|  |  |
| --- | --- |
|  | **УТВЕРЖДАЮ**  Профессор кафедры  ИАНИ ННГУ, д.т.н.  \_\_\_\_\_\_\_\_\_\_\_\_\_ Н.В. Старостин  «\_\_\_\_»\_\_\_\_\_\_\_\_\_\_\_\_\_\_2021 г. |

Научно-технический отчёт

**на опытно-конструкторскую работу**

**Разработка нейронной сети специального вида (автоэнкодера) для решения задачи редукции пространства многомерных функций**

**(Шифр ПО «Enc»)**

2021 г.

Содержание

[Введение 3](#_Toc93965299)

[1. Содержательная постановка задачи 4](#_Toc93965300)

[2. Входные данные 5](#_Toc93965301)

[3. Выходные данные 7](#_Toc93965302)

[4. Разработка автоэнкодера 8](#_Toc93965303)

[4.1. Создание обучающей выборки. Задание исходной функции. Функция потерь. Структура автоэнкодера 8](#_Toc93965304)

[4.1.1. Первая модель автоэнкодера 9](#_Toc93965305)

[4.1.2. Вторая модель автоэнкодера 9](#_Toc93965306)

[4.2. Описание структуры автоэнкодера 9](#_Toc93965307)

[4.2.1. Однослойный автоэнкодер 9](#_Toc93965308)

[4.2.2. Двухслойный автоэнкодер 9](#_Toc93965309)

[4.2.3. Вариационный автоэнкодер 9](#_Toc93965310)

[5. Разработка алгоритма выбора гиперпараметров для построения автоэнкодера с наилучшими показателями точности и сжатия 10](#_Toc93965311)

[6. Тестирование 11](#_Toc93965312)

[Заключение 14](#_Toc93965313)

# Введение

Рассматривается проблема разработки нейронной сети специального вида (автоэнкодера) для решения задачи редукции пространства многомерных функций.

В рамках данного проекта проведены следующие работы:

Разработка структуры автоэнкодера, разработка функции генерации данных для обучения автоэнкодера на базе рандомизированных средств с низкой расходимостью, тестирование автоэнкодера на четырех функциях, разработка функции полного перебора гиперпараметров автоэнкодера.

# Содержательная постановка задачи

Для каждой предоставленной функции построить автоэнкодер, обладающий наилучшими характеристиками по сжатию пространства параметров функции и точности.

# Входные данные

В качестве исходных данных выступает описание исходных функций. Для каждой функции должна быть выполнена программная реализация в рамках ПО «Enc». Программная реализация функций приведена на рисунке 1.

