**«Санкт-Петербургский государственный электротехнический университет**

**«ЛЭТИ» им. В.И.Ульянова (Ленина)»**

**(СПбГЭТУ «ЛЭТИ»)**

|  |  |  |
| --- | --- | --- |
| **Направление** | 13.03.02–Электроэнергетика и электротехника | |
| **Профиль** | 13.03.02-10- Электропривод и автоматика | |
| **Факультет** | ЭА | |
| **Кафедра** | РАПС | |
| *К защите допустить* |  | |
| Зав. кафедрой |  | Белов М.П. |

ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА

БАКАЛАВРА

Тема: **Разработка адаптивного ПИД регулятора на основе нейронной сети**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Студент(ка) |  |  |  | Дубовцев Н.К. |
|  |  | *подпись* |  |  |
| Руководитель | д.т.н., доцент |  |  | Белов М.П. |
|  | *(Уч. степень, уч. звание)* | *подпись* |  |  |
| Консультанты | к.т.н., доцент |  |  | Козлова Л.П. |
|  | *(Уч. степень, уч. звание)* | *подпись* |  |  |
|  |  |  |  |  |
|  | *(Уч. степень, уч. звание)* | *подпись* |  |  |

Санкт-Петербург

2021

**ЗАДАНИЕ**

**на выпускную КВАЛИФИКАЦИОННУЮ работу**

|  |  |
| --- | --- |
|  | Утверждаю |
|  | Зав. кафедрой РАПС |
|  | \_\_\_\_\_\_\_\_\_\_\_\_ Белов М.П |
|  | «\_\_\_»\_\_\_\_\_\_\_\_\_\_\_\_\_\_20\_\_\_ г. |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Студент(ка) | Дубовцев Н.К | | | |  | Группа | 5404 |
| Тема работы: Разработка адаптивного ПИД регулятора на основе нейронной сети | | | | | | | |
| Место выполнения ВКР: СПбГЭТУ «ЛЭТИ», кафедра РАПС | | | | | | | |
| Исходные данные (технические требования): Разработка нейронных сетей как различных регуляторов. | | | | | | | |
| Содержание ВКР: Анализ существующих методик исполнения нейронных сетей. Разработка и проверка нейронной сети, как регулятора тока. Разработка и проверка нейронной сети, как регулятор напряжения в трехфазной нагрузке. Разработка и проверка нейронной сети, как регулятор скорости при управление АД с вентиляторной нагрузкой. | | | | | | | |
| Перечень отчетных материалов: пояснительная записка, презентация в формате PowerPoint | | | | | | | |
| Дополнительные разделы: | | | | | | | |
|  | | | | | | | |
| Дата выдачи задания | | | Дата представления ВКР к защите | | | | |
| «\_\_\_»\_\_\_\_\_\_\_\_\_\_\_\_\_\_20\_\_\_ г. | | | «\_\_\_»\_\_\_\_\_\_\_\_\_\_\_\_\_\_20\_\_\_ г. | | | | |
|  | | |  | | | | |
| Студент | |  | | Дубовцев Н.К. | | | |
| Руководитель т.н., доцент | |  | | Белов М.П. | | | |
| *(Уч. степень, уч. звание)* | |  | |  | | | |

**календарный план выполнения**

**выпускной квалификационной работы**

|  |  |
| --- | --- |
|  | Утверждаю |
|  | Зав. кафедрой РАПС |
|  | \_\_\_\_\_\_\_\_\_\_\_\_ Белов М.П. |
|  | «\_\_\_»\_\_\_\_\_\_\_\_\_\_\_\_\_\_20\_\_\_ г. |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Студент | Дубовцев Н.К. |  | Группа | 5404 |
| Тема работы: Разработка адаптивного ПИД регулятора на основе нейронной сети | | | | |

|  |  |  |
| --- | --- | --- |
| № п/п | Наименование работ | Срок выполнения |
| 1 | Обзор литературы по теме работы | 05.01 – 10.01 |
| 2 | Обучение нейронной сети на однофазную нагрузку | 13.01 – 15.02 |
| 3 | Обучение нейронной сети на трехфазную нагрузку | 16.02 – 18.04 |
| 4 | Обучение нейронной сети на управление двигателем | 20.04 – 01.05 |
| 5 | Оформление пояснительной записки | 01.05 – 03.06 |
| 6 | Оформление иллюстративного материала | 10.06 – 18.06 |

|  |  |  |
| --- | --- | --- |
| Студент |  | Дубовцев Н.К. |
| Руководитель |  | Белов М.П. |

*(Уч. степень, уч. звание)*

**РЕФЕРАТ**

Пояснительная записка 00 стр., 00 рис., 0 табл., 0 ист., 0прил.

НЕЙРОННЫЕ СЕТИ, АДАПТИВНЫЙ ПИД РЕГУЛЯТОР, АСИНХРОННЫЙ ЭЛЕКТРОВДИГАТЕЛЬ, ЧАСТОТНОЕ УПРАВЛЕНИЕ, РЕГУЛЯТОР СКОРОСТИ, РЕГУЛЯТОР ТОКА, РЕГУЛЯТОР НАПРЯЖЕНИЯ

Объектом исследования являются различные нейронные сети, обученные на работу различных регуляторов.

Целью работы является разработка, настройка, обучение и проверка правильности работы нейронных сетей на имитацию работы различных регуляторов. В качестве регуляторов НС будет обучена на регулятор напряжения и управление однофазной нагрузкой, регулятор тока и управление трехфазной нагрузкой, регулятор скорости и управление асинхронным двигателем с вентиляторной нагрузкой.

В процессе выполнения ВКР был произведен анализ существующих решений для различных архитектур нейронных сетей и методологий обучения, которые лучше всего подходят для управления системами управления. Основываясь на проведенном анализе, была выбрана самая подходящая архитектура и методология обучения нейронный сетей для каждого типа регуляторов и нагрузки. Так же была разработана математическая модель НС в математическом пакете MatLabSimulink и получены графики переходных процессовв, различные режимы работы для различных нагрузок.

**ABSTRACT**

Explanatory note 55 pages,38 figures, 8 tables, 17sources, 1appendix.

NEURAL NETWORKS, ADAPTIVE PID CONTROLLER, ASYNCHRONOUS MOTOR, FREQUENCY CONTROL, SPEED CONTROLLER, CURRENT CONTROLLER, VOLTAGE REGULATOR

The object of the study is various neural networks trained for the operation of various regulators.

The purpose of the work is to develop, configure, train and verify the correct operation of neural networks to simulate the operation of various regulators. As regulators, the NS will be trained in voltage regulator and single-phase load control, current regulator and three-phase load control, speed controller and fan load asynchronous motor control.

During the implementation of the WRC, an analysis was made of existing solutions for various neural network architectures and training methodologies that are best suited for managing control systems. Based on the analysis, the most appropriate architecture and methodology for training neural networks for each type of regulators and load was selected. We also developed a mathematical model of the NS in the MatLabSimulink mathematical package and obtained graphs of transients, different operating modes for different loads

**СОДЕРЖАНИЕ**

[ВВЕДЕНИЕ 9](#_Toc68271726)

[1.Теория о нейронных сетях 10](#_Toc68271727)

[1.1 Математическое представление нейронных сетей 10](#_Toc68271728)

[1.2 Состав нейронной сети 12](#_Toc68271729)

[1.3 Методы обучения нейронных сетей 14](#_Toc68271730)

[2. Специализированные программы для работы с нейросетями 20](#_Toc68271731)

[2.1 Нейронные сети в Matlab 20](#_Toc68271732)

[2.2 Python для работы с нейронными сетями 21](#_Toc68271733)

[3 Создание и обучение нейронной сети 23](#_Toc68271734)

[3.1 Работа с нейронной сетью в Keras 23](#_Toc68271735)

[3.2 Создание модели нейронной сети в Matlab Simulink 24](#_Toc68271736)

[4. Обучение нейронной сети на однофазной модели 28](#_Toc68271737)

[4.1 Построение модели в Matlab 28](#_Toc68271738)

[4.2 Обучение нейронной сети 31](#_Toc68271739)

[5. Обучение нейронной сети на трехфазную нагрузку 37](#_Toc68271740)

[5.1 Создание трех фазной модели в Matlab 37](#_Toc68271741)

[5.2 Выбор параметров R и L для обучения нейронной сети 39](#_Toc68271742)

[5.3 Обучение нейронной сети 40](#_Toc68271743)

[5.4 Рекурентные нейронные сети 45](#_Toc68271744)

[5.5. Обучение нейронной сети с обратной связью 47](#_Toc68271745)

[5.6. Результаты обучения нейронной сети на различные параметры трехфазной нагрузки. 48](#_Toc68271746)

[6. Обучение нейронной сети на различные типы асинхронного двигателя 54](#_Toc68271747)

[6.1. Создание системы управления асинхронным двигателем в Matlab Simulink 54](#_Toc68271748)

[6.2. Обучение нейронной сети. 55](#_Toc68271749)

[6.3. Результаты обучения нейронной сети 59](#_Toc68271750)

**ОПРЕДЕЛЕНИЯ, ОБОЗНАЧЕНИЯ И СОКРАЩЕНИЯ**

НС – Нейронная сеть

МОР – Метод обратного распространения ошибки

ПИД -

ШИМ –

РНС – рекуррентная нейронная сеть

ВВЕДЕНИЕ

В современных технических устройствах, комплексов и процессов, важнейшей и неотъемлемой частью являются системы автоматического управления. В современном мире человек все больше и больше зависит от нормального функционирования технических систем, а значит и от систем управления ими.

С каждым годом для решения всё более сложных задач разрабатываются новые алгоритмы и методы и усовершенствуются классические подходы для решения задач. Одним из новых и перспективных методов для решения задач, возникающих в теории автоматического управления, является использование искусственных нейронных сетей. После того как в 2012 году группа ученых под руководительством Джефри Хинтона победила в ежегодном соревновании ImageNet, суть соревнования заключалась в решение непростой задачи, а именно в классификации цветных изображение высокого разрешения на 1000 категорий, с помощью нейронной сети глубокого обучения. Это дало огромный толчок для изучения нейронных сетей и спонсирования этого направления.

Преимуществами НС перед традиционными системами управления

• НС могут обучаться любым функциям, важен только объем предоставленных данных и выбор правильной нейронной модели. Таким образом НС позволяют избежать использования сложного математического аппарата;

• НС обеспечивают возможность осуществлять управление в условиях существенных нелинейностей;

• Высокая степень параллельности НС обеспечивает высокую производительность вычислений (в случае аппаратной реализации НС);

• Архитектура параллельной обработки позволяет НС функционировать даже при повреждении отдельных элементов сети (в случае аппаратной реализации НС).

Цели использования нейронных сетевых моделей вытекают из их преимуществ перед традиционными системами управления:

* моделирование линейных и нелинейных процессов;
* моделирование сложных для понимания процессов;
* имитация процессов, которые трудно смоделировать с помощью первопринципных уравнений;
* моделирование процессов, основанных на не поддающихся измерению параметрах;
* сокращение времени разработки модели для сложных процессов.

Контроллер, использующий нейронные сети для управления нелинейным объектом, предсказывает поведения объекта управления на различные типы возмущающих воздействий и при помощи различных входных воздействий производит управление данным объектом, что бы он, в свою очередь, правильно отрабатывал поставленную перед ним задачу. Построение модели объекта называется идентификацией системы. Процесс построения нейронной сети состоит из следующих этапов:

1. сбор и нормализация данных для обучения;
2. выбор архитектуры сети и алгоритма обучения;
3. обучение;
4. проверка адекватности обучения.

Объектом данной выпускной квалификационной работы является нейронная сеть, а предметом исследования использование нейронных сетей вместо ПИД регуляторов.

Целью данной выпускной квалификационной работы является разработка и обучение нейронной сети выполнять работу ПИД регулятора, для управления током, напряжением и скоростью вращения двигателя.

Структура данной выпускной квалификационной работы обусловлена предметом, целью и задачами исследования. В первой главе рассматривается вопрос о применение нейронных сетей в современных системах управления, а так же рассматриваются основные принципы выбора архитектуры нейронных сетей методик их обучения. Во второй главе рассматриваются различные специализированные программы для работы с нейронными сетями и производится выбор программы для дальнейшей разработки нейронных сетей. Третья глава посвящена разработки нейронной сети в выбранном программном обеспечение, построение модели нейронное сети в Matlab Simulink и проверка правильности построения модели. В четвертой главе производится разработка модели с однофазной нагрузкой и обучением на эту модель работы нейронной сети. Главным аспектом данной главы является подбор правильных данных для обучения нейронной сети, для того чтобы на полученном опыте дальше можно было обучать нейронные сети на другие типы нагрузки. Пятая глава посвящена разработки системы с трехфазной нагрузкой и обучением на ее работу нейронной сети. Основной идеей данной главы является разработка и обучение нейронной сети на различные типы нагрузки, то есть разработка адаптивной нейронной сети, которая имеет возможность сама подстраиваться под различные типы нагрузки. В шестой главе рассматривается обучение и работа нейронной сети, которая управляет различными типами асинхронных двигателей, находящимися под вентиляторной нагрузкой. В седьмой главе БЖД. В заключении подводятся итоги исследования, формируются окончательные выводы по рассматриваемой теме.

1.Теория о нейронных сетях

## 1.1 Применение нейронных сетей в современных системах управления

Нейронные сети все больше и больше находят применение в современных промышленных агрегатах. Чаще всего их применяют, когда традиционные системы управления недостаточно эффективны. Для управления комплексом электроприводов использование нейронных сетей целесообразно, когда комплекс имеет широкое диапазон изменения механических параметров и различные режимы работы, когда применение традиционные системы управления с линейными регуляторами не могут достичь необходимой точности или быстродействия.

Нейронная сеть способна выполнять множество различных функций, например:

* Управление электроприводами в составе комплекса [1].
* Диагностика работающего оборудования [2].
* Прогнозирование различных ситуаций при работе агрегатов [3].
* Мониторинг и контроль работы технологических процессов с минимальным участием человека.

Характерными признаками для нейронных сетей являются обобщенность накопленных данных и обучаемость. Обученная на ограниченном множестве данных сеть в дальнейшем способна обобщать полученную информацию и обрабатывать данные, не использовавшиеся при ее обучении. Нейроуправление динамическими объектами является комбинацией методов автоматического управления и методов искусственного интеллекта. Нейронные сети обладают рядом особенностей, которые позволяют применять их для моделирования и управления нелинейными динамическими объектами.

В настоящее время разработано, исследовано и используется на практике множество типов нейронных сетей, использующих различные обучающие алгоритмы, характеризующихся своими сильными и слабыми сторонами. Наиболее распространенными архитектурами нейронных сетей являются сети прямого распространения, рекуррентные нейронные сети, самоорганизующиеся карты или сеть Кохонена, сверточные нейронные сети, радиально-базисные функции. Количество архитектур нейронных сетей насчитывается десятками, при этом для разных типов задач разные типы нейронных сетей являются наиболее эффективными. Нейронные сети позволяют создать модель объекта, точно передающую его динамику, при этом, не требуя дополнительных знаний о структуре и параметрах объекта. Необходимыми данными являются только значения входных и выходных сигналов, таким образом, объект представляется в виде черного ящика.

Нейронные сети и нейроконтроллеры могут применяться для управления различными объектами, в том числе и электроприводами различных механизмов, при этом нейроконтроллер осуществляет выработку управляющего сигнала. Нейронная сеть на первых итерациях обучающего процесса может выдавать значения управляющего сигнала, недопустимые, например, для электропривода. Поэтому настройку нейроконтроллера следует производить с использованием модели объекта.

## 1.2 Математическое представление нейронных сетей

. Нейронные сети – это попытка математиков различными законами и формулами описать работу биологической нейронной сети. Нейронная сеть представляет из себя последовательно соединенные между собой нейроны. Основной задаче нейронных сетей является получение определенных результатов, по которым будет выполняться определенное предположение, по набору значений, подающихся на вход нейронной сети. Предположения могут быть различными от того какой цвет показывают НС, до того, как правильно ответить на то или иное предложение.

Базовые компоненты, из которых строятся все нейронные сети — это синопсис, нейрон и функции активации. Все эти компоненты отвечают за важные функции, без которых нейронные сети не смогут корректно функционировать.

Синопсис — это связи между нейронами. Они обладают всего одним параметром, который принято называть вес синопсиса или просто вес. Вес необходим для изменения входной информации передающейся от нейрона к нейрону. Этот параметр показывает насколько входная информация будет влиять на выход нейрона. Ниже представлена формула по которой происходит расчет параметров, входящих в нейрон. 



где - вход нейрона;

 - вес синопсиса.

Нейрон — это определенный математический оператор, который складывает значения входов. После к полученному значению прибавляется значение “смещения, если и формируется значения выхода. Рисунок нейрона представлен ниже.

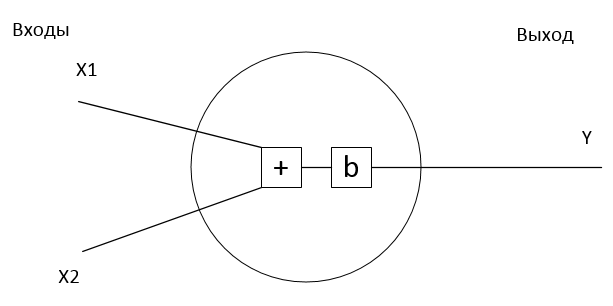


Рисунок 1 – Визуализация нейрона

Далее значение выхода подается на функцию активации для того, чтобы упорядочить результат работы нейрона к определенному диапазону значений, чтобы в дальнейшем передавать упорядоченный ответ следующим нейронам, для их более правильной работы и обучения. В таблице 1 представлены наиболее часто используемые функции активации.

Таблица 1 Функции Активации

|  |  |  |  |
| --- | --- | --- | --- |
| Название | Функция | Диапазон значений | График функции |
| Линейная | F(x)=x | (;) |  |
| Сигмоид |  | [0;1] | Sigmoid function - Wikipedia |
| Гиперболический тангенс |  | [-1;1] |  |
| Линейный выпрямитель |  | [0;) | How ReLU works in convolutional neural network - knowledge Transfer |

Функции активации являются важным инструментом в нейронных сетях на ровне с нейроном и синопсисам, влияющими на нейронную сеть, на ее работу и на ее обучение. Эти функции от части определяют какие нейроны будут активированы и какая информация будет передаваться дальше последующим нейронам.

Линейную функцию активации использую редко. В основном ее используют если необходимо передать данные с выходов нейронов дальше без изменений. Чаще всего в качестве функции активации применяют сигмоиду, её также называют логической функцией. Так как регуляторы могут выдавать и отрицательные и положительные значения поэтому функции Сигмоид в работе не подойдет для работы с числами в данном диапазоне лучше всего использовать гиперболический тангенс.

## 1.3 Состав нейронной сети

Для того чтобы правильно описать состав нейронной сети необходимо ввести понятие слоя. Слой – это набор, не связанных между собой нейронов, на который подаются значения либо от предыдущих нейронов, либо из в не сети. Ниже на рисунке представлены, какие слои входят в состав нейронной сети.

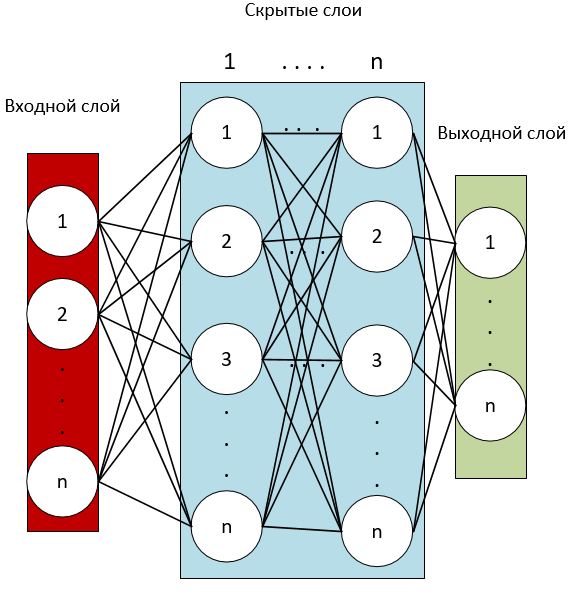


Рисунок 2 – Слои в нейронной сети

Нейронные сети состоят из трех основных слоев:

Входной слой – служит для подачи значений в нейронную сеть;

Скрытые слой – основной вычислительный слой, в котором происходит основная работа нейронной сети. Скрытый слой может состоять из одного или нескольких слоев;

Выходной слой – предназначен для вывода результатов работы нейронной сети.

Выходы нейронов каждого слоя связаны синопсисами с входом каждого нейрона следующего слоя.

Нейросеть может иметь любое количество слоев, и в них может быть любое количество нейронов. Эти параметры влияют на работу нейронной сети и на ее обучения. Не существует формул по которым от количества входных параметров или от типа передаваемых данных или от их диапазона можно было бы рассчитать количество слоев и количество нейронов в каждом слое для корректной работы сети. Именно поэтому значения этих параметров подбираются эмпирическим путем.

## 1.4 Методы обучения нейронных сетей

Выделяют четыре основных метода обучения нейронных сетей, каждый из них имеет свои плюсы и свои минусы.

Машинное обучение с учителем.

Этот тип обучения подразумевает, что во время обучения вместе с входными параметрами нейронной сети в данных для обучения, как их еще называют дата сеты, присутствуют значения выходных параметров сети. Во время обучения нейронная сеть сравнивает полученные данные на своем выходном слое с данными, которые у нее должны были получиться при данных значениях входного слоя. И по методу обратного распространения ошибки меняет весовые коэффициенты своих синопсисов. После этого данные ещё раз подаются, и происходит сравнение выходов и так до тех пор, пока значение разницы не достигнет нуля или не будет близкой к нулю. Тогда можно считать, что сеть обучилась.

Так же для дальнейшего понимания методов обучения необходимо рассказать про метод обратного распространения ошибки. Для подсчёта разницы между выходом и значением, к которому она стремится, используют следующую формулу.

1)

2),

где  - это разница для выходного слоя;

 – разница для скрытых слоев;

 - это идеальный выход;

- это то значение которое рассчитала НС;

 – это производная функции активации слоя;

 – вес синапса;

 – дельта ошибки предыдущего слоя.

Так как нейроны, содержащиеся на выходном слое, не имеют исходящих синапсов, то для подсчета дельты необходимо пользоваться первой формулой. Для нейронов, содержащихся во входном и скрытом слое, второй.

Так как при расчете МОР необходимо использовать производную функции активации, то функций активации необходимо выбирать такие, что бы их можно было продифференцировать. Так же для упрощения расчетов для некоторых функций активации формулу производной можно заменить на упрощенную:



Далее необходимо найти градиент для каждого входящего в нейрон синапса, чтобы в дальнейшем правильно изменить его вес. Значение градиента будем искать по следующей формуле.



, где А - начало синапса;

B - Конец синапса.

Теперь по формуле, представленной ниже мы можем рассчитать вес, на который нам необходимо изменить вес синапса.



