы не требуется варьировать мощность сигнала, предоставленные данные сформированы с сигналом одной мощности 0 дБм.

Рассмотрим модуляцию 16-QAM. Согласно этой модуляции каждый импульс описывается двумя числами (или, иначе, одним комплексным числом). Если рассмотреть эти числа как координаты, то сетка допустимых значений будет следующей:



В данных, прилагаемых к лабораторной работе значение . Если все значения разделить на указанное значение , то получатся числа: -3,-1,1,3. Ниже будет предложен код, позволяющий легко выполнить эту работу

К лабораторной работе прилагаются 2 файла, rx.bin (приёмник) и tx.bin (передатчик).

Формат данных бинарный:

int32: число каналов

int32: число точек

int32: число поляризаций

int32: флаги

[2 \* double]: комплексные символы по шесть чисел (ch0:x, ch0:y, ch1:x, ch1:y, ch2:x, ch2:y)

Можно использовать, к примеру, такую процедуру загрузки данных:

def load\_field(dataset, tag, index):

    filename = dataset + '/' + tag + '/' + index + '.bin'

    with open(filename, 'rb') as file:

        n\_channels = np.fromfile(file, dtype=np.int32, count=1)[0]

        n\_samples = np.fromfile(file, dtype=np.int32, count=1)[0]

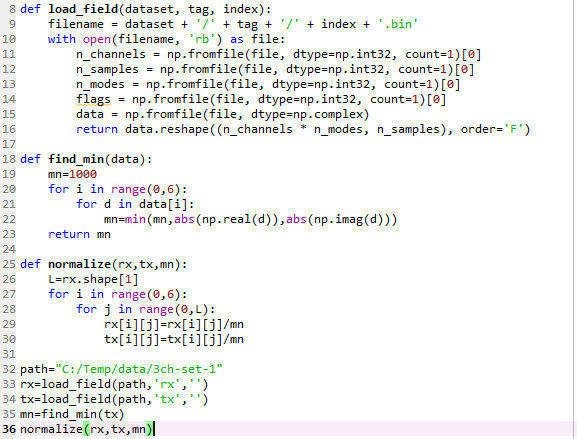
        n\_modes = np.fromfile(file, dtype=np.int32, count=1)[0]

        flags = np.fromfile(file, dtype=np.int32, count=1)[0]

        data = np.fromfile(file, dtype=np.complex)

        return data.reshape((n\_channels \* n\_modes, n\_samples), order='F')

Ниже представлен пример кода, который считывает данные и нормализует их, приводя значения к требуемому диапазону:



В рамках лабораторной работы рекомендуется сосредоточиться исключительно на центральном канале и рассматривать только одну поляризацию. Для тех же, кто заинтересован в реальных подходах к решению данной задачи предлагается ознакомиться с работой: Редюк, А. А., О. С. Сидельников, Е. А. Аверьянов, М. А. Сорокина, М. П. Федорук, and С. К. Турицын. "Метод компенсации нелинейных искажений сигнала в волоконных системах связи на основе теории возмущений и машинного обучения." *Прикладная фотоника* 5, no. 3 (2018): 265-276.

Нелинейные искажения в оптических каналах связи имеют сложную структуру, но в рамках данной лабораторной работы мы будем вести речь исключительно о межимпульсном взаимодействии. Суть его заключается в том, что в процессе перемещения по каналу связи, импульс «расплывается» и начинает влиять на значения последующих и предыдущих импульсов. Чем дальше находятся друг от друга импульсы, тем меньше их влияние. Рассмотрим самую простую линейную модель, будем считать, что влияние соседнего импульса можно учесть, если взять его значение и умножить на комплексный коэффициент, тогда:

где – число рассматриваемых соседей как слева, так и справа, – значение переданного импульса (т.е. до внесения в него искажений), – значение импульса на приёмнике (т.е. с искажением).

Так как речь идёт о комплексных числах, а рассматриваемая в рамках лабораторной работы библиотека умеет работать только с вещественными числами, то необходимо расписать каждое уравнение импульса на 2. Покажем на примере с 1 соседом:

Опишем каждое комплексное число в виде двух составляющих: .

Раскроем скобки:

Сгруппируем по действительной и мнимой частям:

Теперь можно рассмотреть эти части по отдельности и получить тем самым 2 уравнения:

Таким образом, мы можем построить для каждого импульса 2 уравнения и получить максимум (где n – число соседей) уравнений с неизвестными коэффициентами. Необходимо подобрать такое количество уравнений, необходимое для обучения и такое количество соседей, чтобы количество ошибок стало минимальным.

Для определения ошибки необходимо взять значение, полученное на приёмнике, и определить ближайшее к нему исходное на сетке 16-QAM (то есть одно из 16 значений). Чтобы проверить рассматриваемый в лабораторной работе метод, необходимо сначала посчитать количество ошибок без учёта межимпульсного взаимодействия, а потом преобразовать значения на приёмнике при помощи формулы, указанной выше, после чего опять посчитать число ошибок и сравнить полученные результаты.