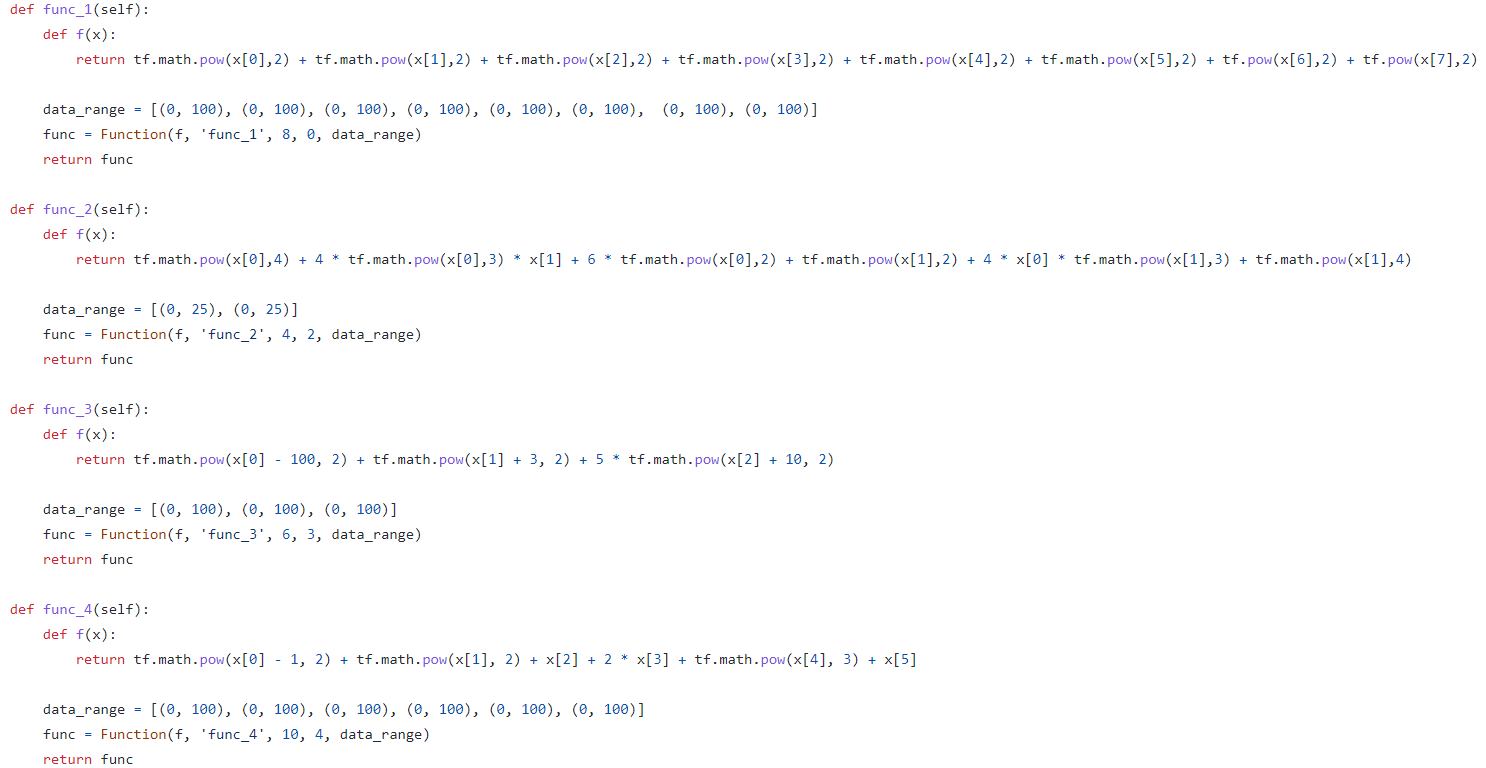


Рисунок 1 – реализация функций

Необходимо определить функцию , задать вектор диапазонов допустимых значений для компонент вектора . Параметрами являются функция, имя функции, размерность пространства параметров, количество незначащих параметров, вектор диапазонов.

В качестве входных данных были даны следующие четыре функции:

1.

Размерность пространства параметров

Количество незначащих переменных

Не спектр, нелинейная

Область определения

Область значений

2.

Размерность пространства параметров

Количество незначащих переменных

Спектр, нелинейная

Область определения

Область значений

3.

Размерность пространства параметров

Количество незначащих переменных

Спектр. нелинейная

Область определения

Область значений

4.

Размерность пространства параметров

Количество незначащих переменных

Спектр. нелинейная

Область определения

Область значений

# Выходные данные

К выходным данным ПО «Enc» относятся:

- описание структуры и параметров обученной нейронной сети, обеспечивающей сжатие параметров исходной функции

Сохранение весов происходит в соответствии со стандартной функцией сохранения весов API Keras. Файл с весами будет сохранен в папке encoderProject-master\Saved models\Weights.

Файл с параметрами нейронной сети будет сохранен в папке encoderProject-master\Saved models\Params. В файле будут записаны выбранные гиперпараметры нейронной сети. Тип автоэнкодера указан в названии файла. Пример файла с параметрами приведен на рисунке 2.