где  – скорость обучения;

 – момент обучения;

 – это вес синапса на предыдущем шаге итерации.

Далее необходимо полученное значение сложить с весом синопсиса.



Более подробно о МОР описано в книге [4].

Чаще всего обучение с учителем используется для решения двух типовых задач это задачи классификации и задачи регрессии.

Задачи классификации чаше всего применяются, когда нужно определить тот или иной предмет на фото. Фотография разбивается на пикселе и каждый пиксель по отдельности подаётся на вход нейронной сети, таким подходом можно добиться от нейронной сети чёткого определения цвета пиксели, и по расположению пикселей обучить на распознавание предметов. Каждое изображение связанно с определённым наборов нейронов в выходном слое. Например, необходимо различить автотранспорт, изображённый на фото. На вход нейронной сети будут подаваться фотографии с изображением поезда, автомобиля или мотоцикла. Каждому изображению будет дана метка соответствующему определённым значениям выходного слоя. Например, когда на вход подаётся изображения поезда, то на выходном слое нейроны должны выдать результат [1;0;0]. При подаче на вход изображения автомобиля на выходе должны быть значения [0;1;0], а при загрузке изображения мотоцикла [0;0;1] соответственно.

После обучения нейронной сети и при загрузке на её входной слой изображения, не участвовавшего в обучения, автомобиля нейросеть должна выдать результат похожим на следующий: [0.05;0.99;0.05], что говорит о том, что нейросеть на 99% уверена, что на изображении автомобиль.

Задача регрессии связана с непрерывной подачей данных на вход нейросети. Одним из примеров регрессии является линейная регрессия. Нейросеть при получение конкретного значения x должна вычислить ожидаемое значение.

Обучение без учителя.

Иногда собрать обучающую выборку для нейросети, где каждому значению входа соответствует определённое значение выходов достаточно сложно, потому что это может занять много времени или потратить огромное количество ресурсов. Именно поэтому используются метод обучения без учителя. Перед нейросетью ставится задача самостоятельно найти корреляции в данных, извлекая полезные признаки и анализируя их.

Одним из примеров такого типа обучения является кластеризация. Нейросеть находит общие признаки у данных и группирует их вместе. На рисунке ниже приведен пример кластеризации.

По большому количеству данных нейронная сеть разделила в разные группы обувь и подарки.

Обучение с частичным привлечением учителя.

Этот метод признано считать самым лучшим, когда обучение происходит на большой выборке данных, потому что он включает в себя и обучение с учителем, и обучение без учителя. Обучающая выборка при таком обучение имеет значения входных данных с выходным результатом и без результатов. Такой метод обучения применяют, если из набора данных трудно извлечь общие признаки или набор данных настолько велик, что дать значения выхода каждому набору входных параметров трудоёмкая задача.

Подобный метод машинного обучения получил широкое применение в медицине, в анализе рентгенов и МРТ.

Обучение с подкреплением.

В отличие от машинного обучения с учителем, при данном типе обучения на вход нейронной сети не подаётся обучающая выборка. Вместо этого сеть обучается “методом проб и ошибок”. При таком подходе обучения нейросет пытается найти оптимальный путь для достижения цели. Если нейросеть ошибается, то она получает “штрафные балы” и начинает обработку данных с начала. Если нейросеть предпринимает действие, которое приведёт её к искомо верному результату, то она получает “награду” и продолжает работу. В конечном итоге нейросеть обучается предсказывать свое следующие действие для получения максимально “награды”.

Каноничным примером обучения с подкреплением является обучение нейросети управлять машинкой, чтобы пройти заданный маршрут как можно быстрее.

Для осуществления поставленной задачи из машинки испускается от шести до восьми лучей, как показано на рисунке 4, которые являются в свою очередь входами для нейронной сети. Эти лучи передают в нейронную сеть расстояние, до какого-либо препятствия. По полученным данным НС корректирует движение машины и избегает столкновение с объектом. По схожему принципу работает автопилот в автомобилях компании Tesla, только в данном случае лучи передают не только расстояние до предмета, а также способны различать этот предмет и в зависимости от его типа принимать то или иное решение.

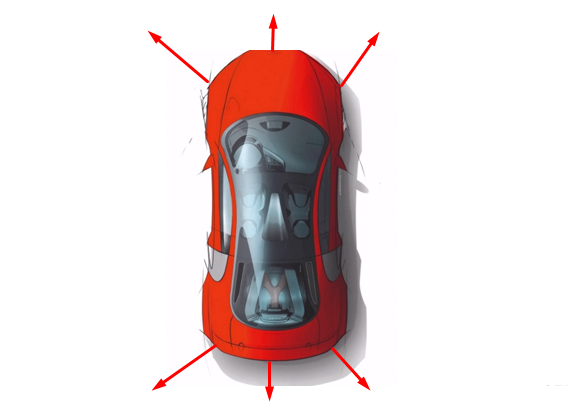


Рисунок 3 – Обучение нейронной сети с подкреплением

Во время обучения машина методом проб и ошибок учится не врезается в препятствия и проходить трассу как можно быстрее.

Такое метод обучения чаще всего используется в компьютерных играх для управления искусственным интеллектом, и в обучение роботов, которые управляют автономными транспортными средствами без участия человека.

Для решения задачи поставленной в рамках данной диссертации больше всего подойдет метод обучения с учителем по нескольким причинам:

Во-первых, в обучающей выборке будет заранее известно какое значение по входным параметрам должна получить нейросеть, для того чтобы с имитировать работу ПИД регулятора.

Во-вторых, обучающая выборка будет получена в ходе работы реального регулятора на объект. Поэтому в разметке обучающих данных не будет проблем.

В-третьих, данный тип обучения решает проблемы регрессии и постоянного потока данных, что будет определяющим фактором при имитации работы ПИД регулятора.

2. Специализированные программы для работы с нейронными сетями

Из-за большой популярности нейронных сетей существует огромное количество программ, позволяющих работать с ними. Все эти программы имеют свои плюсы и свои минусы.

## 2.1 Нейронные сети в Matlab

Matlab – это мощный пакет прикладных программ для решения задач технических вычислений. Он имеет огромное количество библиотек позволяющих смоделировать любую ситуацию. Одна из таких библиотек под названием “Deep Learning Toolbox” позволяет работать с нейросетями. В своем арсенале она имеет уже готовые примеры контроллеров на основе нейросетей, которые можно обучить и настроить на работу.

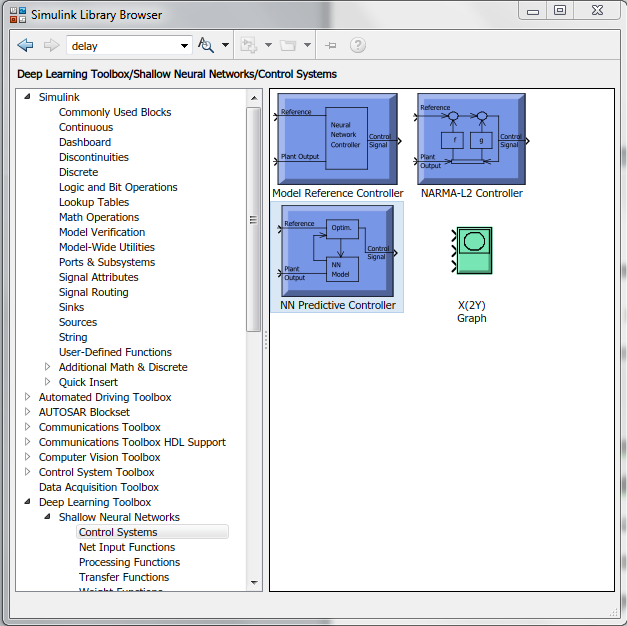


Рисунок 4 – Библиотека Deep Learning Toolbox

Так же в эту библиотеку входят блоки, с помощью которых можно составить свою нейросеть, смоделировать функции активации

Так же в Matlab существует приложение **Deep Network Designer,** которое визуализирует создание глубоких нейронных сетей.

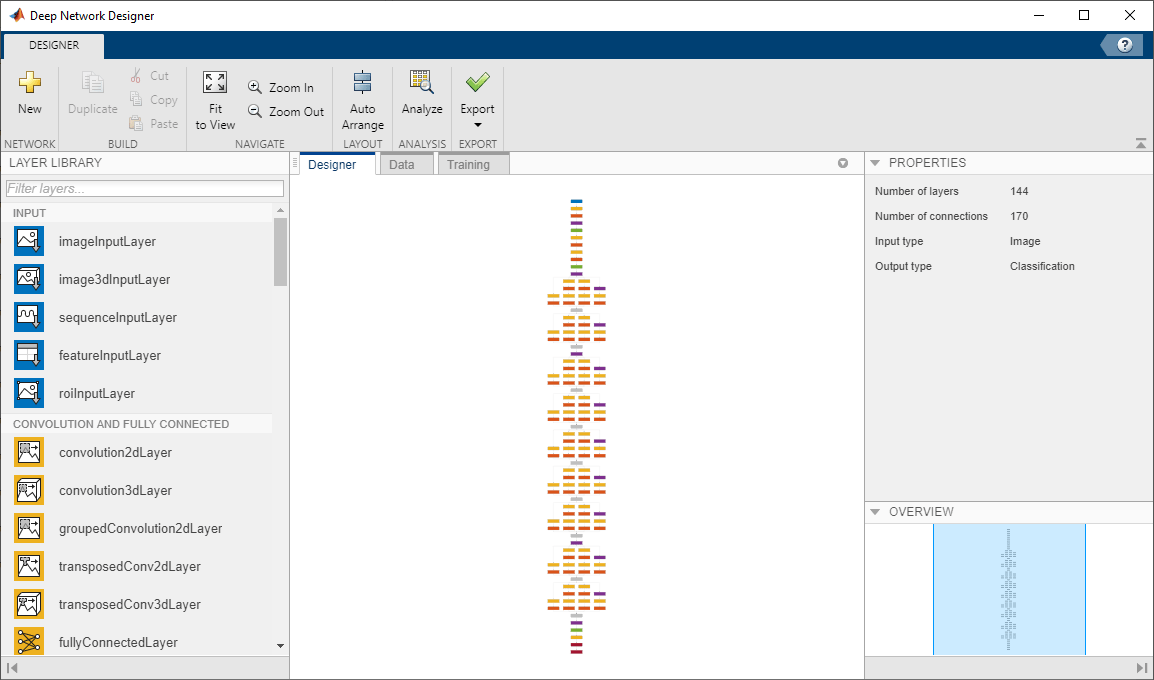


Рисунок 5 – Интерфейс приложения Deep Network Designer

Приложение интуитивно понятно из доступных блоков строится любая топология сети. Все блоки можно подробно настроить под свои нужды. Так же matlab предоставляет большой архив с уже готовыми нейросетями, что помогает пользователю научится правильно выбирать и строить топологию сетей и смотреть уже готовые решения для некоторых проблем.

## 2.2 Python для работы с нейронными сетями

Большую популярность для работы с нейросетями получил язык программирования Python. Python – это высокоуровневый язык программирования с динамической типизации и автоматически управлением памяти. Основной идеей языка является читаемость кода и повышение производительности разработчика, за счет того что многий функционал язык делает за разработчика, например работа с памятью. Python является интерпретируемым языком, что уменьшает скорость работы написанных на нем программ и большим потреблением памяти. Из-за своей популярности и открытости кода python имеет огромное количество библиотек, в том числе и для работы с нейросетями. Одной из таких библиотек является Keras.

Keras является открытой нейросетевой библиотекой, которая представляет из себя надстройку на фреймворками TensorFlow и Theano. Основными преимуществами этой библиотеки перед другими является ее компактность, модульность и предоставление интуитивно более понятных функций, которые помогают, без особых усилий создавать нейронные сети.

В данной работе работа с нейросетями будет вестись с помощью языка программирования Python его библиотеки Keras,потому что “Deep Learning Toolbox” хоть и имеет уже готовые котроллеры на основе нейросетей, но они сложно настраиваемые и не позволят изменить топологию сети и имеют сложный интерфейс и методы работы. Приложение **Deep Network Designer** удобно в использование, имеет обширные возможности для построения нейросетей, но оно в основном заточено на глубокие нейросети и работу с изображениями.

3 Создание и обучение нейронной сети

## 3.1 Работа с нейронной сетью в Keras

Для создания и обучения нейронной сети как говорилось в предыдущей главе будет использоваться язык Python и его библиотека Keras.

В первую очередь необходимо выбрать архитектуру нейронной сети и передать это значение в модель. Выполняется это с помощью следующей команды.

model = keras.Sequential()

Sequential – означает, что будет использоваться линейный стек слоев.

Дальше необходимо добавить слои в модель с помощью функции add().

model.add(Dense(10, input\_shape=(2,), activation='relu',use\_bias=False))

Первое число говорит о том, сколько нейронов будет находиться в слое. Так как вначале создается входной слой то необходимо в параметр input\_shape=(2,) передать число, которое скажет программе какое количество нейронов будут входными. Параметр activation отвечает за то, какая функция активации будет использоваться в данном слое. Функции активации описаны в первой таблице. Параметр use\_bias говорит системе хочет ли разработчик использовать смещение или нет.

С помощью функции add можно добавить в модель сколько угодно слоев.

После того как все слои добавлены в модель необходимо данную модель скомпилировать. Выполняется это с помощью функции

model.compile (loss='mean\_squared\_error', optimizer=keras.optimizers.Adam(0.1)).

Параметр loss - . Это значение, которое модель пытается минимизировать. В данном примере выбрана функция средне квадратичной ошибки

После компиляции модель готова к обучению. Для того чтобы приступить к обучению необходимо воспользоваться функцией fit.

history = model.fit(train, rezults, epochs=10)

В переменную History будет записана история обучения сети.Ее ошибки и точность после каждой эпохи. С помощью переменной history можно визуализировать обучение сети и получить графики по ошибкам и точности сети за весь период обучения.

Переменная train является массивом векторов, которые будут подаваться в сеть для ее обучения. Размер входного вектора должен быть равен количеству входных нейронов.

Переменная results это тоже массив векторов только на основание данных в этом массиве и значений на выходе нейронной сети будет происходить обучение модели и изменение весов синопсисов для уменьшения ошибки.

Переменная epochs принимает количество эпох. Количество эпох отвечает за то сколько раз будет проходить обучения нейросети

Более подробно функционал библиотеке Keras описан в книге [5], написанной самим автором данной библиотеки.

С помощью функции predict можно в уже обученную нейронную сеть подать значения на вход и узнать какой результат выдаст нейросеть. Весь программный код Приложение А.

## 3.2 Создание модели нейронной сети в Matlab Simulink

Такие характеристики как: значения подающиеся на входных слой, веса синопсисов, значения на выходном слое, можно представить в виде матриц. Поэтому работа всей сети сводится к умножению матрицы входа на матрицу весов синопсиса. Полученную матрицу необходимо провести через функцию активации.

где,

 – Матрица весов синопсисов;

 - Матрица входных сигналов;

 – результирующая матрица.

На рисунке ниже представлена нейросеть собранная с помощью библиотечных элементов в matlab Simulink.

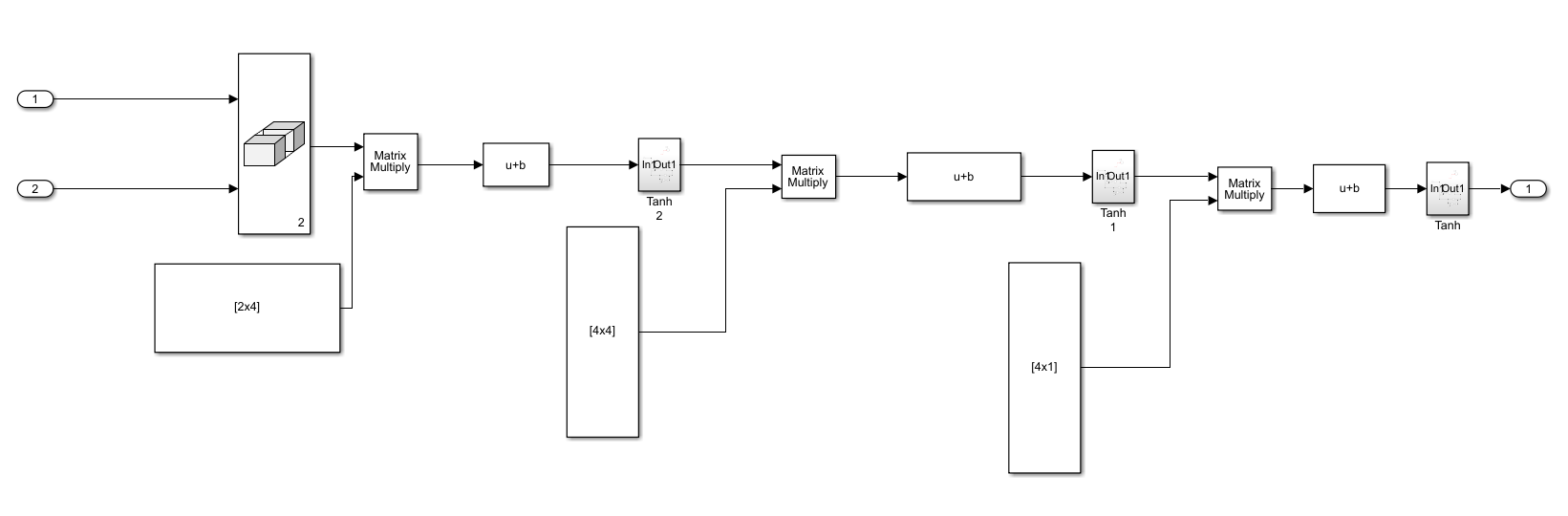


Рисунок 6 – Нейронная сеть, созданная в Matlab Simulink

В модель подаются значения. Далее они преобразуются в матрицу размером 1х2 и умножаются на весовые коэффициенты синопсисов. Полученная матрица пропускается через функцию активации и умножается на веса синопсисов следующего слоя. И так продолжается до тех пор, пока не получится результат на выходе.

Для того что бы проверить правильность собранной модели в Python имеющий два входа, два скрытых слоя по четыре нейрона в каждом и одним выходом, была обучена на логическую операцию «исключающее или» представленную на рисунке ниже.

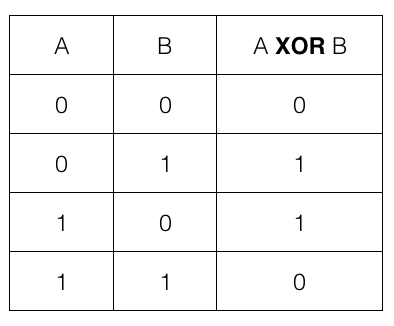


Рисунок 7 – Исключающее или

На графике ниже можно увидеть, как ошибка между выходом нейронной сети и обучающим значением менялась с количеством пройденных эпох.

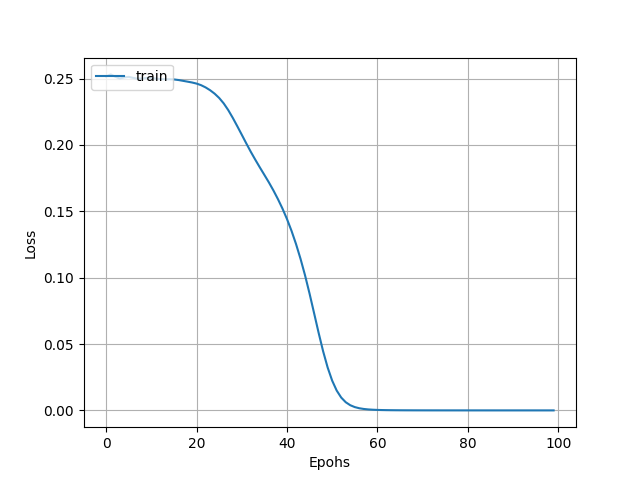


Рисунок 8 – График изменения ошибки при обучении

Значение ошибки на сотой эпохе составляет 0.000005. По графику видно, что модель обучилась к 60 эпохе, а дальше уже не обучалась. Если модель учить слишком много эпох, то возможен момент переобучения, когда ошибка начнет беспричинно расти.

Воспользовавшись функцией Predict и подав на вход нейронной сети значения 0 и 1 функция, показала значение 0.9959903, что по рисунку 8 очень близко к верному значению.

Теперь необходимо выгрузить значения весов и смещений из Python и занести в модель matlab. На рисунке ниже представлен результат работы нейросети из матлаб.

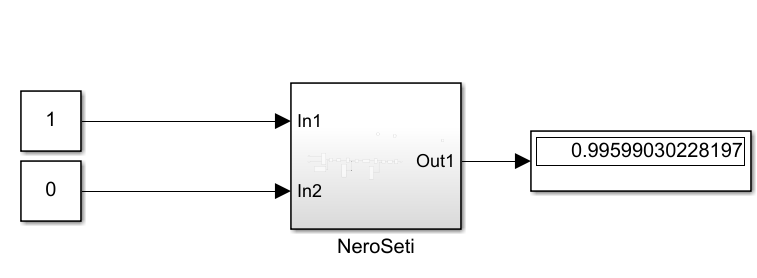


Рисунок 9 – Пример работы обученной сети в Matlab

Результат получился такой же, как и при использовании функции predict в Python, а это значит, что модель, созданная в matlab Simulink, является правильной и отвечает всем параметрам.

4. Обучение нейронной сети на однофазной модели

## 4.1 Построение модели в Matlab

Обучение нейронной сети должно происходить по схеме, показанной ниже.[6]

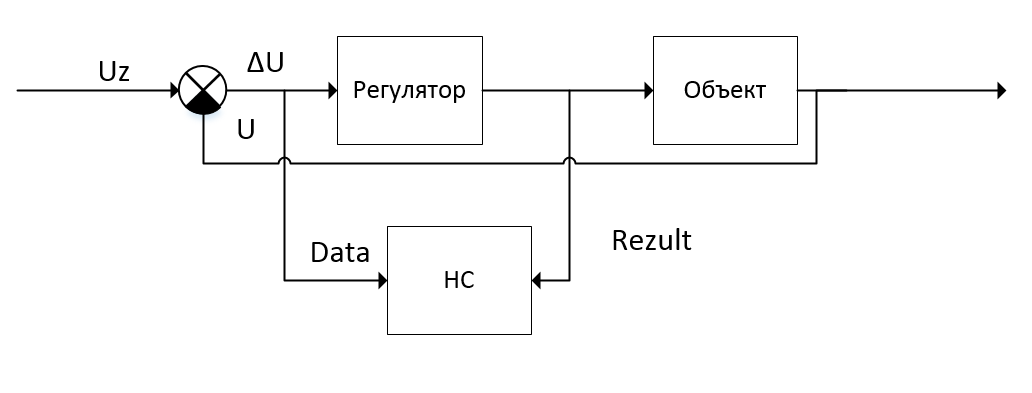


Рисунок 10 – Схема обучения нейронной сети на реальном объекте

Data – это параметр в котором в нейросеть будут передаваться данные на основе которых она будет обучаться. Этот параметр очень важен, потому что от входных данных зависит обучение и работа способность всей сети.

Rezult – результат работы исходного регулятора, который подается на выход нейросети, где она подсчитывает ошибку и меняет свои весовые коэффициенты для уменьшения ошибки.

Прежде чем обучать нейронную сеть на трехфазную систему. Необходимо обучить нейросеть на однофазную систему, чтобы выбрать следующие параметры: архитектуру сети, какие данные для обучения нужно использовать. На рисунке ниже представлена однофазная система, построенная в Matlab.

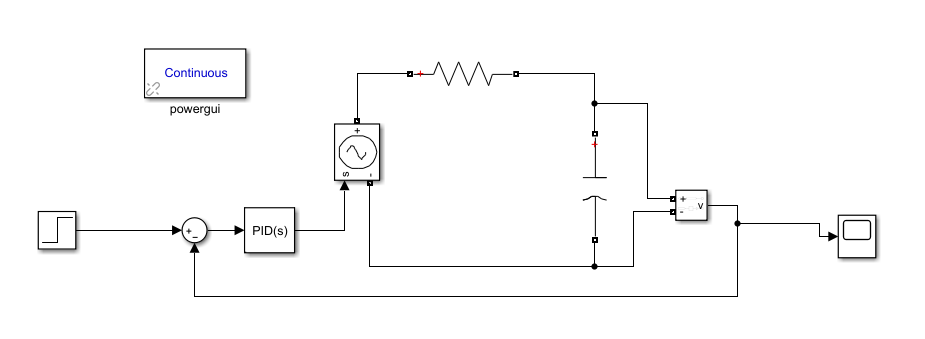


Рисунок 11 – Однофазная модель в Matlab

На вход ПИД регулятора подается ступенькой единица. Настройки регулятора представлены ниже. Регулятор подает сигнал управления на источник. Резистор имеет сопротивление 0.1 Ом.

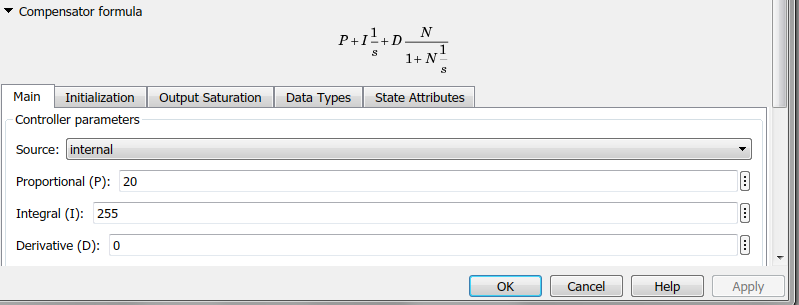


Рисунок 12 – Настройка ПИД регулятора

На вход регулятора ступенькой подается значение от нуля до единицы. Скачок происходит в момент времени равном 0.1 секунде.

Выход регулятора представлен на графике ниже. По графику можно увидеть корректную отработку регулятора

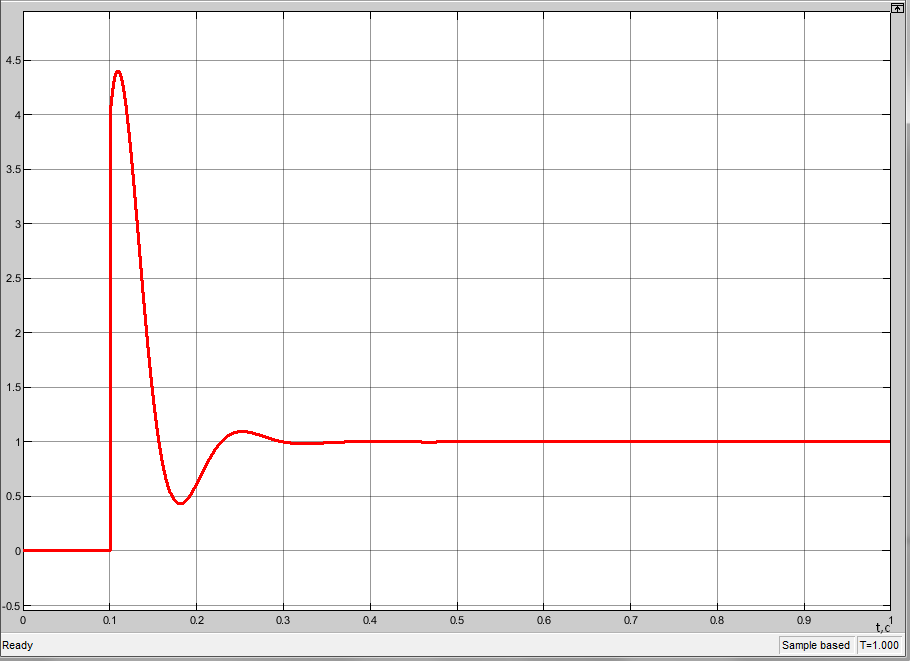


Рисунок 13 – График ПИД регулятора

По графику можно увидеть корректную отработку регулятора. На рисунке ниже представлен график напряжения на конденсаторе.

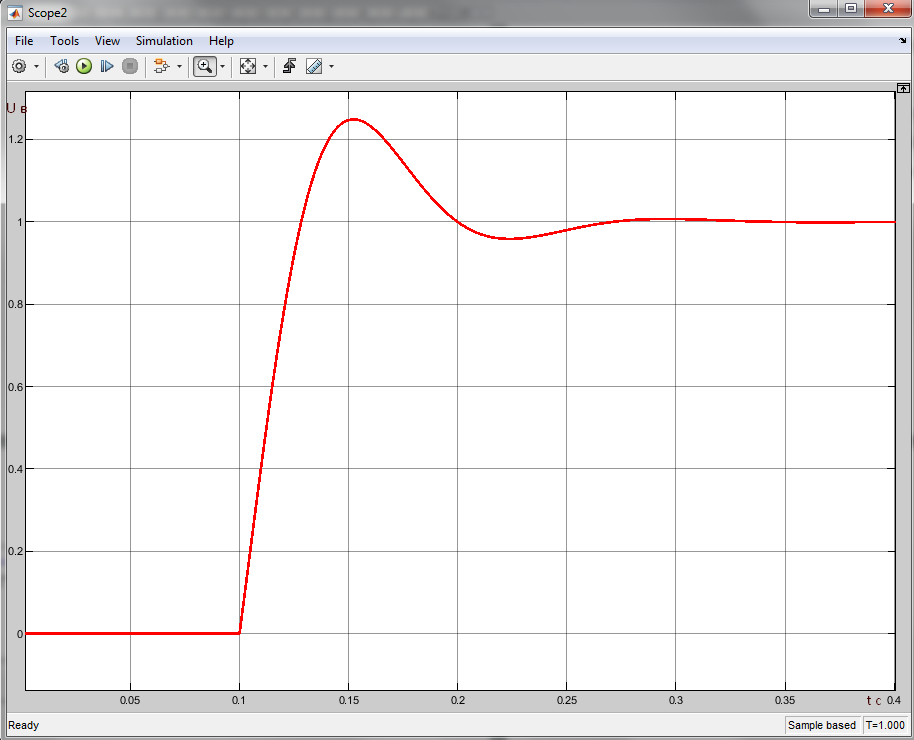


Рисунок 14 – Напряжение на конденсаторе

Эти данные будут приняты за эталон. Нейронная сеть должна будет обучится повторять эти значени, что значит вести себя как ПИД регулятор.

## 4.2 Обучение нейронной сети

Необходимо выбрать данные, которыми будет обучаться нейросеть. В начале архитектура сети буде один нейрон на входном слое, два скрытых слоя по два нейрона и один нейрон на выходном слое. Функциями активации между слоями будет функция линейного выпрямителя. Для выходного слоя гиперболический тангенс. Обучаться нейронная сеть будет только на Uz. График ошибки и график напряжения показаны ниже.

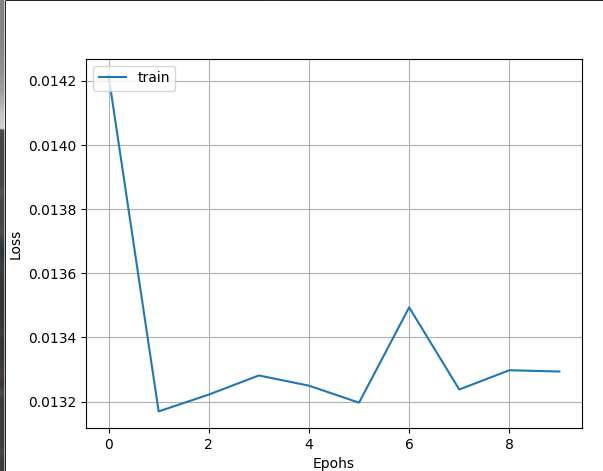


Рисунок 15 – График ошибки при обучение на UZ

Хоть ошибка и близка к нулю, но для НС это слишком большой показатель, что бы уменьшить ошибку необходимо, либо изменить входные данные, либо изменить архитектуру сети.

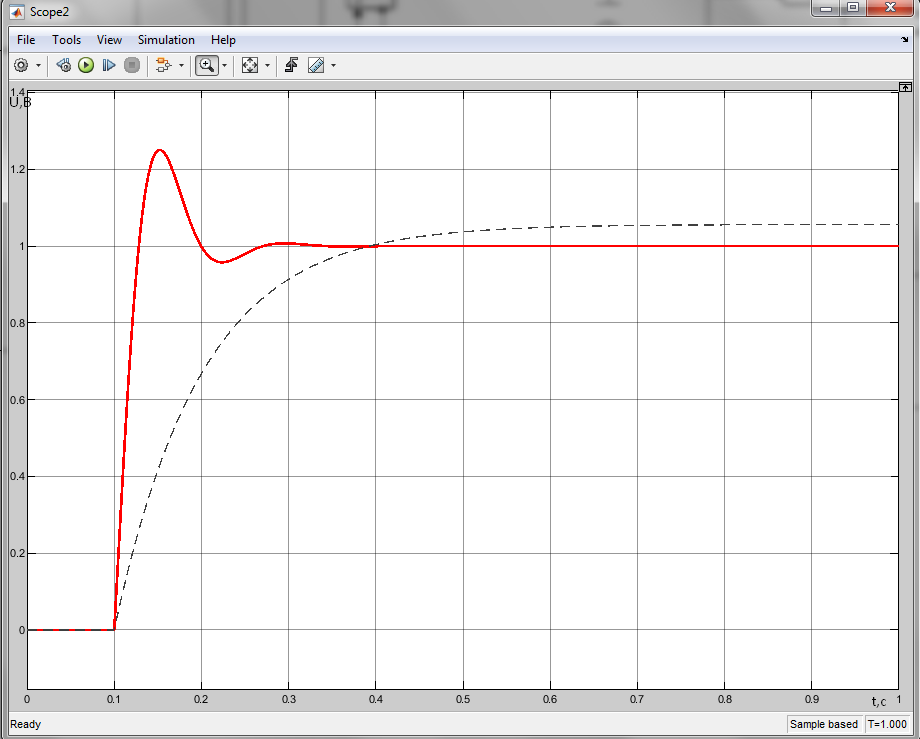


Рисунок 16 – График напряжения при обучение Uz

Сплошной линей показано напряжение на объекте с идеальным регулятором, пунктирной на объекте управляемым нейронной сетью. По графику можно увидеть, что напряжение от НС далеко от идеального, потому что не повторяет режим пере модуляции и не выходит на установившееся значение.

Для того чтобы улучшить показатели модели в качестве обучающих данных на модель будут подаваться Uz и U.Графики после обучения нейронной сети на Uz и U представлены ниже.

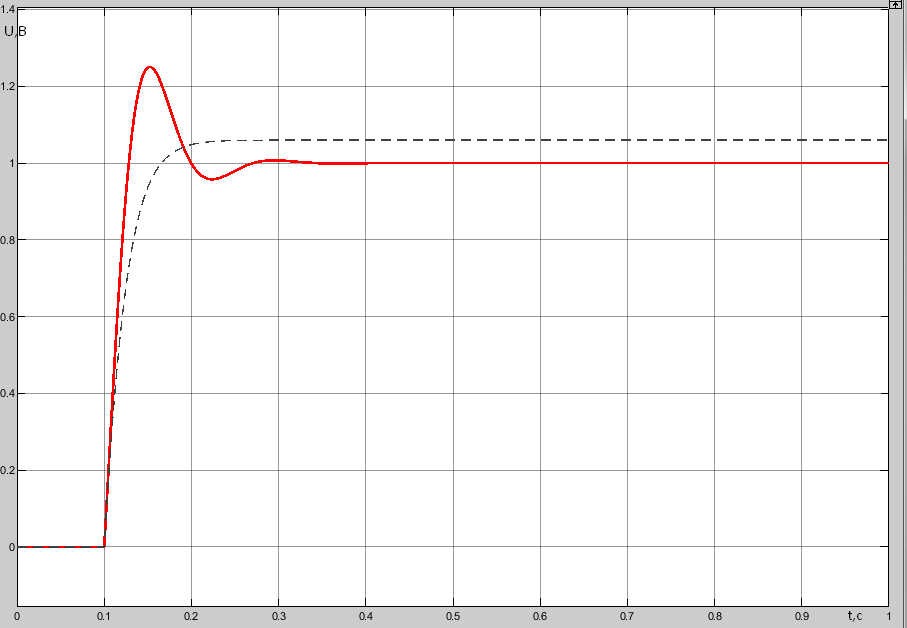
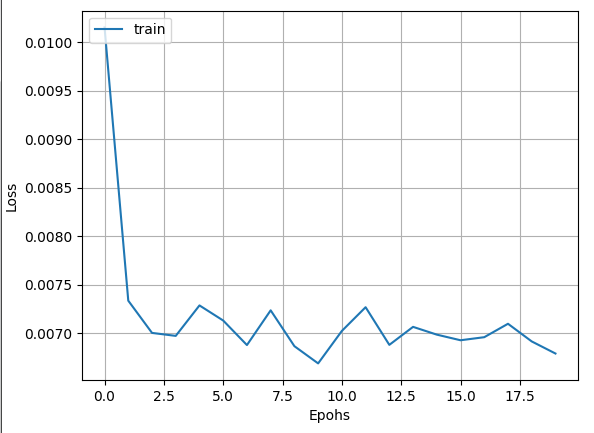


Рисунок 17 Функция ошибки и напряжения при обучение Uz и U

Ошибка обучения не может прийти к какому-то единому минимуму она постоянно меняется, и напряжение не повторяет эталонное, но уже лучше, чем в предыдущем опыте. Для улучшения показателей нейронной сети можно изменить ее архитектуру, увеличив количество нейроном в скрытых слоях на 10 и 5. График ошибки обучения представлен ниже.

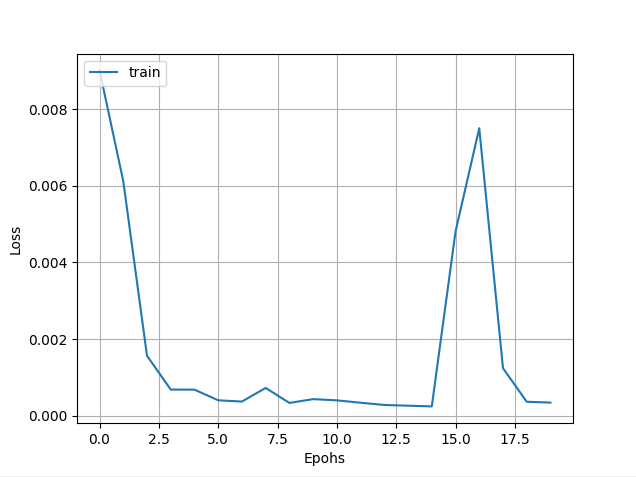


Рисунок 18 – График ошибки при изменение архитектуры сети

Хоть в моменте обучения и был аномальный всплеск ошибки, но за три эпохи ошибка снова опустилась в минимум и стала меньше чем при топологи сети, где только по два нейрона были в скрытых слоях.Из полученного графика можно сделать вывод, что изменение топологии сети привело к лучшей обучаемости сети а значит и повлияло на работоспособность сети.Ниже представлен график напряжения после изменения архитектуры сети.

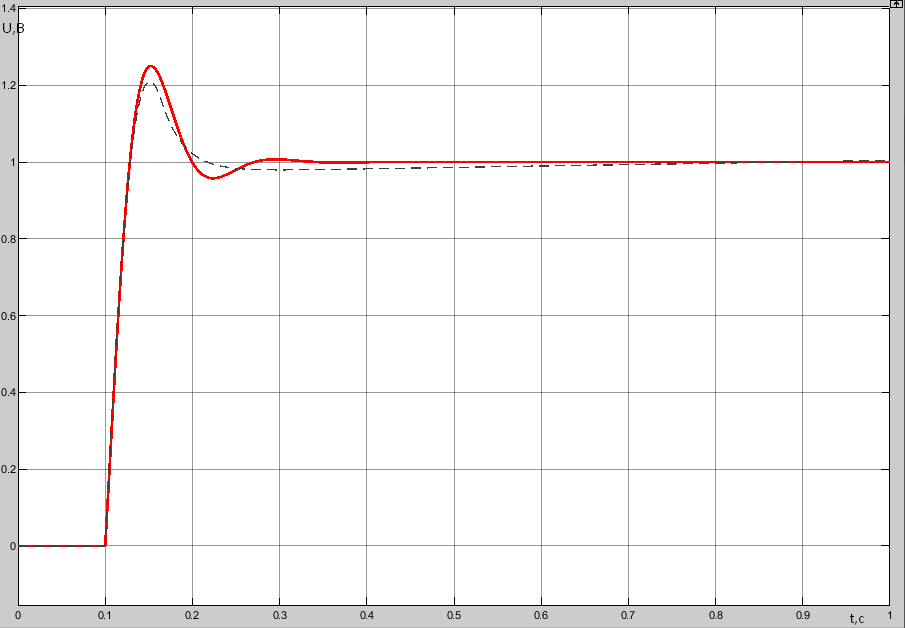


Рисунок 19 – График напряжения

По графику видно, что изменение топологии сети помогло нейронное сети воспроизвести момент перерегулирования напряжения, что тоже является хорошим показателем. Теперь нейронную сеть необходимо обучить на Uz и ΔU, чтобы выбрать при каких параметрах сеть обучается лучше. Ниже представлен график ошибки при заданных параметрах.

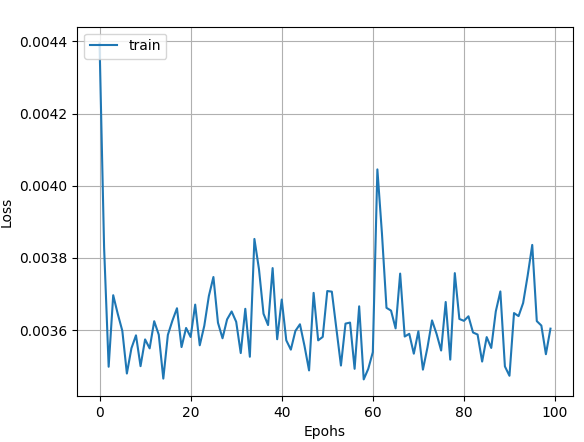


Рисунок 20 – График ошибки при обучение Uz, ΔU

Даже при увеличении количества эпох ошибка обучения не вышла на свой минимум, ее показание постоянно меняется, что является показателем того, что топология нейронной сети или входные параметры выбраны не удачно. Такие параметры обучения нейронной сети не являются сложными поэтому в первую очередь необходимо проверить ошибку в топологии нейронной сети. Известно, что при данном количестве скрытых слоев и нейронов в них нейронная сеть способна давать удовлетворяющий результат. Необходимо подумать о изменение функций активации в скрытых слоях. На данном этапе функциями активации является линейный выпрямитель. Его необходимо заменить на более конечную функцию, например, сигмоиду. График ошибки и напряжений после изменения функции активаций на сигмоиду представлен ниже.

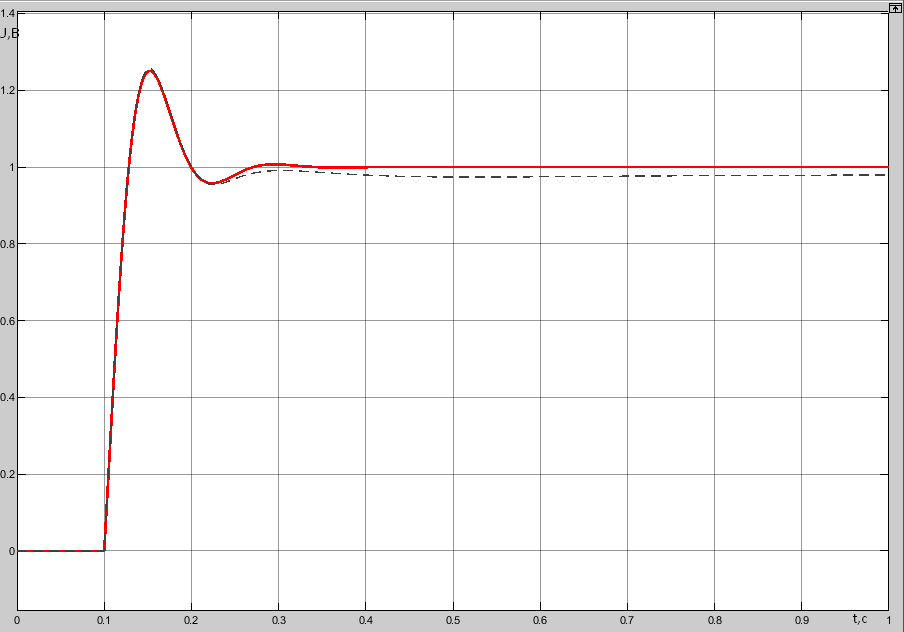
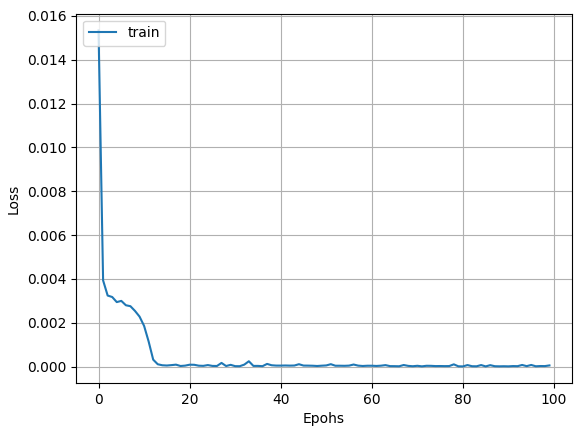


Рисунок 21 – Графики ошибки и напряжения после изменения функции активации

Данная топология сети и выходные параметры для обучения Uz и ΔU, лучше повторяют пререгулирование, но хуже удерживают систему при выходе на заданное значение. Последним эксперементом для однофазной сети, будет обучение сети на все три параметра: U, Uz, ΔU. Для того, что бы можно сделать вывод при каких параметрах обучения нейросеть лучше себя показывает.Ниже представлены графики ошибки и напряжений при обучение на U, Uz, ΔU.