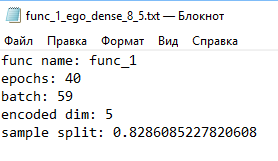


Рисунок 2 – файл, содержащий параметры автоэнкодера.

- показатели нейронной сети по степени сжатия и точности аппроксимации

Для обученной сети ПО «Enc» строит графики отклонений по всем параметрам и указывает среднее отклонение по . Диапазон каждого параметра разбит на 10 равных частей для выявления областей с наибольшей ошибкой. Пример графиков указан на рисунке 3.

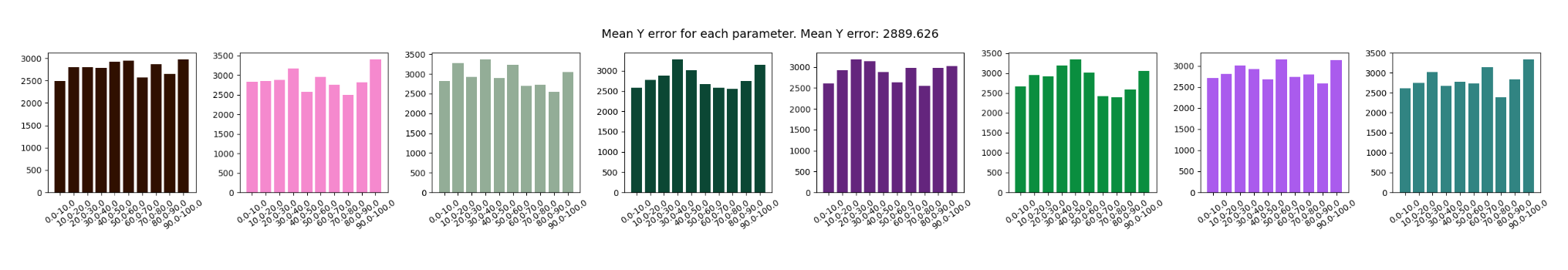


Рисунок 3 – Графики отклонений по параметрам функции

# Разработка автоэнкодера

# Создание обучающей выборки. Задание исходной функции. Функция потерь. Структура автоэнкодера

Для каждой компоненты вектора задается диапазон допустимых значений. Создается векторов , являющих последовательностью Соболя – квазислучайной последовательностью с низкой расходимостью. Данная функция реализована с помощью библиотеки sobol\_seq [https://github.com/naught101/sobol\_seq]. На рисунке 4 продемонстрировано расположение 1000 точек последовательности Соболя на двумерной плоскости, а также рассчитана их дисперсия , при увеличении количества точек дисперсия будет уменьшаться и стремиться к константе. Доказательство, что последовательность Соболя является последовательностью с низкой расходимостью, приведено в *[Соболь, И.М. и Левитан, Ю.Л. (1976). «Производство точек, равномерно распределенных в многомерном кубе» Тех. Доп. 40, Институт прикладной математики АН СССР].*

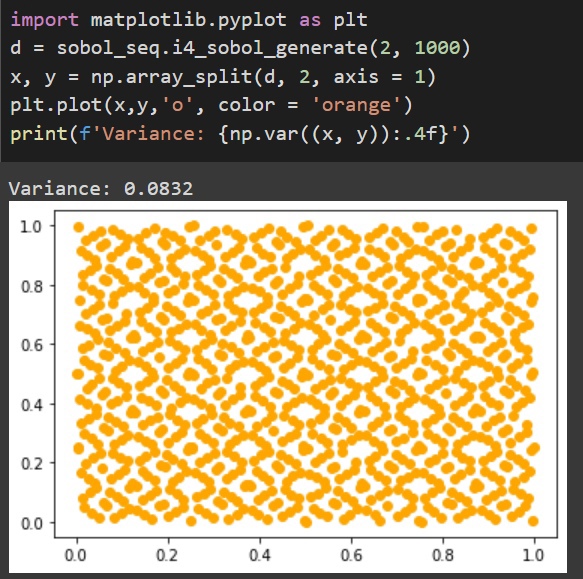


Рисунок 4 – расположение 1000 точек последовательности Соболя.

векторов выбирается для тренировочной выборки, оставшиеся выбираются для валидационной выборки.

# Первая модель автоэнкодера

Для каждого вектора высчитывается соответствующий вектор и конкатенируется с ним. Получившийся вектор подается на вход автоэнкодера. Функцией потерь выбирается средняя разность квадратов между поданным на вход вектором и выходным вектором автоэнкодера.

# Вторая модель автоэнкодера

Вектор подается на вход автоэнкодера. Функцией потерь выбирается средняя абсолютная разность между вектором рассчитанном по входному вектору и вектором , рассчитанном по выходному вектору автоэнкодера.

# Описание структуры автоэнкодера

Общая структура автоэнкодеров. Входной слой, слои кодировщика, внутренний слой – последний слой кодировщика, входной слой декодировщика, слои декодировщика, выходной слой – последний слой декодировщика.

Входной слой, слои кодировщика и внутренний слой образуют структуру кодировщика (Encoder).

Входной слой декодировщика, слои декодировщика и выходной слой образуют структуру декодировщика (Decoder).

# Однослойный автоэнкодер

Слои кодировщика и декодировщика имеют по одному полносвязному слою.

# Двухслойный автоэнкодер

Слои кодировщика и декодировщика имеют по два полносвязных слоя.

# Вариационный автоэнкодер

В слое декодировщика входной вектор сжимается до размера внутреннего слоя полносвязным слоем, получившийся вектор принимается за математическое ожидание и логарифм дисперсии. Создаётся вектор случайных величин с нормальным распределением, математическое ожидание и дисперсию которых получили на предыдущем слое. Слой декодировщика имеет два полносвязных слоя.

# Разработка алгоритма выбора гиперпараметров для построения автоэнкодера с наилучшими показателями точности и сжатия

Для тестирования были выбраны следующие гиперпараметры:

– размер внутреннего слоя

– процент разделения тестовых данных на обучающую выборку и валидационную

– тип автоэнкодера

– количество эпох обучения

количество пакетов, на которое разобьётся выборка

Для построения автоэнкодера с наилучшими показателями точности и сжатия был реализован алгоритм полного перебора гиперпараметров. Данный алгоритм работает очень долго и для ускорения был применен алгоритм эффективной глобальной оптимизации. (smt.applications.EGO)

Алгоритм эффективной глобальной оптимизации основан на Байесовской оптимизации. Более подробно с данным алгоритмом можно ознакомиться в статье [*Jones, D. R., Schonlau, M., & Welch, W. J. (1998). Efficient global optimization of expensive black-box functions. Journal of Global optimization, 13(4), 455-492.*] или в документации к используемой библиотеке [https://smt.readthedocs.io/en/latest/\_src\_docs/applications/ego.html]. В проекте используется алгоритм EGO с количеством начальных точек n + 1, где n – размерность исходного пространства.

Алгоритм полного перебора обучает нейросеть для каждого набора гиперпараметров. Гиперпараметры выбираются следующим образом:

- будут рассмотрены все разработанные типы автоэнкодеров: однослойный, двухслойный и вариационный

перебирается с по с шагом

перебирается как степени двойки с по

перебирается с половины размерности пространства параметров по размерность пространства параметров минус один с шагом один

перебирается с по с шагом

Для нахождения решения генерировались наборы данных размерностью 60000.

# Тестирование

Результаты тестирования автоэнкодеров приведены на рисунках 5-16. Тестирование проводилось алгоритмом эффективной глобальной оптимизации при количестве эпох для всех типов автоэнкодера.