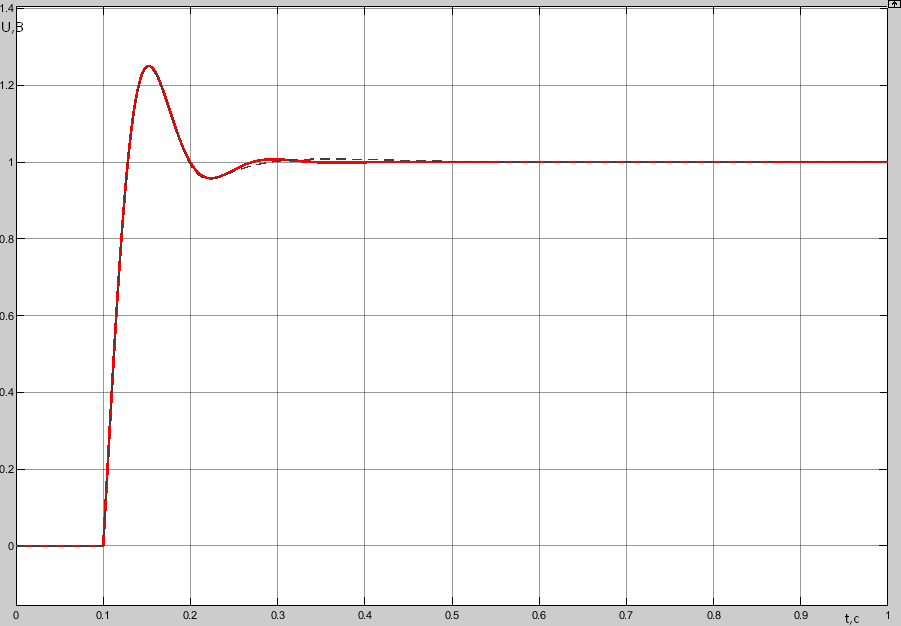
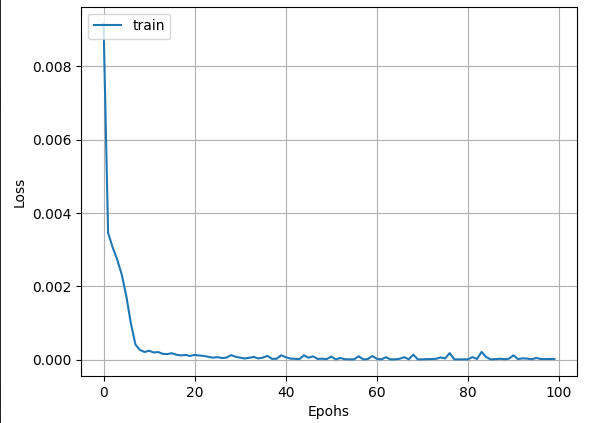


Рисунок 22 – Графики ошибки и напряжения при обучение нейронной сети на U, Uz, ΔU

На данных графиках видно, что добавление третего параметра в обучающую выборку на функции ошибки никак не отразилось, но сильно повлияло на работу нейронной сети с объектом. Теперь нейронная сеть лучше повторяет момент перемодуляции и лучше удерживает систему в заданном значени.

После проведенных эксперементов на однофазной системы, можно сделать выводы, что для обучения трехфазной системы в обучающих данных необходимо использовать заданное значение, значение на объекте и их разницу. Для того, что бы нейросеть правильно отрабытывала она должна содержать как минимум два скрытых слоя и в каждом должно быть как минимум пять нейронов. Во время обучения и работы лучше всего себя показало совместное использование функций активации сигмоида и гиперболический тангенс.

5. Обучение нейронной сети на трехфазную нагрузку

## 5.1 Создание трех фазной модели в Matlab

Обучение нейронной сети на трехфазную модель будет происходить по схеме показанной на рисунке 11. Модель будет создана как и предыдущая в Matlab Simulink, но будет отличаться не только количеством нагрузок, но и методом управления. Созданная модель показана на рисунке ниже.

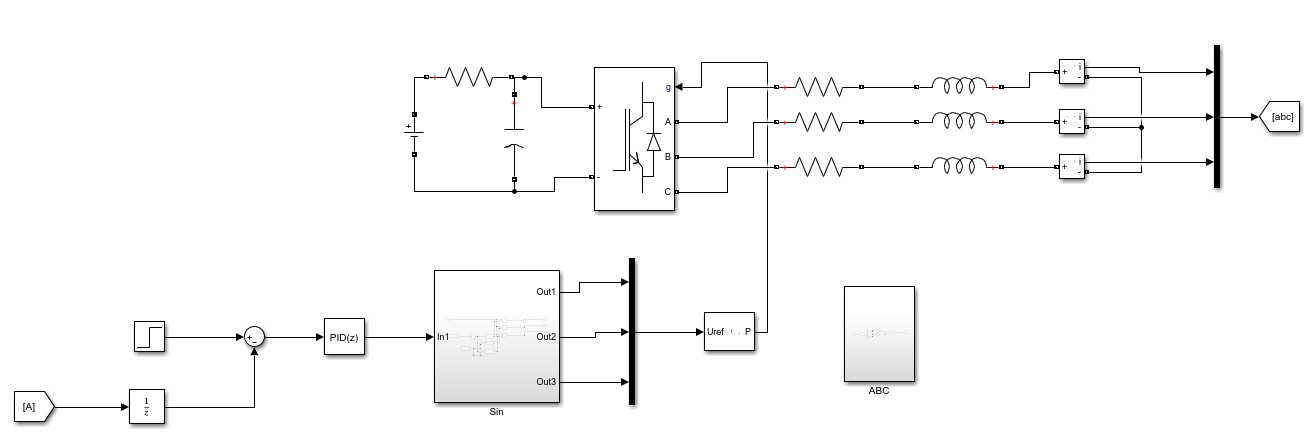


Рисунок 23 – Трех фазная модель

Данная модель имеет трех фазную нагрузку подключенную звездой, которая имитирует фазы в реальной машине. Так же в модели присутствует блок под названием Sin, в который подаётся значение регулятора и на выходе генерируется три синуса. Полученные синусы подаются на блок ШИМ, который генерирует импульсы управления IGBT модулем. Данный модуль реализован с помощью блока Universal Bridge. На модуль подаётся постоянное напряжение, которое равно 500 В. Из этого напряжения формируется переменное напряжение подающиеся на нагрузку. Для обеспечения обратной связи реализован блок под названием АВС. Подробное состав блока АВС представлен на рисунке ниже.

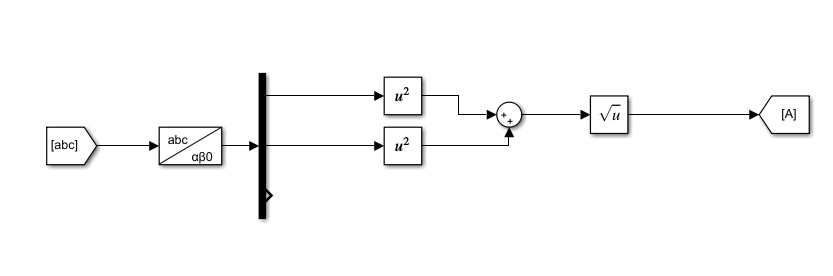


Рисунок 24 – блок для обеспечения обратной связи

Для обеспечения обратной связи по току необходимо вычислить амплитуду тока в нагрузке. Для этого синусы из нагрузки подаются на блок перевода из ABC в . Перевод осуществляется по следующей формуле:

,

Где ,, - токи в трёхфазной системе координат;

, - Токи в подвижной системе координат.

Более подробно это преобразование описано в книге Калачева Ю.Н.[7]

Для того, что бы получить амплитуду токов, необходимо применить теорему Пифагора к полученным токам . Это значение будет использовано как обратная связь для регулятора.

## 5.2 Выбор параметров R и L для обучения нейронной сети

Не целесообразно обучать нейронную сеть на все возможные вариации R и L нагрузок, потому что это ресурсоемкая задача для сбора такого большого количества данных.

Для того чтобы правильно подобрать диапазон изменения нагрузок для обучения на них нейронной сети необходимо ввести некоторые ограничения на составленную ранее модель. Так как питающее напряжение IGBT модуля составляет 500 В, то на каждой из фаз в зависимости от нагрузки можно получить значение тока не больше 220А на каждой фазе. Для наглядности обучения выбирается значение в 100 А на каждой фазе. Для того что бы получить фазный ток выбранного значения необходимо, чтобы общая нагрузка одной фазы составляла не больше 5 Ом. Питание всех фаз сети происходит за счёт подачи на каждую из фаз переменного напряжения частотой 20 Гц. Исходя из всего выше перечисленного, общую нагрузку каждой фазы необходимо рассчитывать по формуле:

,

где - ёмкостное сопротивление;

 - индуктивное сопротивление;

 - частота питающей сети.

В цепи отсутствует ёмкость, поэтому  будет равно нулю. Параметры R и L уже будут меняться в диапазоне, что бы  и не превышало значения в 5 Ом

Известно, что постоянная времени в RL цепи рассчитывается как:



Постоянная времени влияет на кривизну графика тока, время нарастания, до установившегося значения или нуля, следовательно, влияет на время переходного процесса. Так же чем больше значение  тем медленнее будет затухать переходный процесс.

Ниже представлена таблица, в которой отражены все нагрузки и их 

Таблица 2 – Параметры нагрузки для обучения сети

|  |  |  |  |
| --- | --- | --- | --- |
| R | L | Z |  |
| 0.1 | 0.01 | 1,259975 | 10 |
| 0.5 | 0.015 | 1,949219 | 33 |
| 1 | 0.015 | 2,132945 | 64.52 |
| 1,5 | 0.009 | 1,878245 | 166.67 |
| 2 | 0.005 | 2,096279 | 400 |

Данного количества точек для обучения нейронной сети должно быть достаточно. Значения, которые не присутствуют в обучающей выборке, но входят в ее диапазон для R элемента этот диапазон составляет 0.1 – 2, для L элемента 0.01-0.005, нейронная сеть должна будет сама под них подстроится и правильно отработать их.

## 5.3 Обучение нейронной сети

После обучения нейронной сети на однофазную нагрузку были выбраны параметры для обучения НС – это параметры I, Iz, ΔI, и выбрана архитектура сети – три нейрона на входном слое, десять нейронов на первом скрытом слое, пять нейронов на втором скрытом слое и один нейрон в выходном слое.

Пред обучением нейронной сети необходимо настроить ПИД регулятор, для правильной работы всей системы.

Первыми параметрами, на которые будет обучена нейронная сеть будут параметры сети R=0.1 L=0.001. На рисунке ниже представлены график тока и график обратной связи, для оценки переходного процесса в модели.

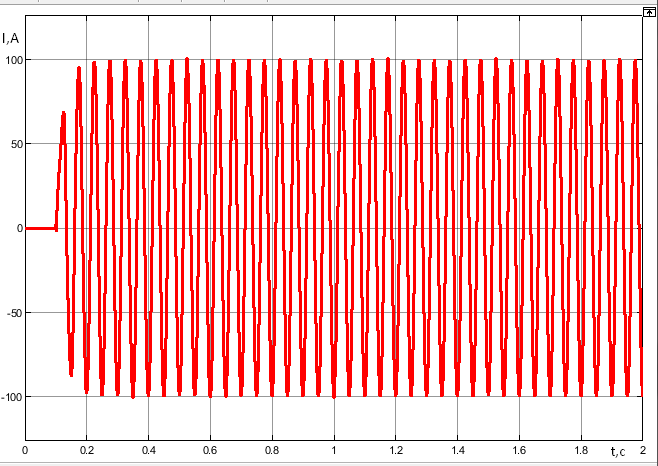
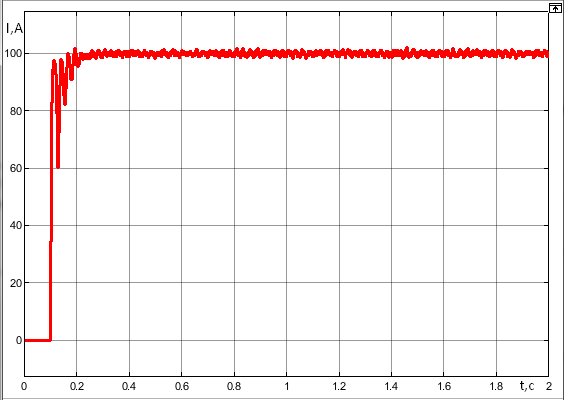


Рисунок 25 – Графики токов

По представленным графикам можно сделать вывод, что ПИД регулятор настроен приемлемо, во время переходного процесса значения сильно изменяются, но в допустимых значениях.

Ни же представлен график ошибки обучения НС.

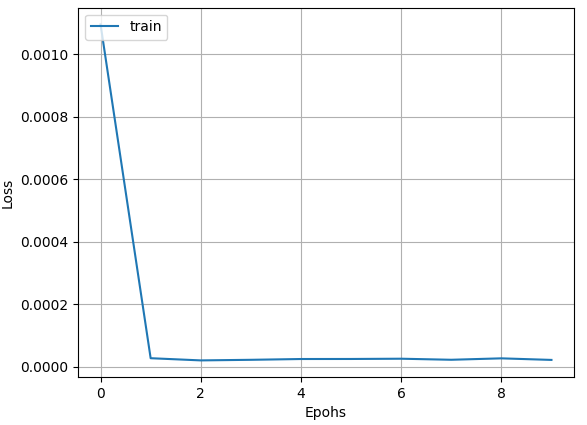


Рисунок 26 – График ошибки при обучение нейронной сети

По выше представленному графику можно сделать вывод, что НС обучилась, и архитектура сети выбрана правильно.

Ниже представлены графики токов при работе ПИД регулятора и НС на трехфазную нагрузку.

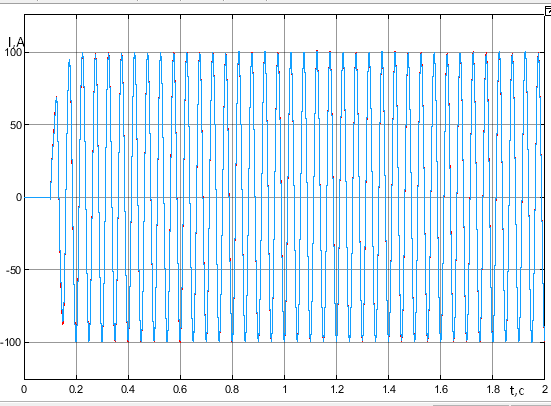
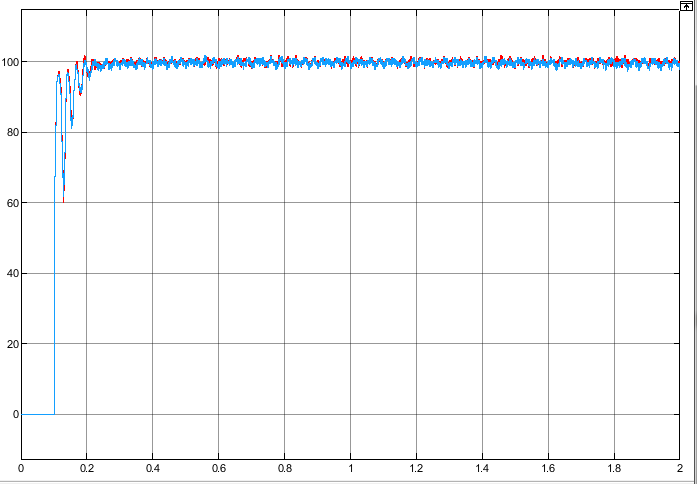


Рисунок 27 – Графики токов Нейронной сети и ПИД регулятора.

На графиках синим цветом является работа нейронной сети красным работа регулятора.По полученным графикам можно сделать вывод, что НС обучилась правильно и работает как нужно, потому что она в точности повторяет работу ПИД регулятора, на котором она и обучалась.

Далее необходимо обучить нейронную сеть на следующий параметр из таблицы 2. Для этого ПИД регулятор настраивается под новую нагрузку сети, снимаются данные для обучения и по новым данным модель доучивается на новые параметры сети.Ниже показан рисунок ошмбки обучения.

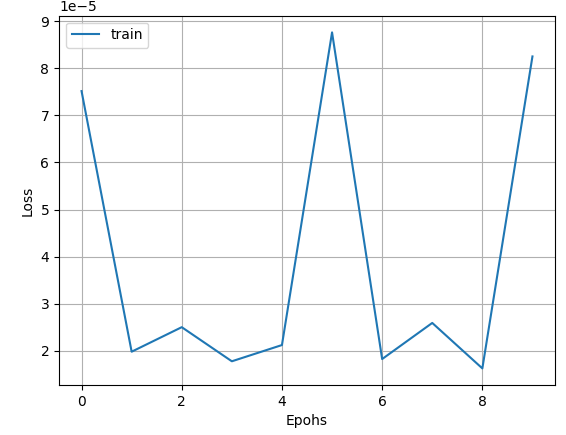


Рисунок 28 – Ошибка обучения при обучение на второй параметр

На показаном выше графике можно увидеть скачаки по ошибке, но они не значительны и общая ошибка остаеться близкой к нулю. На рисунках ниже представлены графики токов при обучение на второй параметр.

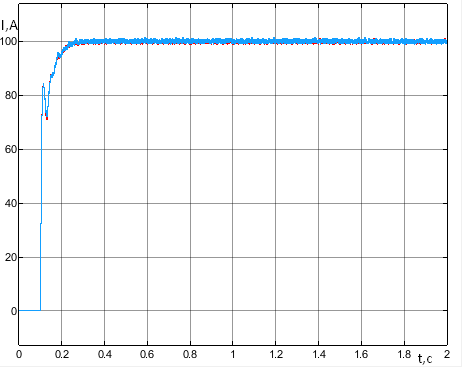
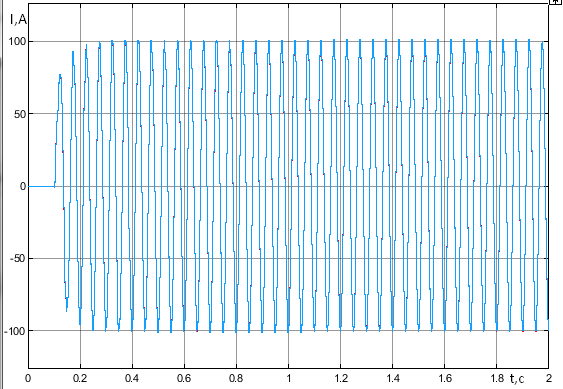


Рисунок 29 – Графики токов при обучение на второй параметр

По полученным графикам токов можно сделать вывод, что сеть обучилась на второй параметр из таблицы 2. Для того, что бы убедится, что сеть правильно отрабатывает на оба обученных параметра необходимо снова нагрузить систему первым параметром. На показанном ниже рисунки приведены графики токов.

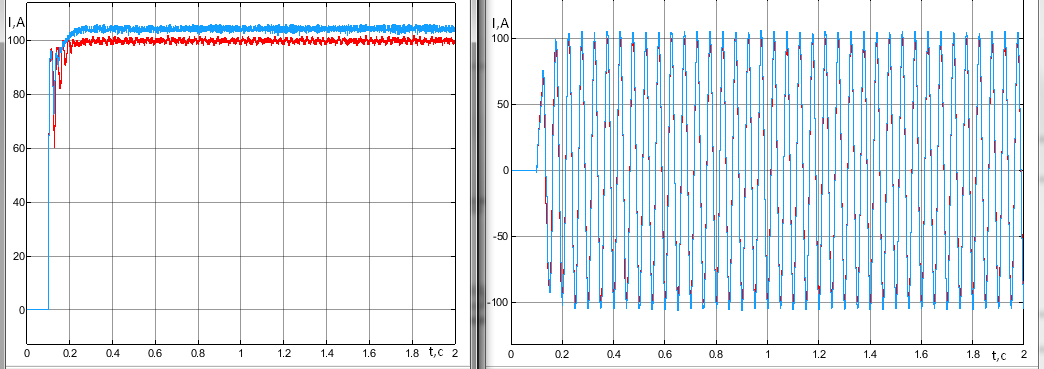


Рисунок 30 – Графики токов при первых параметрах нагрузки

Из полученных графиков можно сделать вывод, что нейронная сеть отрабатывает вторые параметры, но из-за доубочения сети она начала “забывать” первые параметры нагрузки. Возможно это связанно с тем, что в скрытых слоях малое количество нейронов, а параметров для обучения стало слишком много. Поэтому необходимо увеличить количество нейронов в скрытых слоях до 30 в каждом слое. На рисунке ниже представлен график ошибки.

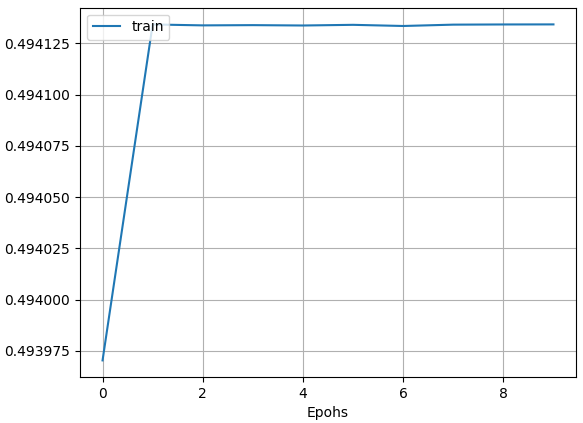


Рисунок 31 – График ошибки при изменение количества нейронов в скрытых слоях.

По полученному графику можно сделать вывод, что такое количество нейронов пагубно влияет на обучаемость сети. Из-за большого количества нейронов сеть начинает переобучаться и не выходит на минимум по ошибке. Изменим количество нейронов на 20 и 15.Ниже на рисунке показан график ошибки обучения и график тока.

По полученному графику ошибки можно сделать вывод, что изменение количества нейронов в скрытых слоя на 20 и 15 лучше влияет на НС, чем 30 нейронов в каждом слое. Нейронная сеть перестала переобучаться.

По графику тока можно сделать вывод, что увелечение количества нейронов не помогло. Нейронная сеть все равно путается в параметрах модели.