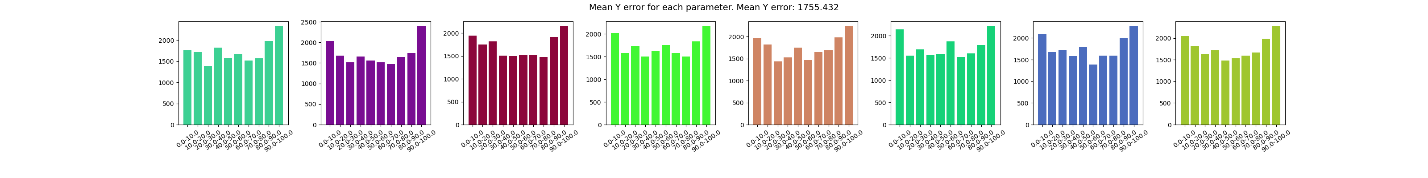


Рисунок 5 – Функция 1, однослойный автоэнкодер

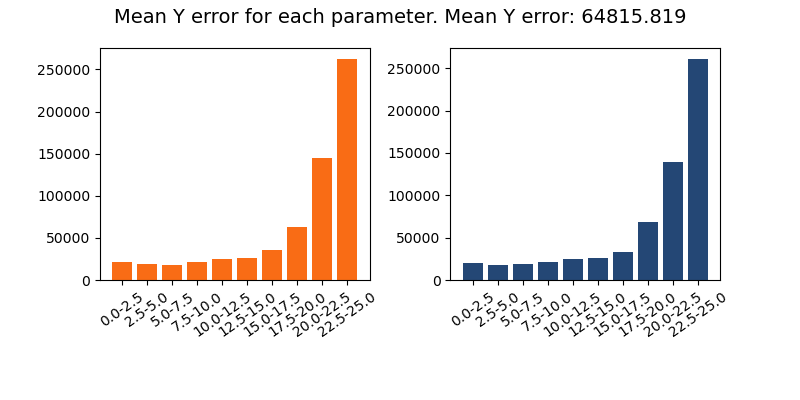


Рисунок 6 – Функция 2, однослойный автоэнкодер

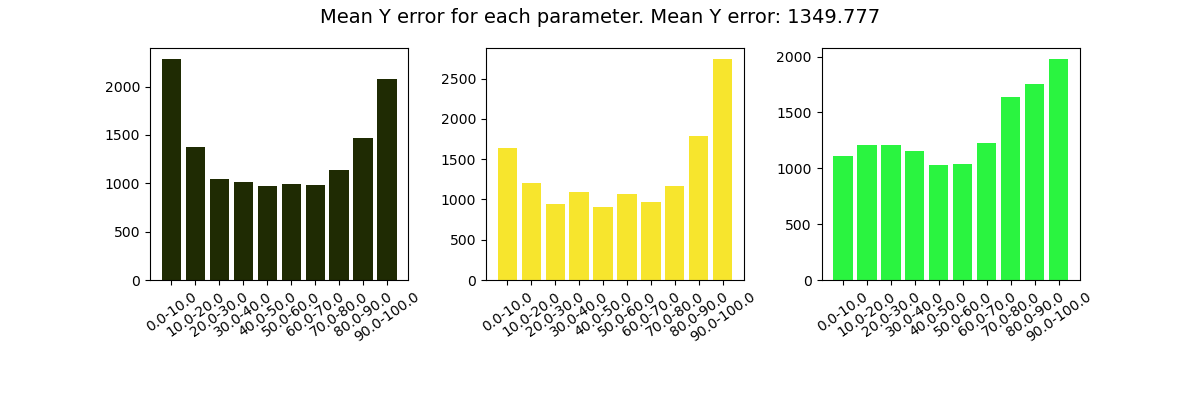


Рисунок 7 – Функция 3, однослойный автоэнкодер