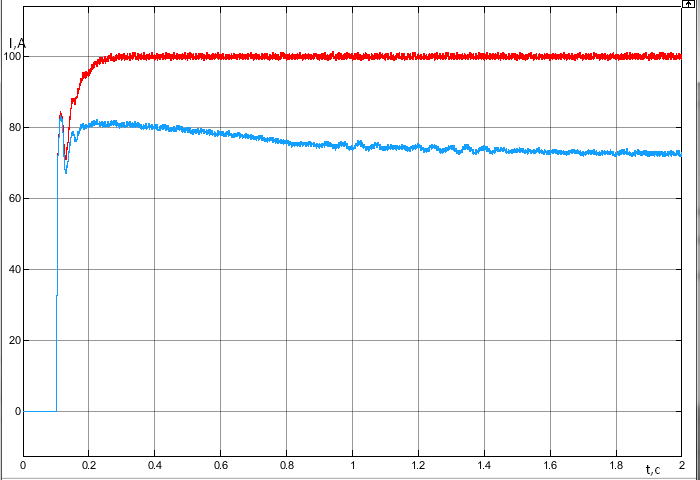
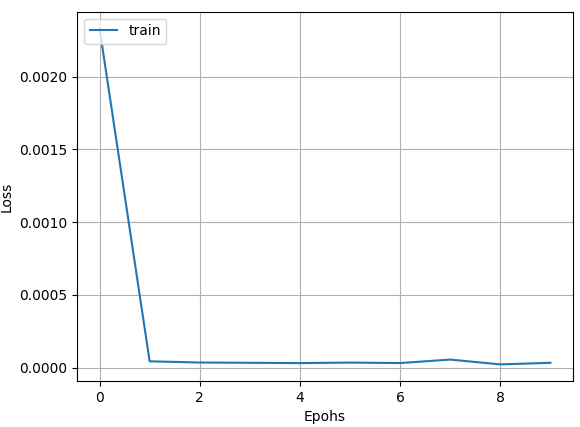


Рисунок 32 – Графики ошибки и тока.

Для того, что бы НС не путалась можно дополнительным параметром для входа ввести значения обратной связи по самой нейронной сети. Тогда такая НС уже будет называться рекурентной.

## 5.4 Рекурентные нейронные сети

Обычные нейронные сети не опираются на свои предыдущие значения. Они работают только с данными которые на них подаются. А рекурентные засчет введения обратной связи смотрят на свой предыдущий ответ и основываясь на полученных данных и предыдущем ответе делают предположения. На рисунке ниже приведена схема данной нейронной сети.

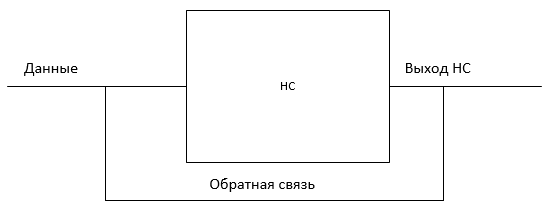


Рисунок 33 – Схема работы рекуррентной нейронной сети

На рисунке выше выход нейронной сети подается на вход. Это может показаться странным, но если немного обдумать эту схему то можно понять, что рекурентные нейронные сети можно представить как цепочку копий из обычных нейронных сетей, которые передают друг информацию. Если представить обратную связи как описанно выше, то получится следующая схема.

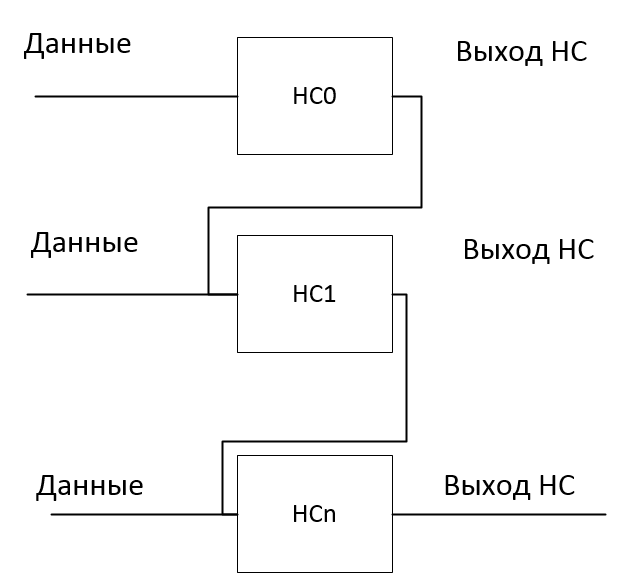


Рисунок 34 – Рекуррентная нейронная сеть в развертке

Такой тип нейронных сетей получил огромную популярность в целом ряде задач: распознавание речи, языковое моделирование текстов, перевод текстов с одного языка в другой, распознавание изображений, распознавание текстов и др. Более подробно о применение рекуррентных нейронных сетей можно прочитать на сайте Андрея Карпатого [8].

Для решения задач в данной выпускной дипломной работе, будет применяться РНС. Архитектура РНС будет применена из работ Джеффа Элмана [9], написанной в 1990 году, с небольшими изменениями.

Выход РНС через задержку данных в один такт будет подаваться на вход сети, что обеспечит ее обучение на предыдущем шаге работы.

## 5.5. Обучение нейронной сети с обратной связью

Необходимо заново обучить нейронную сеть на оба параметра из таблицы №2 с добавлением нового входного параметра, предыдущий шаг итерации. Ниже на рисунке представлен график ошибки при обучение

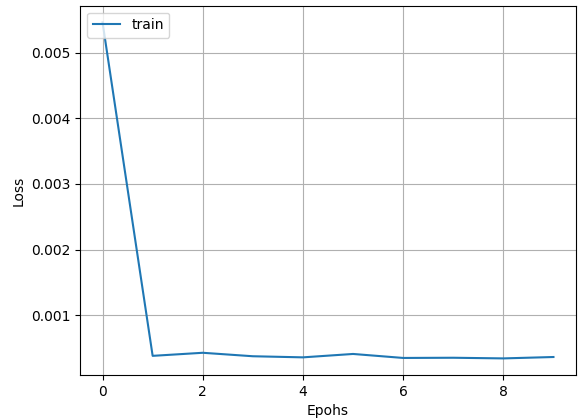


Рисунок 35 – График ошибки при обучение.

По полученному графику ошибки можно сделать вывод, что добавление обратной связи совсем немного влияет на обучаемость сети, значение ошибки немного уменьшилось. Ниже приведены графики тока при работе РНС на первый и второй параметр нагрузки

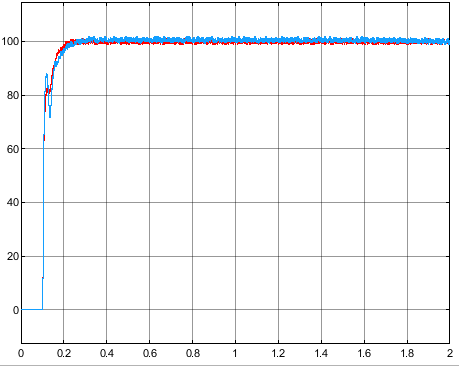
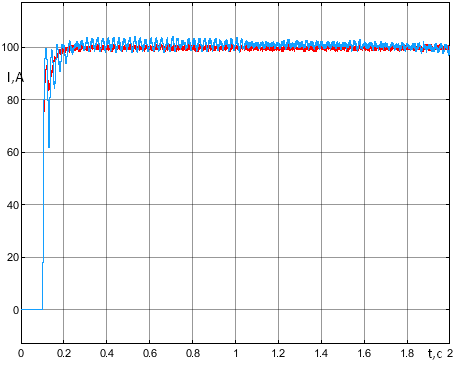


Рисунок 36 – Графики тока при работе на первый и второй параметр нагрузки

По выше представленным графикам тока можно сделать вывод, что добавление обратной связи нейронной сети помогло научить сеть, основываясь на своих предыдущих результатах различать параметры нагрузок. На первом графике параметр тока колеблется чуть больше чем регулятор. Возможно, это связано с недоученностью сети на этот параметр нагрузки, чтобы избежать этого сеть необходимо еще раз обучить на первый параметр нагрузки.

## 5.6. Результаты обучения нейронной сети на различные параметры трехфазной нагрузки.

Нейронная сеть имеет архитектуру четыре нейрона на входном слое, двадцать нейронов на первом скрытом слое, пятнадцать на втором скрытом слое, один нейрон на выходном слое. Функцией активации между скрытыми слоями является сигмоида, для выходного слоя гиперболический тангенс.

Обучение проводилось на все параметры, представленные в таблице №2. В данной главе описывается проверка нейронной сети на различные воздействия.

Основываясь на принципе работы нейронных сетей, обучающие данные, были не подробными и включали в себя только основные точки работы. Например, на управляющее воздействие нейронная сеть обучена только на значения ноль и сто, но несмотря на это она должна правильно отрабатывать любые значения, входящие в этот диапазон, возможно даже немного на большее значения этот момент тоже требует тщательной проверки.

Параметры нагрузки трех фазной сети тоже являются диапазон сопротивление нагрузки должно отрабатываться в диапазоне от 0.1 до 2, индуктивность имеет диапазон от 0.005 до 0.015.

На рисунке ниже представлены графики тока при роботе нейронной сети на различные управляющие воздействия, параметрами нагрузки сети выбраны 3 параметры из таблицы №2.

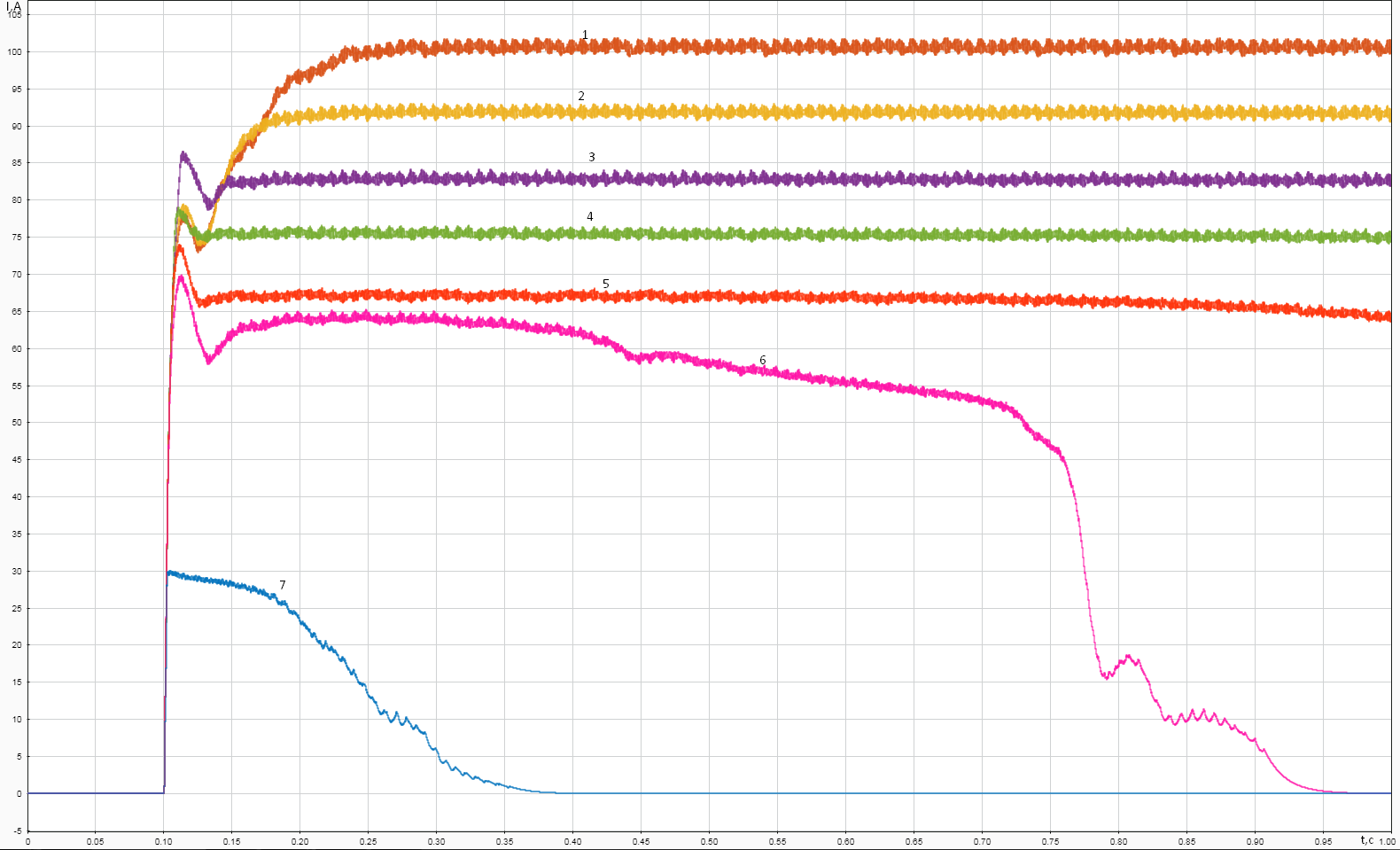


Рисунок 37 – Графики Амплитудных токов при различных управляющих воздействиях

На графике представленном выше показаны значения амплитуды тока при различных управляющих воздействиях, для наглядности графики подписаны цифрами. Расшифровка цифр представлена в таблице ниже.

Таблица 3 – Расшифровка номеров графиков

|  |  |
| --- | --- |
| Номер графика | Значение управляющего воздействия, А |
| 1 | 100 |
| 2 | 90 |
| 3 | 80 |
| 4 | 70 |
| 5 | 60 |
| 6 | 50 |
| 7 | 30 |

При управляющем воздействие равном 100 нейронная сеть правильно отрабатывает на управляющее воздействие и выводит значение амплитуды тока на заданное значение и удерживает его там. Когда управляющее воздействие снизилось до значения 90 нейронная сеть отрабатывает это значение, но уже чуть хуже чем предыдущее. После завершения переходного процесса значение тока колеблится около значения 92, что является плохим показателей но еще не особо критичным.При уменьшение значения управляющего воздействия, после перерегулирования нейронная сеть все хуже и хуже выходит на значение управляющего воздействи и разница между управляющим воздействием и установившемся значением растет практически линейно.

При значение управляющего воздействия равному 50 нейронная сеть потеряла возможность выйти на это значение и удержать его. Данная тенденция сохраняется и при подаче значения управляющего воздействия равному 30. Но если значние 50 нейронная сеть хоть как-то пытается отработать, то 30 она сразуже после выхода на него, не может удержать, потому что не знает что делать, в обучающей выборке небыло таких значений, и выводит значение тока в 0.

Для того что бы увеличить точность работы нейронной сети, необходимо было производить обучение на большее значение параметров управляющего воздействия. Нужно было обучать НС на параметры, хотя бы с шагом в 10, потому что по графикам можно увидеть, что в 15% процентном диапазоне нейронная сеть работает в пределах нормы и не дает большую ошибку по значениям.

Далее производилась проверка возможности нейронной сети осуществлять переход во время работы с одного значения управляющего воздействия на другое, но по выше сделаным выводам данная проверка проводилась со значениями в диапазоне от 80 до 100. Параметры нагрузки так же как и в предыдущем опыте выбирались из таблицы №2. На рисунке ниже представлены графики токов при данном опыте.

На графиках видно, что переходы выполняются с небольшой перемодуляцией, что является нормой, и выходят на значения уставки.

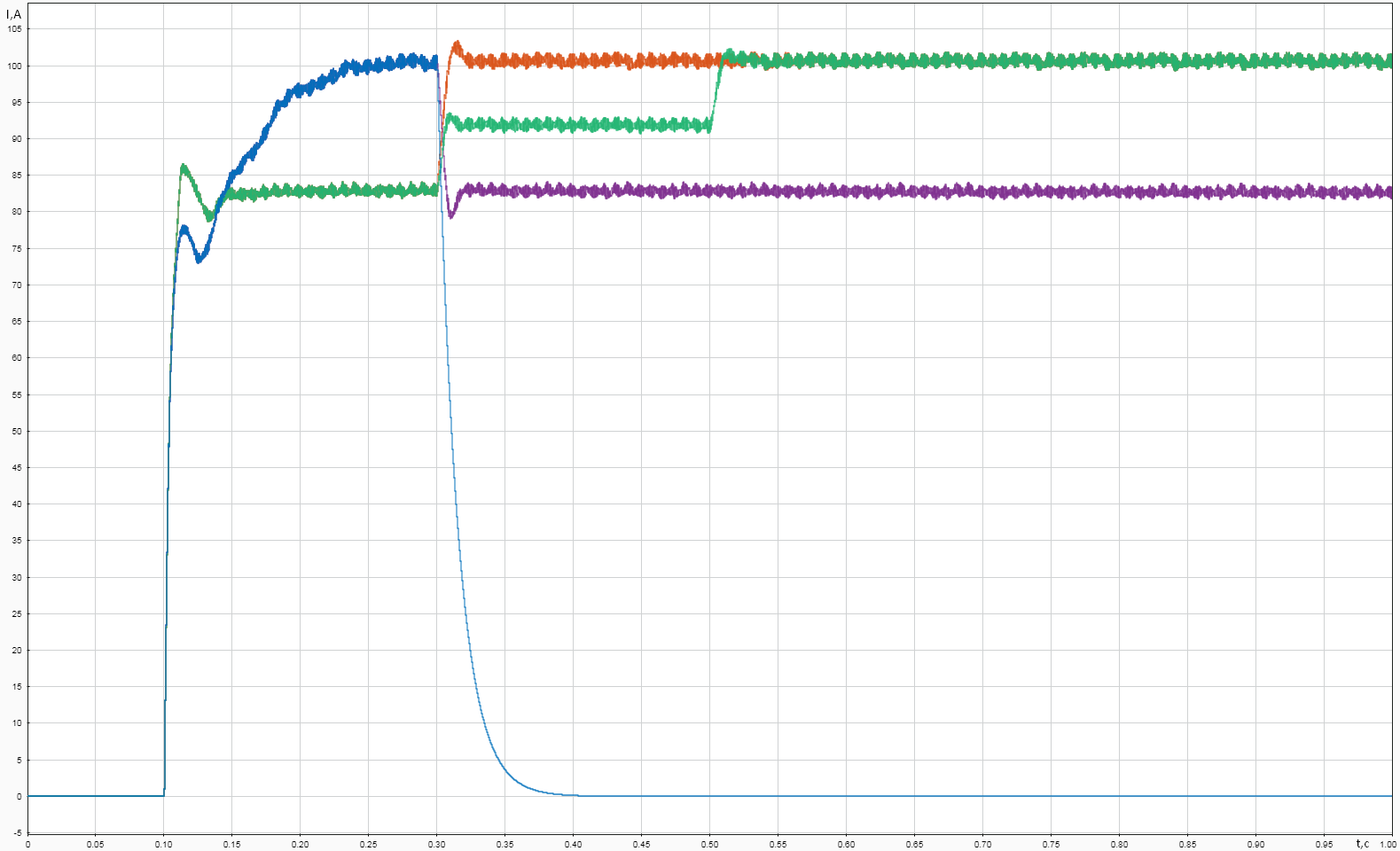


Рисунок 38 – Графики токов при различных возмущающих воздействиях.

По выше представленному графику можно сделать вывод, что нейроная сеть преходы между управляющими воздействиями выполняются правильно и точно также как и ПИД регулятор на основе которого производилось обучение.

Следующим проводится опыт при изменение параметров нагрузки. Параметры для нагрузки выбирались из обучающей выборки нейронной сети. Графики токов при данном эксперименте представлены ниже.

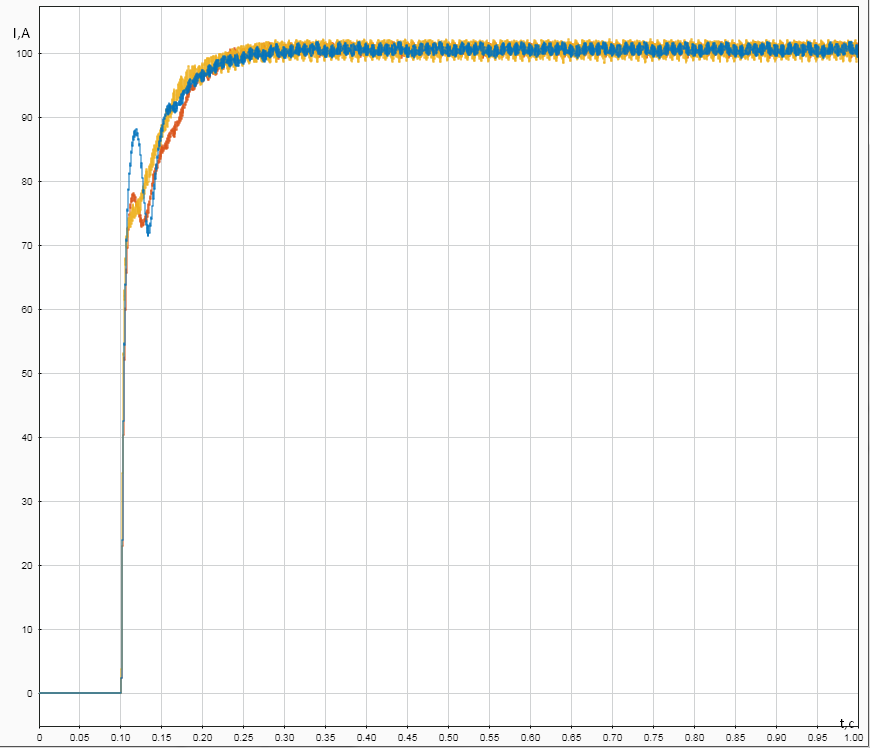


Рисунок 39 – Графики токов при различных параметрах нагрузки

По выше представленнному графику можно сделать вывод, что нейронная сеть правильно отрабатывает на обученные параметры. На графиках можно увидеть, что при различных параметрах нагрузки сети переходные процессы отличаются, что будет являться и различием при различных ПИД регуляторах.

Так же был проведен эксперемент в котором параметры сети задавались не из обучающей выборки и были найдены значения, которые выходят за диапазон обучающей выборки, приводящие нейронную сеть к неправильной работе.Графики токов данного эксперимента представлены ниже.

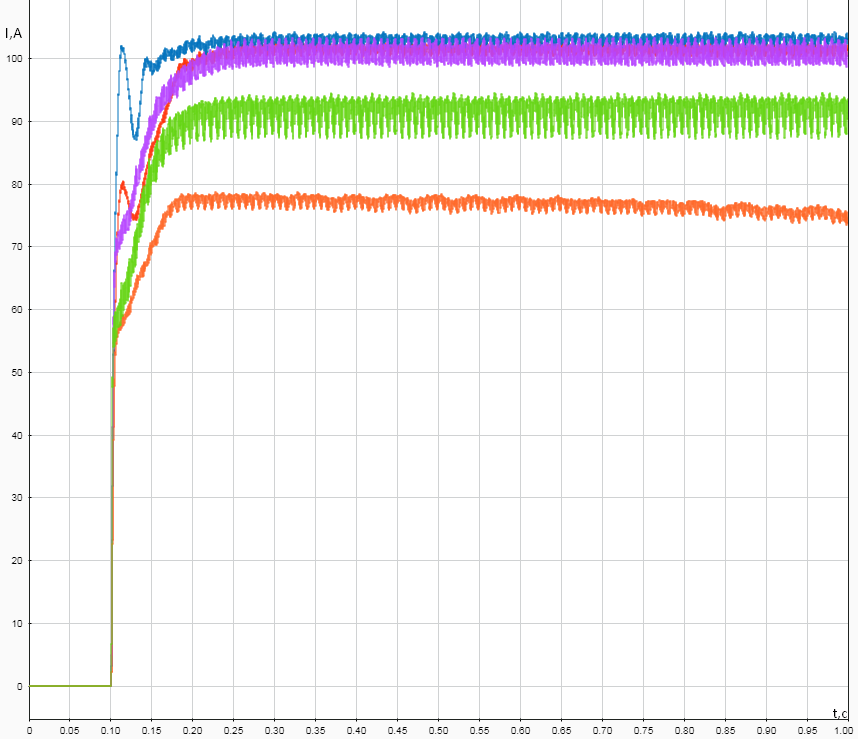


Рисунок 40 – Графики токов при различных нагрузках

На графике представленном выше показаны значения амплитуды тока при различных нагрузках не входящих в обучающую выборку. Для наглядности графики имеют различные цвета. Расшифровка, каждого цвета и значения параметров нагрузки представлены в таблице ниже.