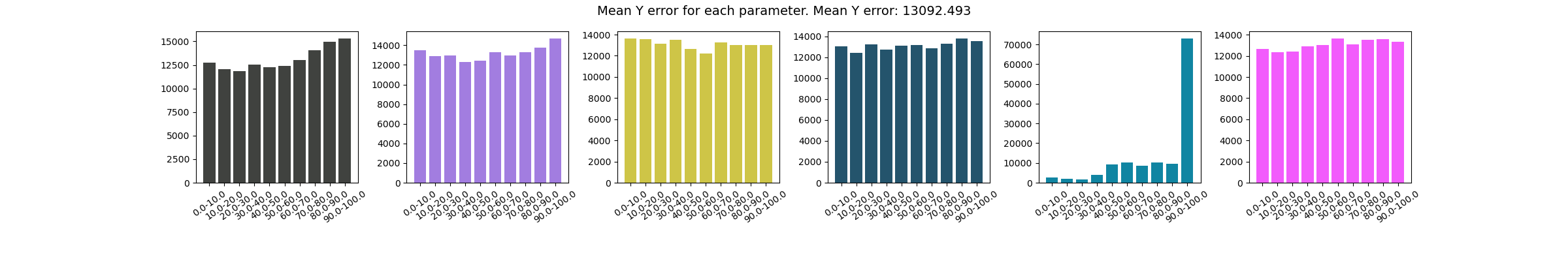


Рисунок 8 – Функция 4, однослойный автоэнкодер

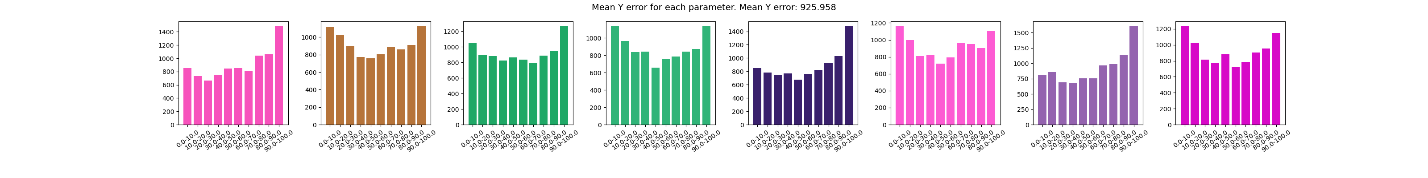


Рисунок 9 – Функция 1, двухслойный автоэнкодер

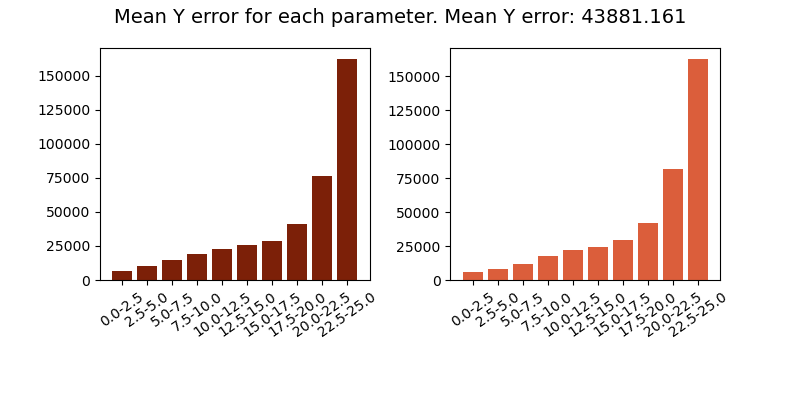


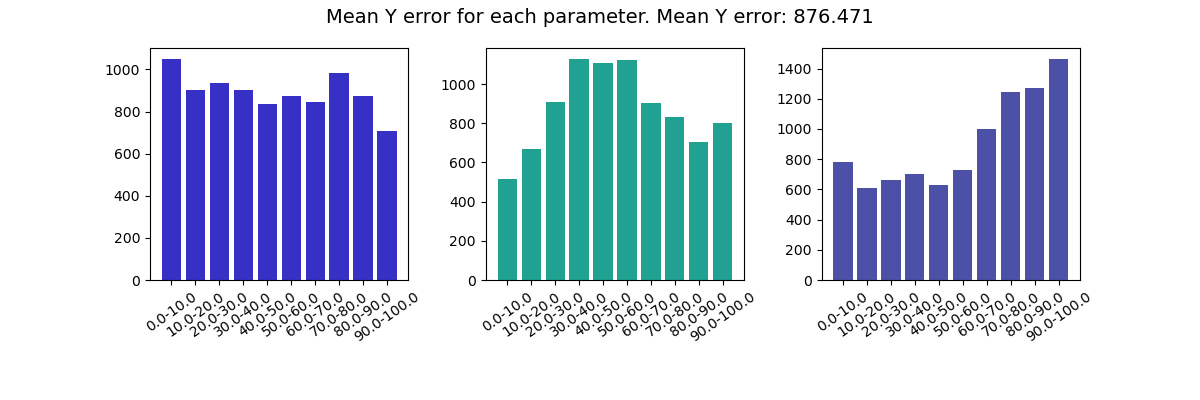
Рисунок 10– Функция 2, двухслойный автоэнкодер 

Рисунок 11 – Функция 3, двухслойный автоэнкодер

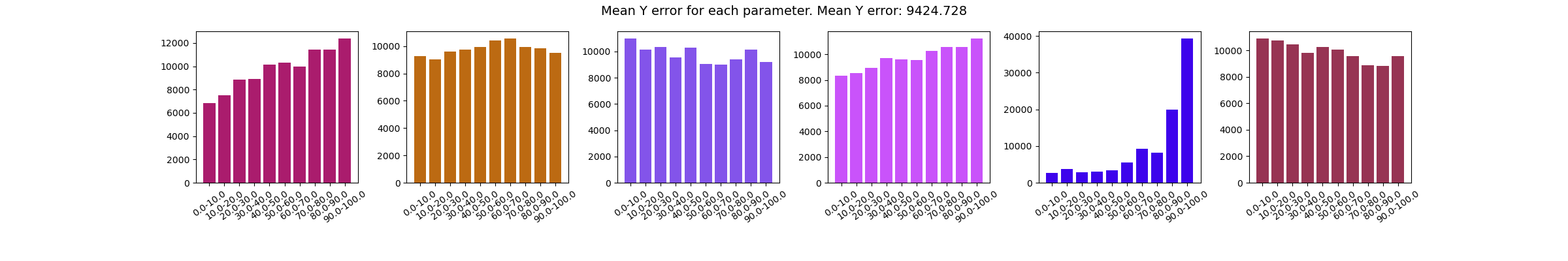


Рисунок 12 – Функция 4, двухслойный автоэнкодер 

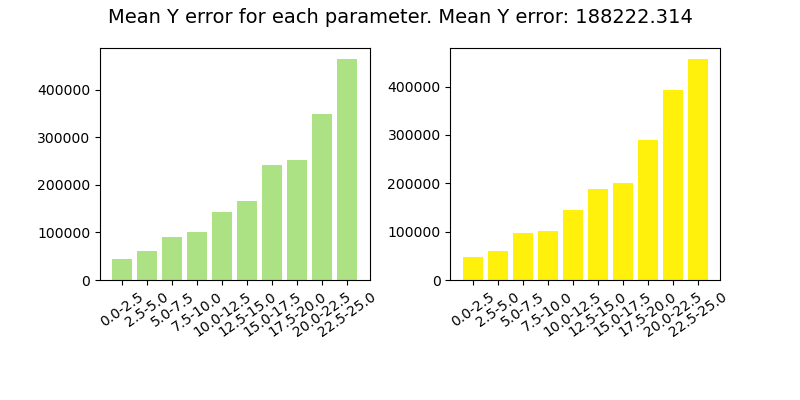
Рисунок 13 – Функция 1, вариационный автоэнкодер 