Таблица 4 – Расшифровка цветов графиков

|  |  |
| --- | --- |
| Цвет | Значение параметров нагрузки |
| Красный | R = 0.8 Ом; L = 0.012 Гн |
| Синий | R = 0.3 Ом; L = 0.011 Гн |
| Фиолетовы | R = 1.7 Ом; L = 0.007 Гн |
| Ораньжевый | R = 0.3 Ом; L = 0.018 Гн |
| Зеленый | R = 2.4 Ом; L = 0.015 Гн |

По выше представленному графику можно сделать вывод, что нейронная сеть правильно отрабатывает различные параметры нагрузки, которые не входят в обучающую выборку, но входят в ее диапазон.

При задаче параметров нагрузки сети не входящих в обучающую выборку и выходящих за ее диапазон. Можно увидеть тенденцию как и в предыдущем эксперементе со сменой значения уставки. Нейронная сеть начинает неправильно обрабатывать значения приходящие ей на вход, и поэтому не может корректно удерживать управляющее воздействие. Примерно при 15%-20% отклонения от диапазона параметров нагрузки можно заметить неправильность работы сети.

6. Обучение нейронной сети на различные типы асинхронного двигателя

## 6.1. Создание системы управления асинхронным двигателем в Matlab Simulink

Следующим этапом исследования возможностей нейронных сетей будет, замена регуляторов скорости, в системе управления асинхронного электропривода. Для проведения опытов и создания обучающей выборке была разработана математическая модель электропривода в среде MatLab, она состоит из звена постоянного тока, инвертора, асинхронного двигателя, ШИМ модуля, который подает управляющие сигналы на полупроводниковый мост, а так же блока реализующего законы управления, в нём и будет заменен ПИД-регулятор скорости на нейронную сеть. Разработанная математическая модель представлена на рисунке 42.

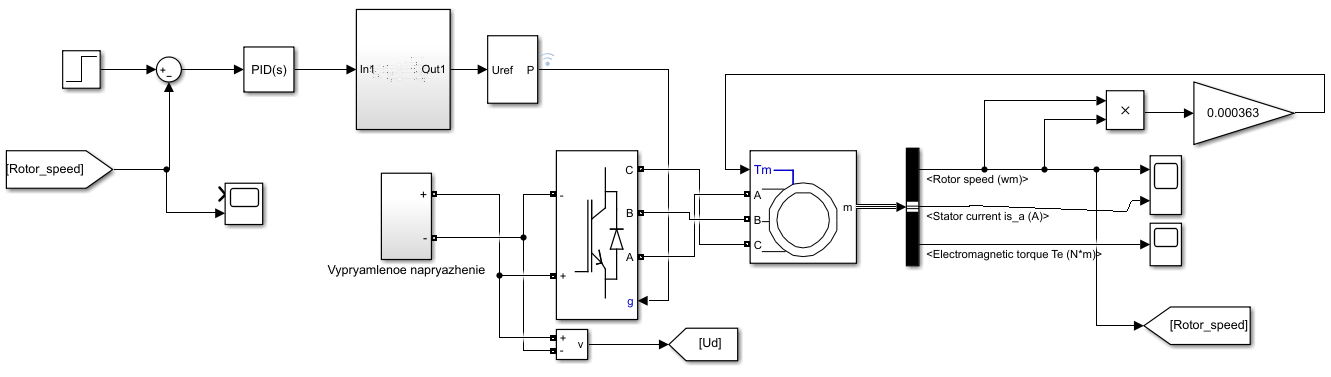


Рисунок 41 – Схема асинхронного двигателя в Matlab Simulink

Для создания обучающей выборки будут взяты несколько стандартных моделей асинхронных двигателей, которые предоставляет среда разработки MatLab, в системе управления, для каждого двигателя будет настроен ПИД – регулятор скорости, на двигатели будет подаваться вентиляторная нагрузка, зависящая от квадрата скорости вращения ротора электродвигателя.

Задачей нейронной сети будет являться отработка заданного значения скорости на различных типах двигателей, которые входят в диапазон обучающей выборки.

## 6.2. Обучение нейронной сети.

Основываясь на предыдущих экспериментах и результатам этих экспериментов, в обучающую выборку будут входить следующие параметры:

1. - скорость, которая будет задаваться, и которую нейронной сети необходимо будет достичь и удерживать. Основываясь на опыте, проведенном с трехфазной цепью, и учитывая, что задание скорости не изменяется скачкообразно, диапазон значений для  должен быть от нуля до номинальных оборотов АД. Так же для правильного обучения на весь диапазон значений необходимо производить обучение с 10% шагом.

2.  - скорость вращения ротора двигателя.

3.  - разница между  и .

4. Обратная связь нейронной сети – значение выхода нейронной сети на предыдущем шаги итерации.

Возможно, для обучения нейронной сети на работу как регулятора скорости необходимо будет введение в обучающую выборку дополнительных параметров, например, мощность асинхронного двигателя или тип нагрузки, что бы обучать нейронную сеть на различные нагрузки.

Для обучения нейронной сети будут использоваться стандартные асинхронные двигатели, предоставляемые Matlab Simulink в блоке под название “Asynchronous Machine SI”. Праметры предоставляемых двигателей представлены в таблице 5.

Таблица 5 – Параметры двигателей для обучения нейронной сети

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Номер двигателя | Мощность (кВт) | Номинальное напряжение двигателя (В) | Номинальная частота (Гц) | Номинальные обороты | Номинальная нагрузка(Н) |
| 1 | 3.68(5) | 460 | 60 | 183 | 20.1 |
| 2 | 7.36(10) | 460 | 60 | 184 | 40 |
| Номер двигателя | Мощность  (кВт) | Номинальное напряжение двигателя  (В) | Номинальная частота  (Гц) | Номинальные обороты  (об/мин) | Номинальная нагрузка  (Н) |
| 3 | 14.71(20) | 460 | 60 | 184 | 79.9 |
| 4 | 36.78(50) | 460 | 60 | 185 | 198.8 |
| 5 | 73.55(100) | 460 | 60 | 185 | 397.5 |

Обучение нейронной сети будет происходить на двигатели под номерами 1,3 и 5, для того чтобы проверить сможет ли НС правильно работать при двигателях другой мощности, не входящих в обучающую выборку, но входящих в диапазон обучения. Во время обучения для каждого двигателя будет настроен ПИД регулятор и собраны обучающие выборки с параметрами описанными выше. Во время её сбора двигатель в течение одной секунды будет разгоняться до своих номинальных оборотов, в качестве нагрузки, как писалось раннее, будет использоваться вентиляторная нагрузка.

Архитектура нейронной сети будет следующей: 4 нейрона на входном слое 20 нейронов на первом скрытом слое, в качестве функции активации будет использоваться сигмоида, 15 нейронов на втором скрытом слое, так же будет использована в качестве функции активации сигмоида, и 1 нейрон на выходном слое с функцией активации гиперболический тангенс.

Для начала необходимо обучить нейронную сеть на первый двигатель и проверить правильность ее работы. Ниже на рисунке представлены график ошибки во время обучения и график скорости двигателя.

По графику ошибки можно сделать вывод, что нейронная сеть обучилась и ошибка находится очень близко к нулю. По графику скорости двигателя при работе от нейронной сети можно сделать вывод, что НС обучилась неправильно.

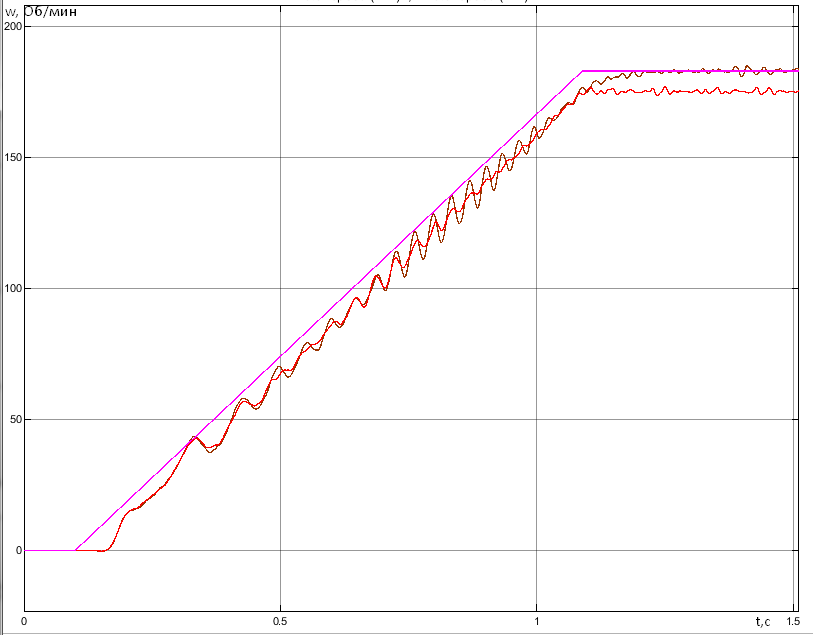
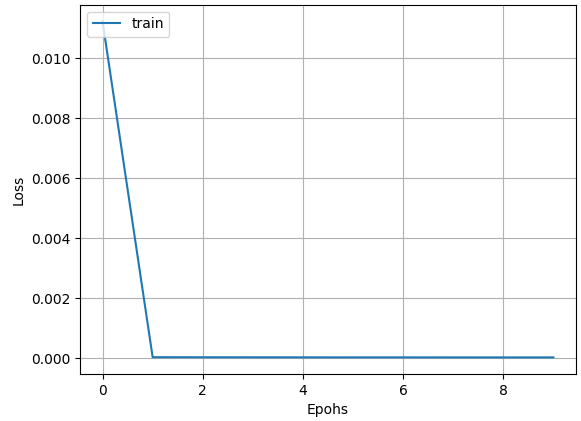


Рисунок 42 – График ошибки и скорости двигателя

На графике скорости представленном выше приняты следующие обозначения:

* розовым цветом показано управляющее воздействие;
* коричневым цветом показана скорость вращения двигателя при работе от ПИД регулятора;
* красным цветом показана скорость вращения двигателя при работе от нейронной сети.

Как говорилось раннее НС не выходит на номинальные обороты. Эта проблема связанна с тем, что выходом нейронной сети является диапазон чисел от -1 до 1. Из-за того, что функции активации сигмоида и гиперболический тангенс асимптотически стремятся к значениям -1 и 1 они никогда не достигают этих значения. А так, как при работе двигателя на номинальные значения оборотов необходимо питать его номинальным значением напряжения, а диапазон напряжений был выбран от 0 до 460, то и значение 460 никогда не получится подавать на нейронную сеть. Для решения этого вопроса было принято решение увеличить диапазон напряжения в обучающей выборке с 460 до 500.

Так же для улучшения обучения необходимо увеличить время разгона двигателя в обучающей выборке с одной секунды до двух. При разгоне за одну секунду слишком быстро происходит нарастание скорости, и из-за это нейронная сеть обучается на меньшем количестве точек, что так же может приводить к неправильной работе НС при быстрых разгонах двигателя.

Так же по полученному графику скорости можно сделать вывод, хотя нейронная сеть и не выходит на номинальные обороты двигателя, но она меньше пульсирует во время разгона, чем ПИД регулятор на примере которого она обучалась. Это значит, что ей удалось найти более оптимальные параметры управления, чем регулятор.

Ниже на рисунке представлен график скорости при внесение выше описанных изменений в обучающую выборку.

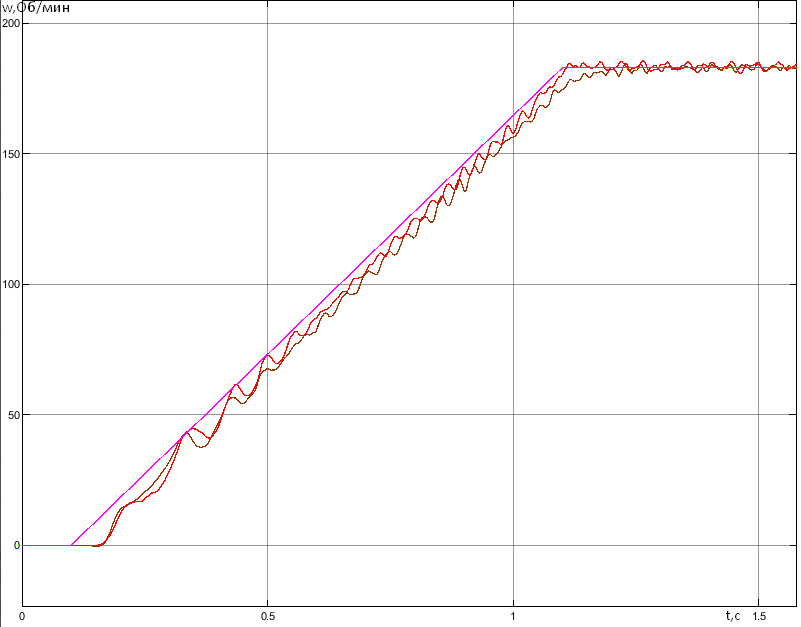


Рисунок 43 – График скорости при изменение обучающей выборки

По выше представленному графику можно сделать выводы, что изменение обучающей выборки положительно повлияло на работу нейронной сети. Нейронная сеть чуть больше колеблется, чем при предыдущем обучение, но зато она без проблем удерживает номинальные обороты двигателя и выходит на них чуть быстрее чем ПИД регулятор на основе которого она обучалась.

## 6.3. Результаты обучения нейронной сети

Первым делом необходимо проверить, как обучилась нейронная сеть, на все остальные двигатели, которые входили в обучающую выборку.

Ниже на рисунке представлен график скорости 5-го двигателя, из таблицы № 5, в котором сигнал управления приходит от нейронной сети и график скорости такого же двигателя, только сигнал управления задается заранее настроенным ПИД регулятором.

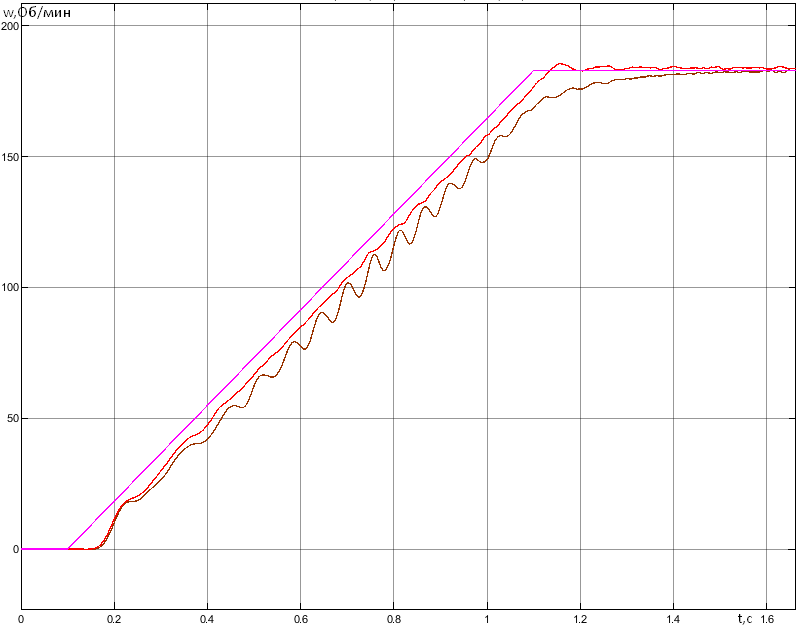


Рисунок 44 – График скорости 5-го двигателя из таблицы № 5

На выше представленном рисунке, можно заметить колоссальное различие в работе нейронной сети и ПИД регулятора. В качестве обучающей выборки хоть и подавались данные с регулятора, но уже видно, что нейронная сеть намного лучше отрабатывает сигналы и задаёт управляющее воздействие на двигатель, чем ПИД регулятор. Нейронная сеть плавнее выходит на заданное значение скорости. Подавая, управляющий сигнал она практически не колеблет скорость двигателя при разгоне, в то время как ПИД регулятор колеблет значение скорости очень сильно. Так же нейронная сеть намного быстрее выходит на заданное значение скорости.

Лучшая работа нейронной сети по сравнения с регулятором, на данных которого она обучалась, предположительно, связанно с тем, что в обучающей выборке данные были с различных ПИД регуляторов и обучившись на них всех она научилась, находить более оптимальные значения для управления асинхронным двигателем, чем обычный ПИД регулятор.

Следующим опытом, необходимо проверить работу нейронной сети в качестве регулятора скорости для двигателя не входившего в обучающую выборку, но входящего в диапазон обучения. График скорости представлен на рисунке ниже.

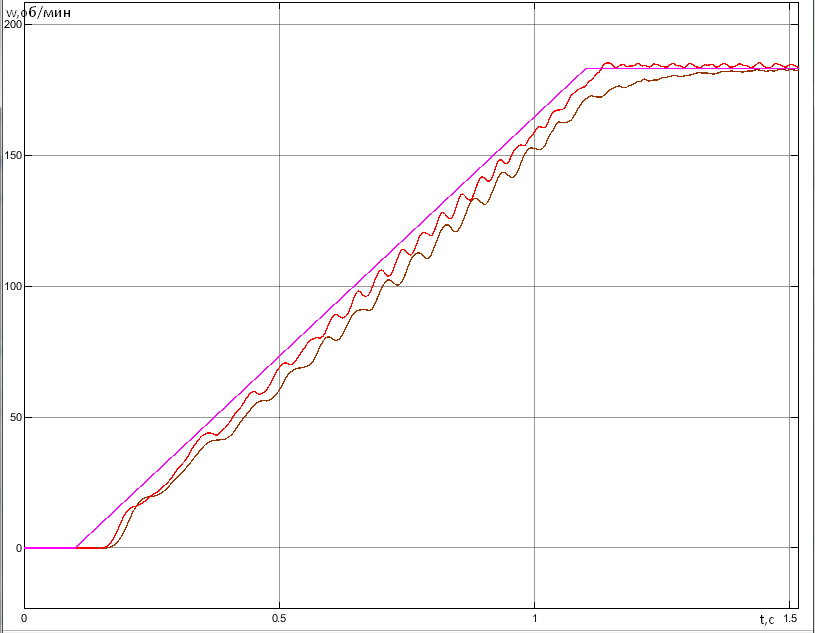


Рисунок 45 – График скорости при работе нейронной сети и двигателя №2 из таблицы.

Для того что бы во время проведения эксперимента были данные для оценки правильности работы нейронной сети на графике скорости показана скорость вращения двигателя управляемого ПИД регулятором настроенным на двигатель №3 из 5 таблицы. В качестве объекта управления будет выбран двигатель №2 из таблицы 5, который не входил в обучающую выборку

По графикам, представленным выше можно сделать вывод, что нейронная сеть правильно выдает управляющие воздействия для управления двигателем. Она плавно и быстро выходит на заданное значение, в то время как регулятор справляется с поставленной задачей хуже, дольше выходит на заданное значение по скорости и во время разгона намного сильнее колеблится скорость во время разгона двигателя. Так происходит, потому что регулятор настроен на двигатель совсем другой мощности. Данный эксперимент показывает, что нейронные сети более вариативны, чем обычные регуляторы, потому что их необходимо настраивать под каждый двигатель, а нейронную сеть достаточно обучить на выборку двигателей и она будет правильно выдавать на них управляющие воздействие, для того что бы выходить на заданную скорость.

На диапазоне двигателей, который входили в обучающую выборку нейронная сеть отрабатывает правильно. Для проверки того как НС поведет себя, если в качестве управляющего объекта выбрать двигатель, который не входил в обучающую выборку, будет выбран двигатель со следующими параметрами:

1.Мощность двигателя 110.33 кВт.

2.Номинальное напряжение двигателя 460 В.

3.Номинальная частота 60 Гц.

4.Номинальные обороты двигателя 184.5 об/мин.

5.Номинальная нагрузка 597 Н.

В качестве нагрузки, как и в предыдущих опытах, будет использована вентиляторная нагрузка.

На рисунке ниже представлен график скорости данного опыта.

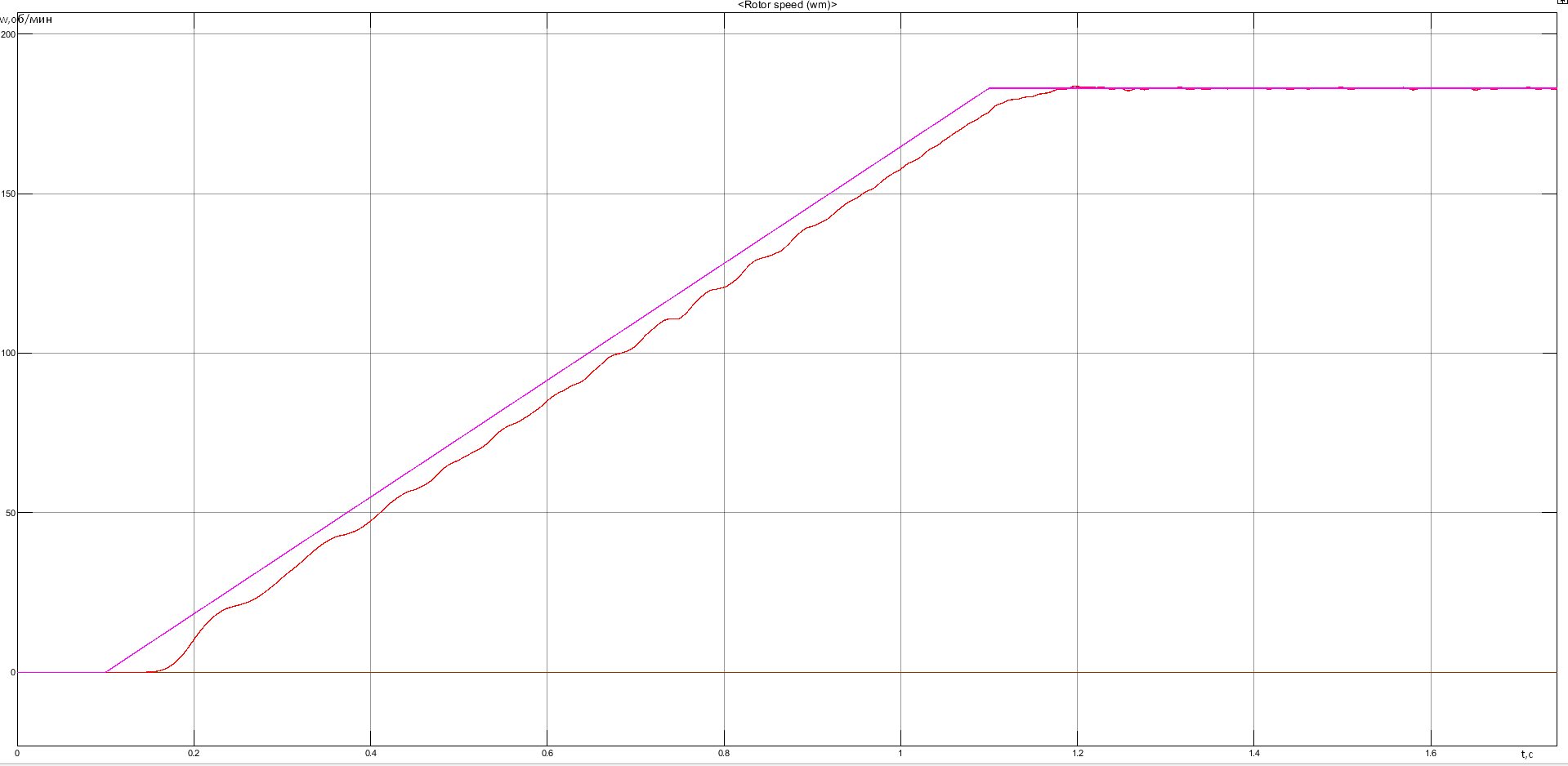


Рисунок 46 – График скорости двигателя при работе нейронной сети на двигатель, не входивший в обучающую выборку

Нейронная сеть при управление двигателем должна не только уметь разгоняться, но еще и уметь тормозить. В обучающей выборке нейронная сеть не обучалась на торможение, поэтому следующий опыт будет проведен на торможение. На рисунке ниже представлен график скорости двигателя во время разгона и торможения.

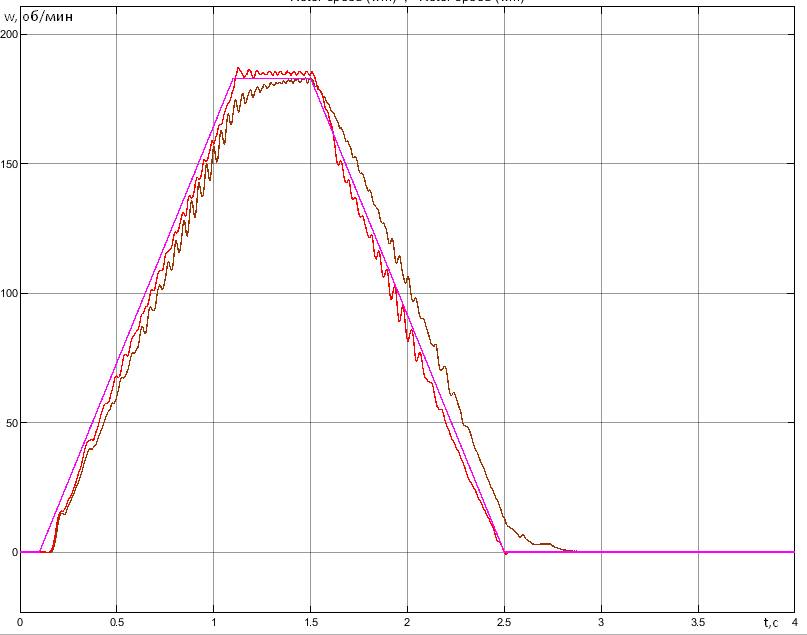


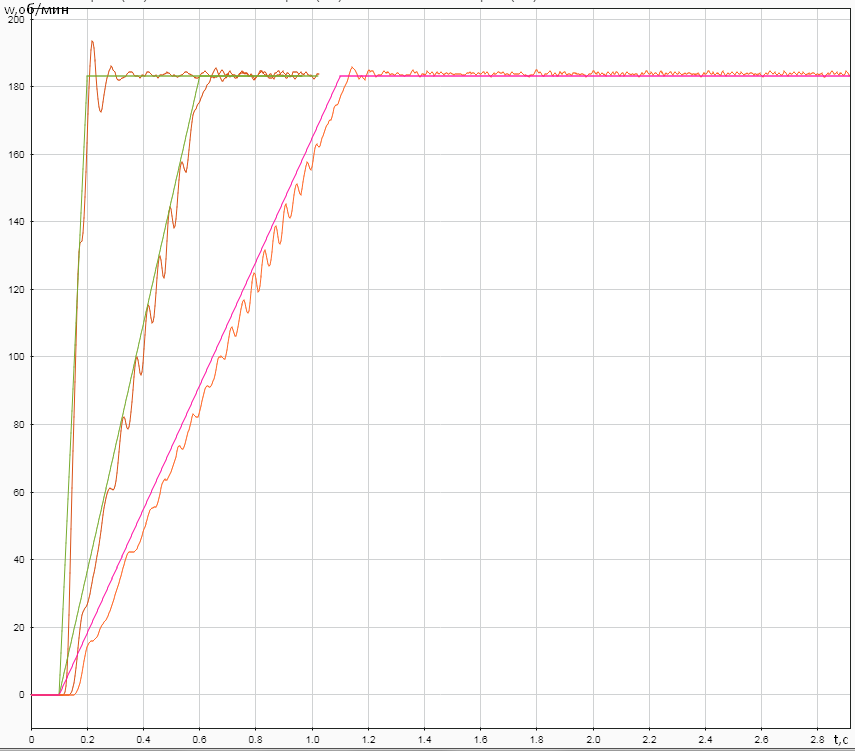
Рисунок 47 – График скорости при разгоне и торможение двигателя.

На представленном выше графике показана работа нейронной сети и регулятора настроенного на данный двигатель. Разгон происходит за одну секунду в промежутке времени 0.1 и 1, далее двигатель 0.4 секунды вращается на номинальных оборотах и в 1.5 секунде происходит торможение.

По графику можно сделать вывод, что нейронная сеть разгоняет двигатель лучше, чем ПИД регулятор без колебаний и за меньшее время происходит торможение. За представленный промежуток времени нейронная сеть выходит на заданное значение скорости, а ПИД регулятор не успевает за отведенное время даже выйти на номинальную скорость.

При обучение нейронной сети в обучающей выборки двигатель разгонялся до номинальных оборотов за две секунды, нейронная сеть не была обучена на быстрый разгон. Именно поэтому в следующем эксперименте необходимо проверить, как нейронная сеть будет управлять двигателем, если управляющее воздействие подавать линейно и выходить на номинальные обороты за различные промежутки времени.

На рисунке ниже представлены графики при данном эксперименте.



3

2

1

Рисунок 48 – Графики скорости при различных задающих воздействиях

На рисунке выше представлены три графика скорости вращения ротора двигателя, имеющего параметры третьего двигатели из таблицы 5, при различных изменяющихся скоростях нарастания задающего воздействия. Графики имеют следующие обозначения:

1 – задающее воздействие достигло номинального значения за 0.1 секунду;

2 – задающее воздействие достигло номинального значения за 0.5 секунду;

3 – задающее воздействие достигло номинального значения за 1секунду.

По полученным выше графикам скорости можно сделать вывод, что нейронная сеть правильно управляет двигателем с различной скоростью разгона. На первом графике при разгоне двигатель меньше всего колеблется по скорости, но и имеет самую большую перемодуляцию, что является ожидаемым результатом эксперимента, потому что значение задающего воздействия подаётся практически скачком. На втором и третьем графике скорости вращения двигателя практически не отличаются. Во время разгона обе скорости немного колеблются и выходят на номинальное значение за короткое время и с небольшое перемодуляцией.

Нейронная сеть в обучающей выборке обучалась на работу с двигателем, который нагружался вентиляторной нагрузкой. Из этого следует, что нейронная сеть не обучена на различные типы нагрузки. Именно поэтому следующим экспериментом нагрузка на двигатель, на который управляющее воздействие подаётся от нейронной сети, будет изменен тип нагрузки с вентиляторной на мгновенную подачу номинальной нагрузки.

На рисунке 50 представлен график скорости двигателя при данном эксперименте. Для наглядности также представлен график скорости двигателя, но управляемого ПИД регулятором. В промежутке времени с 0.1 секунды до 0.8 секунды двигатели разгоняются до номинальных оборотов без нагрузки. В момент времени равному 0.8 секунде происходит скачкообразное изменение нагрузки с 0 до номинального значения, для выбранного двигателя данное значение равно 79.9 Н\*м.

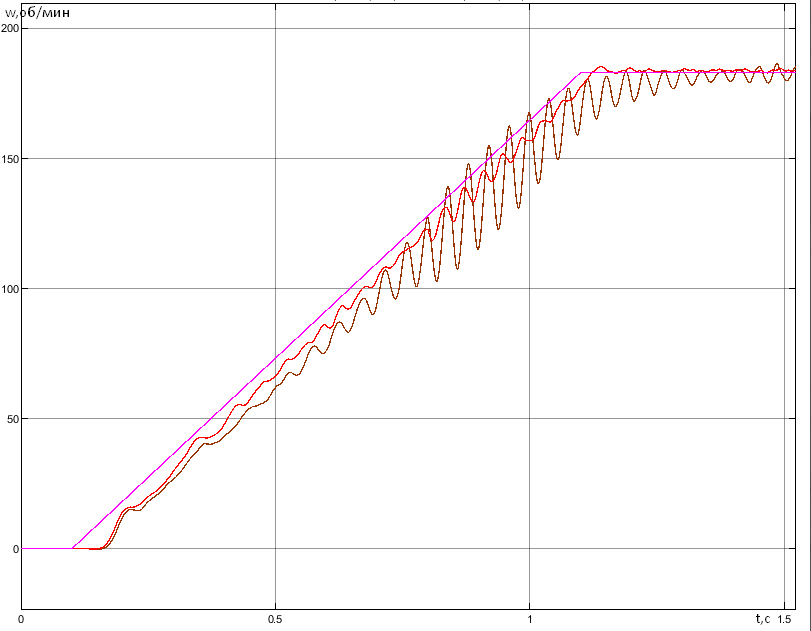


Рисунок 49 – график скорости при мгновенном изменение нагрузки на 0.8 секунде.

На полученном выше графике можно увидеть, что двигатель, управляемый ПИД регулятором, очень сильно колеблется во время разгона и сохраняет эту тенденцию и после того как подалась нагрузка на двигатель. Такой график скорости можно объяснить тем, что ПИД регулятор был настроен на работу с вентиляторной нагрузкой. Для того, чтобы регулятор правильно отрабатывал скачкообразную нагрузку необходимо его заново настраивать.

Двигатель, управляемый нейронной сетью в свою очередь намного лучше разгоняется, его не так сильно болтает, как двигатель, управляемый ПИД регулятором, болтание скорости происходит, только во время 0.8, то есть во время появления номинальной нагрузки.

На рисунке ниже представлены график тока в статоре и график момента двигателя управляемые нейронной сетью.

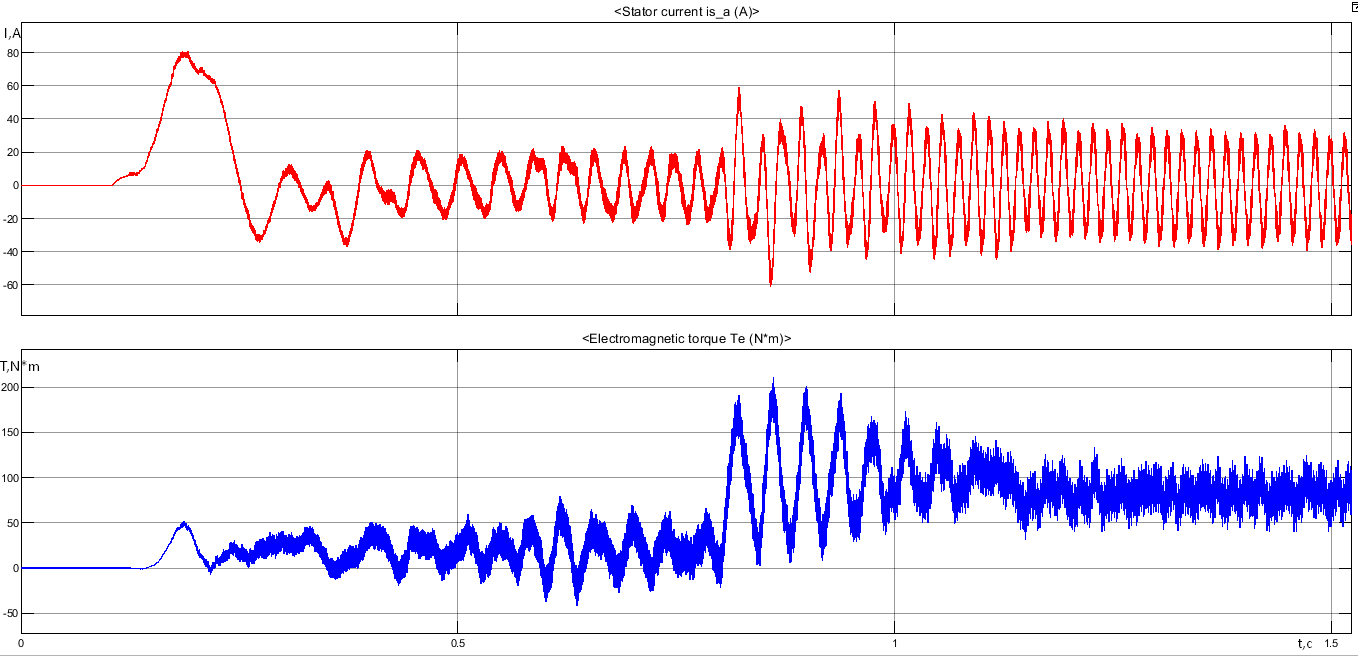


Рисунок 50 – График токи и график момента на двигатели управляемый нейронной сетью

По представленным выше графикам тока и момента видно, что нейронная сеть отрабатывает изменение нагрузки. По графику тока видно, что в момент времени 0.8 повещается значение тока до 60А, потом за 0.3 секунды система нормализуется и значение тока больше так сильно не колеблется.

По графику момента, видно, как в момент времени 0.8 нагрузка резко повышается, но нейронная сеть все рано справляется с нагрузкой не дает двигателю затормозить и выравнивает его скорость.

По графикам на рисунках 50 и 51 можно сделать вывод, что двигатель, управляемый нейронной сетью, справляется нагрузкой, на которую не происходило обучений. Для другого типа нагрузки необходимо перенастраивать ПИД регулятор, чтобы он правильно отрабатывал, а нейронная сеть сама сумела подстроиться, что показывает её адаптивность не только под различные типы двигателей, но и под различные нагрузки. Чем не может похвастаться обычный ПИД регулятор, который при даже небольшом изменение в управляющем объекте необходимо заново перенастраивать, а это занимает время и ресурсы.

Заключительным опытом является модуляция, реализующая режим аварии, а именно срыв крепления вала двигателя от клетчатки вентилятора и соответственно резкого снижение нагрузки на двигатель: при номинальной частоте вращения, с двигателя практически полностью снимается нагрузка. Задачей в данном опыте является проследить, что бы двигатель восстановил нормальный режим работы, не переходя в аварийный режим и смог затормозить за минимальное время. Зависимости момента и скорости вращения от времени, полученные в ходе опыта представлены на рисунке 52

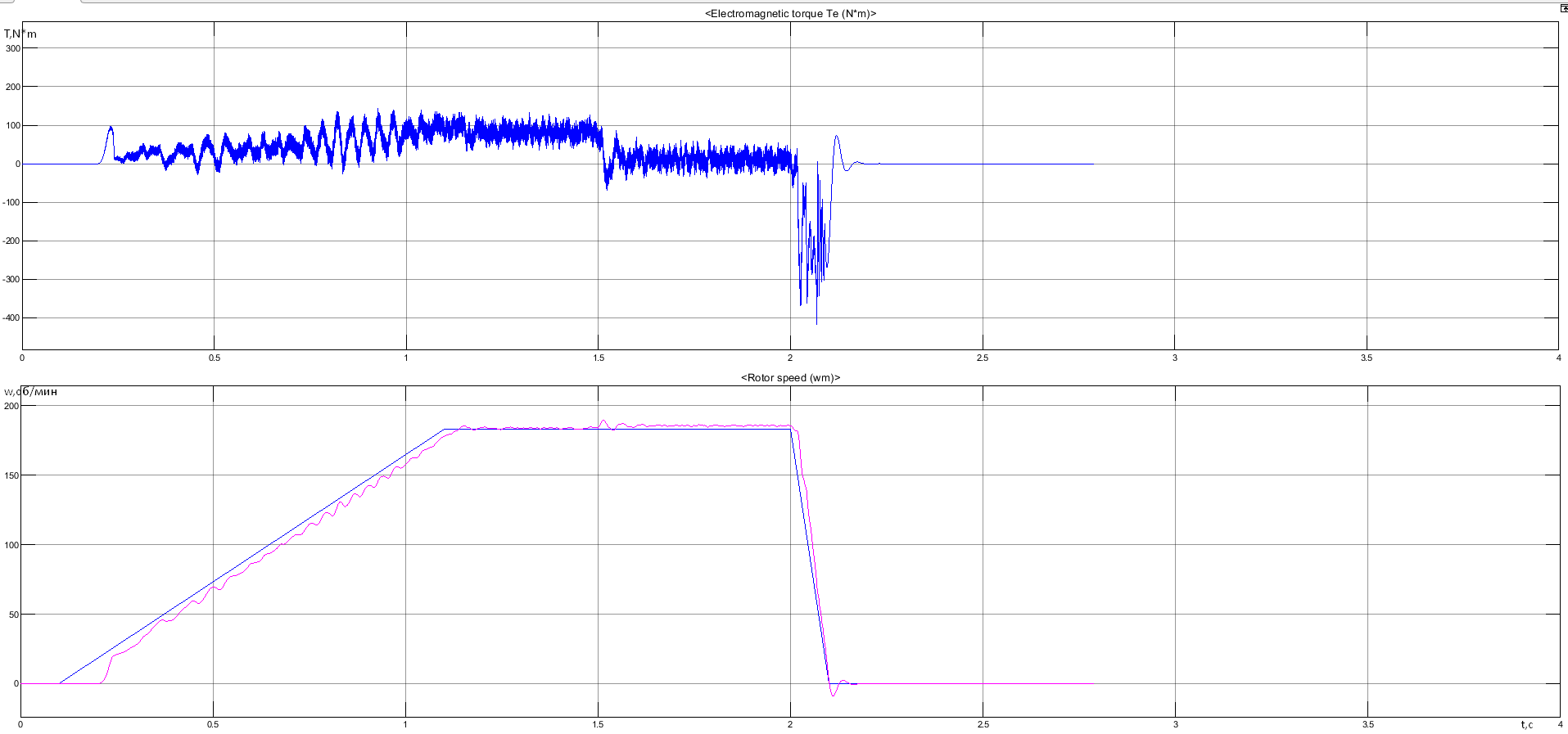


Рисунок 51 – Графики скорости и момента при модуляции аварийного режима работы двигателя

По представленным выше графикам можно сделать вывод, что с аварийным режимом работы двигателя нейронная сеть тоже справляется. Во время резкого снятия нагрузки двигатель не перешел в аварийный режим, а восстановил нормальный режим работы. И потом без нагрузки нормально затормозил с не большим перерегулирование.

7. Специальные вопросы обеспечения безопасности

Поскольку большая часть ВКР выполнялась с помощью персональной электронно-вычислительной машины (ПЭВМ), то в связи с этим тема дополнительной главы – аспект эргономики программного обеспечения (ПО).

Эргономика – это область науки, которая изучает человека и его деятельность в условиях производства с целью совершенствования орудий, условий и процесса труда. Объектом изучения эргономики является система «человек - машина». Предметом изучения эргономики является взаимодействие оператора с техническими средствами (машинами). Общей целью эр-гномики является обеспечение комфортных условий для эффективной работы человека, эффективное функционирование систем «человек — машина». Эргономичность взаимодействия человека и техники предусматривает единство свойств как

* Управляемость;
* Обслуживаемость;
* Освояемость;
* Обитаемость.

Для большинства пользователей именно интерфейс ПО ассоциируется с программным приложением (у пользователей впечатление от работы с программным приложением формируется зачастую непосредственно от работы с пользовательским интерфейсом). Поэтому все большее количество разработчиков программных приложений учитывают вопросы эргономики и применимости пользовательских интерфейсов и удобства работы. Это означает, что пользователи могут быстро и легко выполнять поставленные задачи, не обременяя себя долгим изучением интерфейса ПО. Учет применимости и эргономики в жизненном цикле программных приложений имеет следующие последствия:

* Увеличение скорости работы и удовлетворенности пользователей;
* Уменьшение расходов на эксплуатацию программных приложений;
* Уменьшение расходов на развитие ПО
* Уменьшение времени и расходов на обучение пользователей;
* Увеличение продаж ПО

Эргономичный пользовательский интерфейс должен удовлетворять следующим требованиям:

* Способствовать быстрому освоению ПО пользователем;
* Обеспечивать ввод информации пользователем наиболее удобным для него способом, не заботясь о ходе вычислений;
* Обеспечивать согласование ПО и пользователя, т.е. информация должна быть понятной пользователю, объем представляемой информации не должен быть избыточным;
* Обеспечивать интуитивное управление ПО пользователем;
* На протяжении всего времени работы информационной системы пользовательский интерфейс должен находиться под контролем пользователя, при этом никакие его действия не должны приводить к прерыванию работы ПО;
* Обеспечивать исправление ошибок оператора при вводе исходных данных, либо указывать на ошибки при отсутствии возможности их исправления;
* Обеспечивать обратную связь пользователя с ПО (справочная подсистема должна обеспечивать пользователя информацией, которая позволит настраивать работу с диалоговыми окнами, идентифицировать и устранять ошибки и определять порядок дальнейшей работы) [16].

В данной работе для разработки и проверки работоспособности нейронных сетей использовалось следующее ПО:

1. Matlab Simulink – для проверки работы нейронной сети в составе комплекса устройств.
2. PyCharm – для обучения и построения нейронных сетей на языке Python с использованием библиотеки Keras.

Оценим требования к интерфейсам данных ПО на основании ГОСТ Р ИСО 9241-110-2016[17].

## 7.1 Оценка Интерфейса Matlab Simulink

На рисунке 52 представлен скриншот интерфейса системы MATLAB. Система позволяет сформировать оконное расположение самому пользователю, что говорит о том, что каждый пользователь сможет подстроить программу под себя.