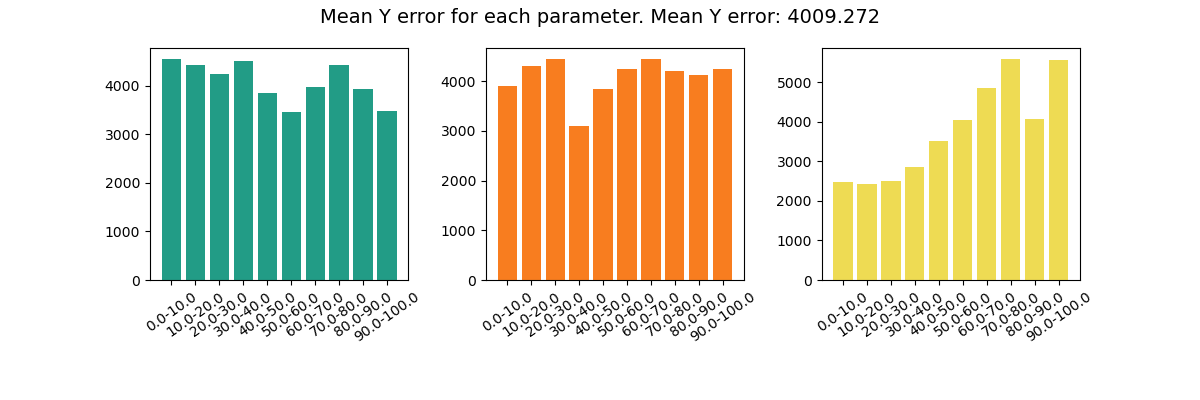
Рисунок 14 – Функция 2, вариационный автоэнкодер 

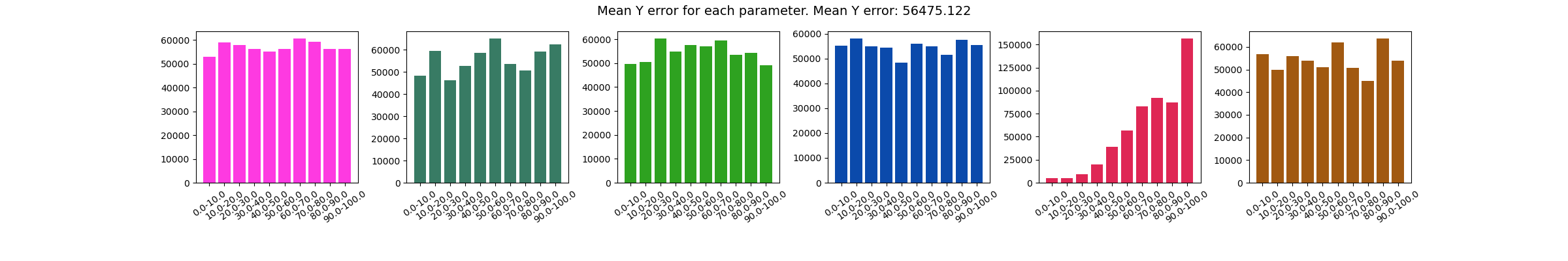
Рисунок 15 – Функция 3, вариационный автоэнкодер 

Рисунок 16 – Функция 4, вариационный автоэнкодер

# Заключение

Была поставлена задача разработки нейронной сети специального вида (автоэнкодера) для решения задачи редукции пространства многомерных функций. В ходе работы все требования к функциональным характеристикам разрабатываемого ПО были выполнены. Из результатов видно, что двухслойная сеть в 1.5 раза лучше по результатам, чем однослойная. Также заметно, что вариационный в среднем лучше сжимает пространство, но по отклонению в текущей реализации не так хорош, как хотелось бы.