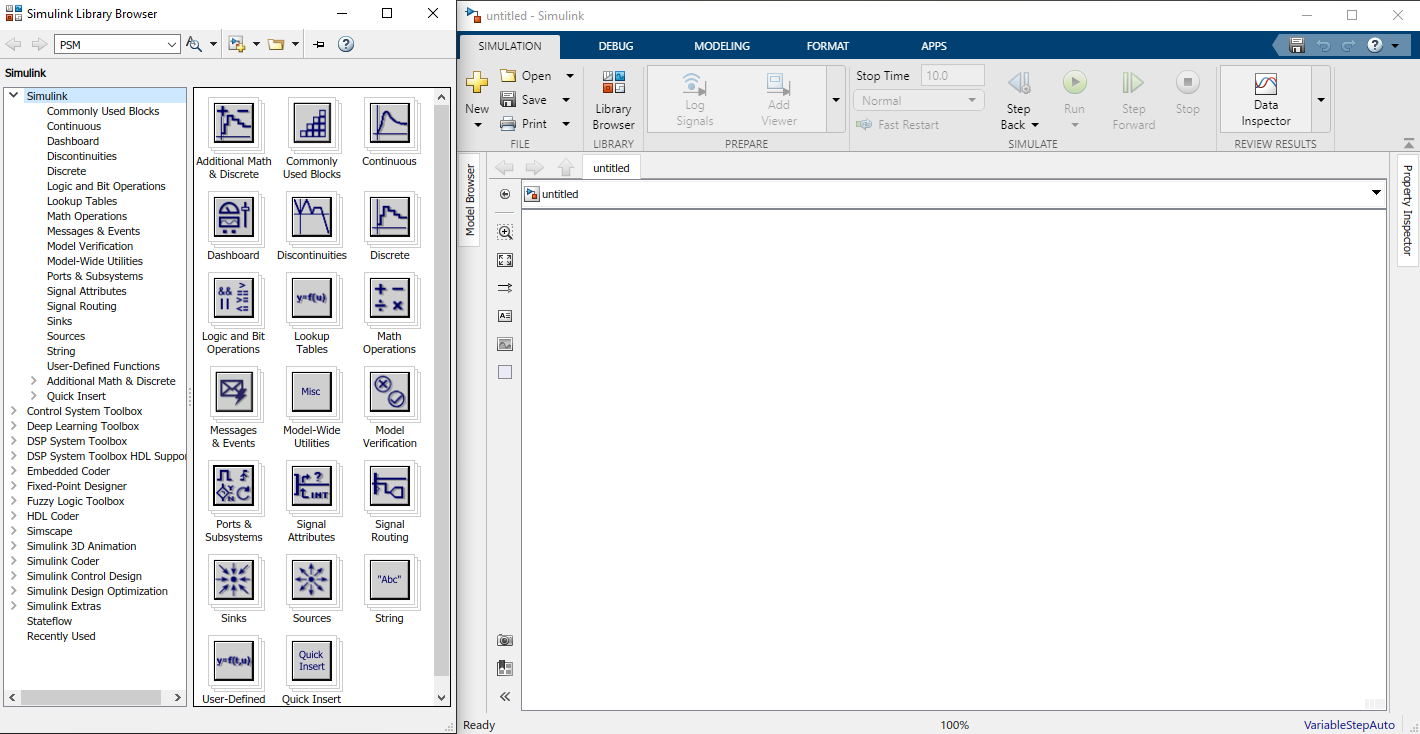


Рисунок 52. Интерфейс Matlab Simulink

В случае допущения пользователем ошибки в построение модели или недопустимого выбора параметров, система в ходе компиляции оповещает его о наличии данной ошибки, выделяя её и поясняя суть ошибки, предоставляя пользователю варианты для исправления ошибки самому.

Библиотека стандартных блоков, из которых происходит построение моделей, открывается отдельным окном, не препятствуя работе с самой программой. Также в библиотеке имеется поиск, что упрощает её использование. Библиотека остаётся в доступе пользователя до тех пор, пока он его не закроет, как и любое возможное окно либо рабочая область системы. Каждая рабочая область несёт свою смысловую нагрузку. Если учесть, что пользователь сам выбирает, какие рабочие поля ему нужны, то избыточности интерфейс не несёт, не перегружая оперативную память пользователя и его внимание.

Интерфейс системы дружественный, интуитивно понятный, что упрощает диалог «пользователь-система», позволяя за минимальное время приспособиться к интерфейсу данного ПО. В руководстве описан весь функционал системы, дана справка по каждой встроенной в систему функции и библиотечном блокам. В руководстве также приведены демонстрационные программы, раскрывающие широкие возможности ПО.

## 7.2. Оценка интерфейса PyCharm

Данная среда программирования довольна проста в использование и имеет широкие функциональные возможности, такие как: возможность написания кода на разных языках программирования, возможность разработки приложений для большого количества разнообразных платформ, возможность быстрого ввода и редактирования данных, возможность отладки кода программы, возможность навигации по коду, удобный и понятный пользовательский интерфейс с наглядным представлением информации и многие другие. PyCharm имеет как большое количество встроенных библиотек, так и возможность подключения других сторонних библиотек или плагинов, во время работы над данной ВКР при разработке кода использовалась сторонняя библиотека Keras, которая является в свою очередь надстройкой API для другой библиотеке TensorFlow, для составления и работы с нейронными сетями, что позволило использовать данную среду для конкретного задания. Загрузка и установка сторонних библиотек в данной среде разработки намного понятнее и удобнее, чем через консоль и прописыванием системной команды Pythonа PIP. На рисунке 53 изображено окно загрузки библиотек.

На рисунке справа представлено окно уже установленных библиотек, что позволяет пользователю своевременно отслеживать установление библиотеки и знать какими он имеет возможность использовать в данном проекте. С левой части рисунка представлено, окно поиска библиотек. Поиск происходит на серверах Python, что дает возможность выбирать из всех доступных и сертифицированных библиотек.

Так же рядом можно прочитать краткую информацию о библиотеке, о её версии, о авторе, а также есть возможность перейти на сайт библиотеки и посмотреть примеры, и описания всех функций.

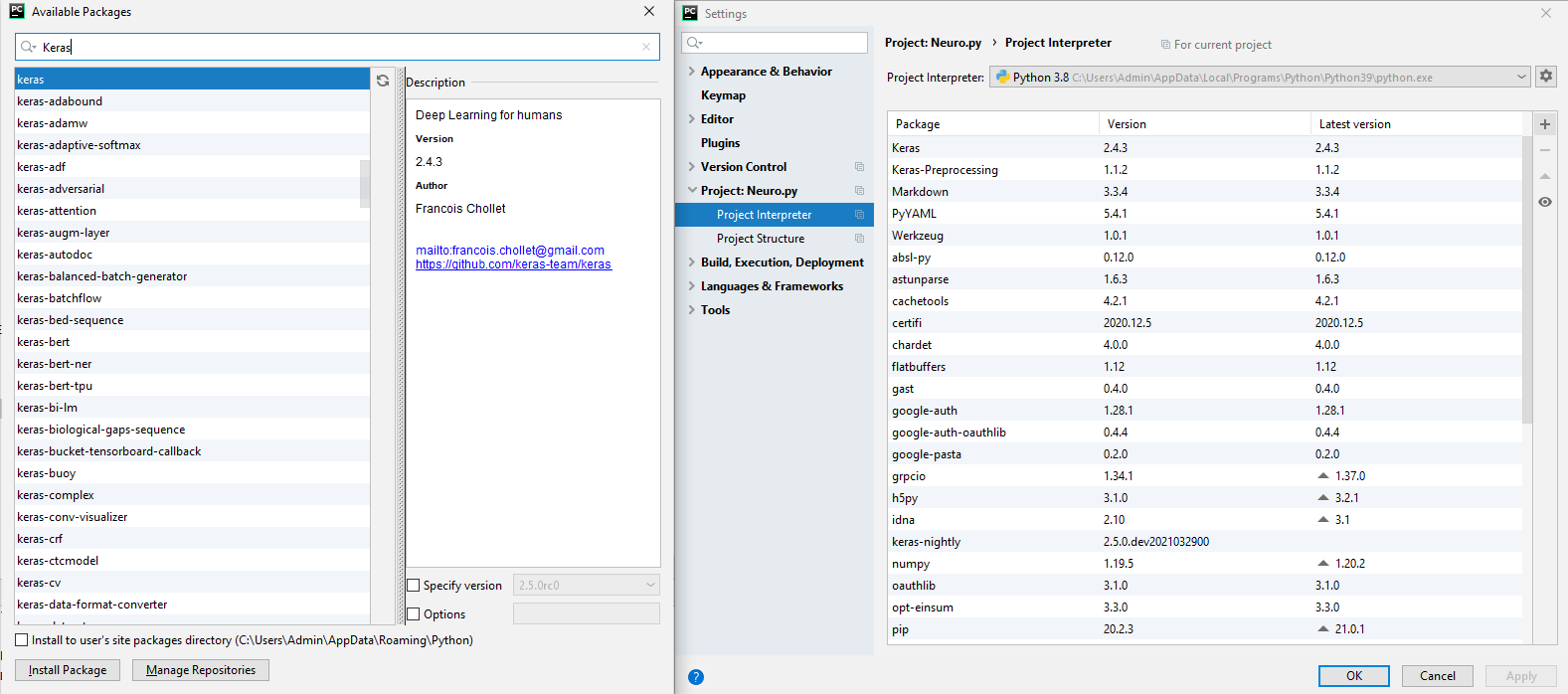


Рисунок . Установка библиотек.

Основной интерфейс PyCharm изображен на рисунке 54. Он разбит на блоки, где главный из них – это блок рабочей области для редактирования исходного кода, а остальные блоки (навигации по проекту, вывод, список ошибок и другие) являются дополнительными, которые могут настраиваться пользователем (удаляться, масштабироваться, менять место расположения и т.д.).

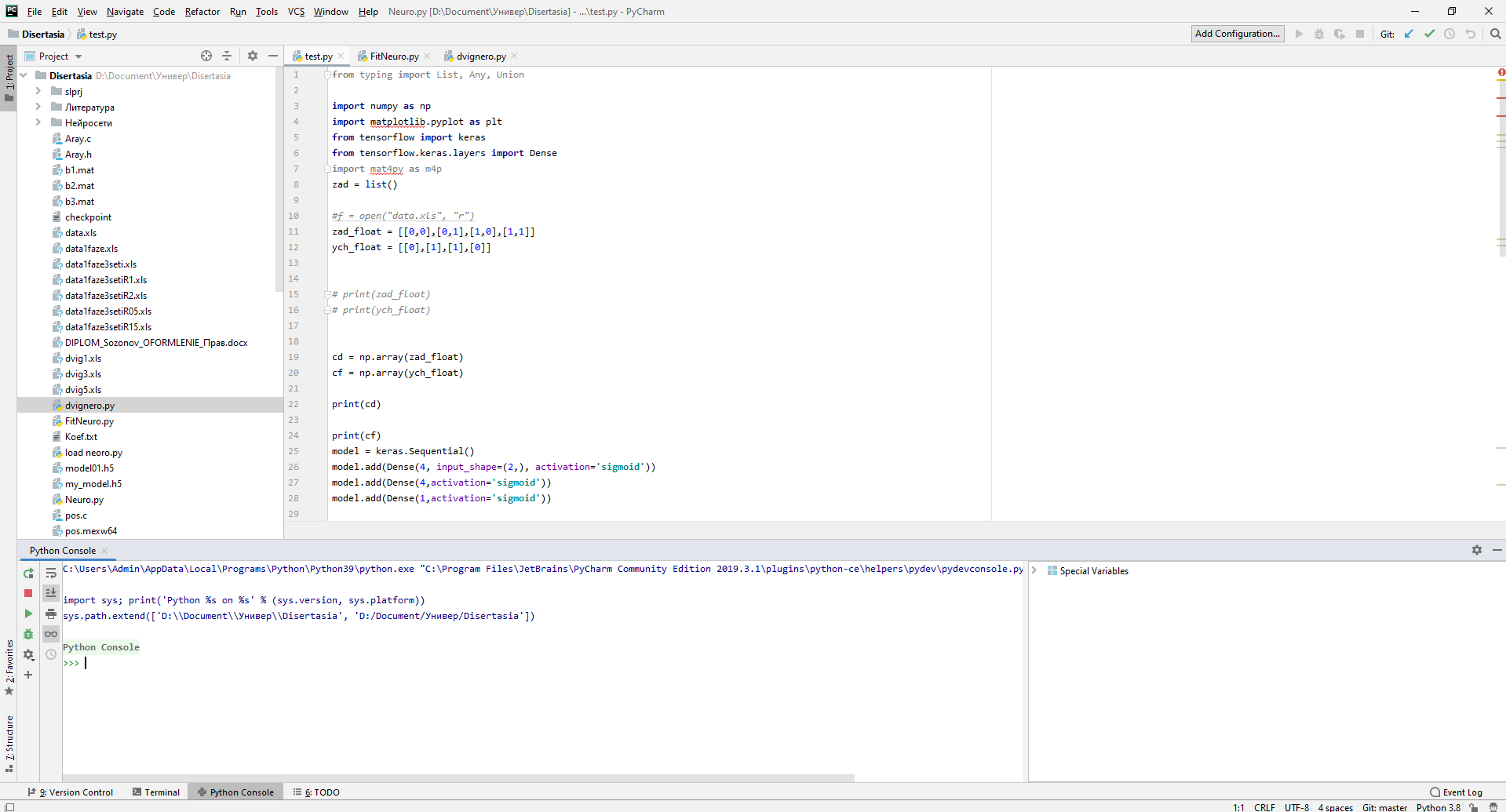


Рисунок . Основной интерфейс программы

Таким образом мы видим, что данная среда справляется с обеспечением организации диалога для производственного задания. Пользователь работает напрямую с диалоговым окнам разработки и редактирования, совершая при этом минимальные действия.

Информированность представляет собой способность пользователя в любое время работы понимать, в каком диалоге он находится, и какие действия он может предпринять. При работе с PyCharm не возникает проблем по поиску необходимого диалоговые окна, они все обозначены и переключаться между ними можно по вкладкам над рабочей областью (рисунок 55).



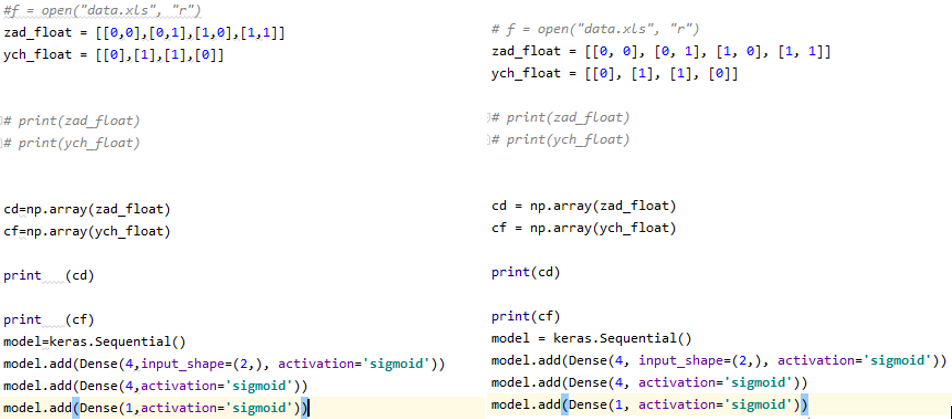
Рисунок . Оконные вкладки

В самом низу под блоком список ошибок можно видеть строку текущего состояния (рисунок 43), в которой располагается информация о текущем положении курсора, готовности кода к отладке, типе кодировке, выбранном стиле табуляции, название проекта и так далее. Эти параметры несут основную информацию о текущем проекте, при этом не отвлекают внимание и при работе над кодом.

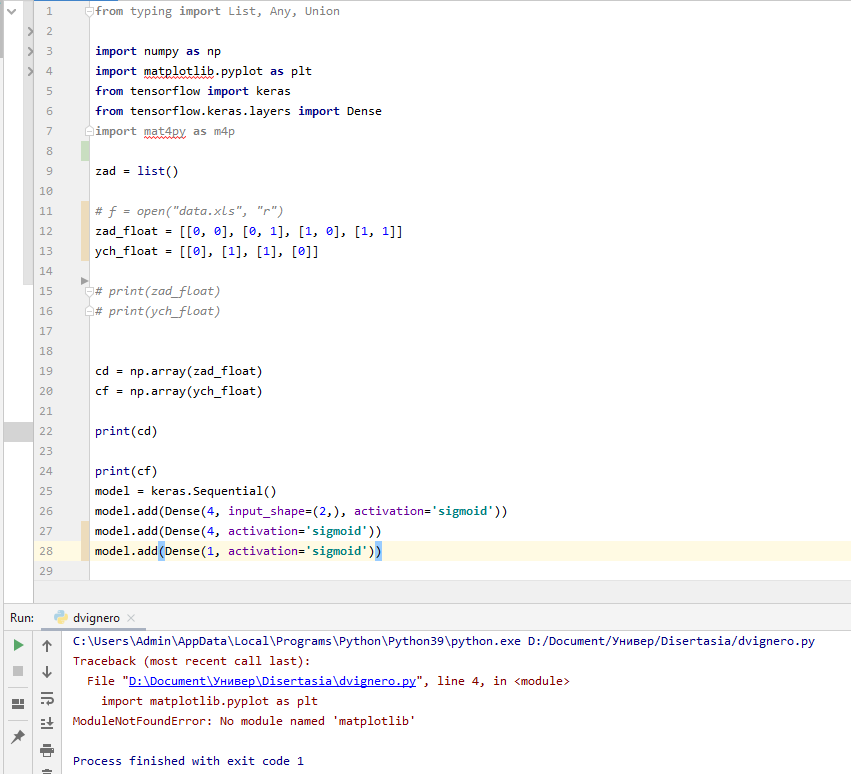


Рисунок . Строка состояния

Таким образом, видим, что с точки зрения информативности среда PyCharm является достаточно удобной для пользователя, интерфейс не перегружен лишней информацией и достаточно прост в обращении.



ТУТ нужно написать о ПЕП-8 то что пишет код в едином принятом стиле



ПРИМЕР ОШИБОК И ИХ ВИЗУАЛИЗАЦИИ.

## 7.3. Оценка ПО относительно ГОСТ Р ИСО 9241-110-2016

Диалог является адаптируемым, если пользователь может вносить изменения в формат взаимодействия с системой и формы представления информации [17].

В основном окне рассматриваемых ПО пользователь может изменять количество нужных внутренних окон, их положение и их масштаб по своему усмотрению. Также существует возможность создавать новые окна или изменять уже существующие используя макеты окон. Кроме того, есть возможность изменитьцветовую тему визуального представления интерфейса.

Так как основная задача завязана на написании кода программ и составление математических моделей, у пользователя есть возможность выбирать как ему написать ту или иную часть кода, а также составлять модель в Matlab с помощью библиотечных блоков или с помощью кода. Кроме того, пользователь может задать тип выходных данных (в графике или таблице, визображении или текстовом виде, и т.д.).

Таким образом пользователю предоставляется возможность менять как интерфейс данных среды для удобной работы, так и весь код разрабатываемой программы.

Проанализировав все пункты можно сделать следующие выводы, что интегрированная среда разработки PyCharm и пакет прикладных программ Matlab. удовлетворяет требованиям ГОСТ Р ИСО 9241-110-2016, и является пригодным для решения технических задач, связанных с разработкой кода программ на языке Python и составлением математических моделей и систем. При использовании данных сред программирования пользователь чувствует себя комфортно и не испытывает недопонимания программы. Весь интерфейс интуитивно понятен и не отвлекает пользователя от поставленной задачи.

Заключение

С развитием современной техники и усложнением систем управления все острее стоит вопрос о адаптивности систем управления, что бы одну общую структуру управления можно было бы применять с различными объектами. Именно с проблемой адаптивности неплохо справляться нейронные сети глубокого обучения.

В соответствии с поставленными задачами, в ходе данной разработки были исследованы различные архитектуры нейронных сетей, данные для их обучения и сделан анализ совокупности этих факторов на обучаемость и работоспособность сетей для выполнения задач. Так же было проведено исследование возможности нейронных сетей в замене собой различных регуляторов (регуляторов тока, напряжения, скорости). Лучше всего нейронная сеть показала себя, как регулятор скорости по причине того, что динамический процесс был длительнее, чем при работе нейронной сети в роли регулятора напряжения или регулятора тока.

Сбор обучающих данных нейронной сети важная задача от которой зависит корректность работы всей системы. Обучающая выборка должна содержать в себе моменты близкие к критическим точка, а в отдельных случаях и сами эти точки, по причине того, что нейронной сети необходимо знать, какую модель поведения выбирать в критических ситуациях. При подаче параметров, которые не входили в обучающую выборку на вход нейронной сети, она может повести себя непредсказуемо. Нейронная сеть представляет из себя черный ящик и без специализированного ПО сложно предсказать то, какой результат будет являться следствием её работы.

В ходе выполнения работы был разработан программный код на языке Python с использованием библиотеки Keras. Данный код достаточно универсален и имеет возможность быстрой настройки и обучения нейронных сетей. Так же код имеет возможность удобной и быстрой передаче весовых коэффициентов нейронов в модель, собранную в Matlab Simulink.

Как уже говорилось ранее, в ходе работы была разработана и протестирована модель нейронной сети, которая настраивается при помощи весовых коэффициентов нейронов и может быть использована для тестирования и работы с различными объектами управления, которые можно проектировать в Matlab Simulink.

Практическая значимость данной работы состоит в том, что при использовании нейронных сетей в управлении различными объектами появляется возможность обучение нейронной сети на диапазон объектов. Что в свою очередь дает возможность быстрее и без особых усилий вводить новые объекты в работу, которые входят в диапазон обучения нейронной сети.

В дальнейшем хотелось бы проверить результаты данного исследования на реальных объектах, в ходе работы реального объекта, могут возникнуть ситуации, которые сложно предположить изначально, это станет хорошим опытом для дальнейших разработок и сбора данных для обучающих выборок.

Так же для внедрения данной системы в реальные объекты управления необходимо провести исследование и решить, то какие аппаратные средства использовать для встраивания нейронных сетей в системы управления, поскольку вычисления нейронных сетей требуют матричного умножения, которое тратит большое количество ресурсов вычислительного устройства. А поскольку системы управления работают на прерываниях в реальном времени, факт траты ресурсов на нейронные сети необходимо учитывать и создать для этого благоприятные условия, либо искать способы ухода от матричного умножения используя заранее заготовленные таблицы, хранящиеся во внешней памяти устройств.

# Список литературы

1. Lin F.-J., Wai R.-J., Hong C.-M. “Hybrid supervisory control using recurrent fuzzy neural network for tracking periodic inputs”, IEEE Trans. on Neural Networks. Vol. 12, no. 1 2001, Pp. 69–90.

2. Soshi M., Yu S., Ishii S. and Yamazaki K., " Development of a high torque–high power spindle system equipped with a synchronous motor for high performance cutting", CIRP Annals - Manufacturing Technology, Vol. 60, Issue 1, 2011, pp 399-402

3. Елисеев В. Л. Нейросетевое оптимальное управление движением мобильного робота // Доклады 10-го Всероссийского семинара “Нейроинформатика и ее приложения”. Красноярск: 2002. — октябрь.

4. Tariq Rashid, «Make Your Own Neural Network», ISBN 9781530826605, *March* 2016, 222 pages

5. Шолле Франсуа. Глубокое обучение на Python – СПБ: Питер, 2018ю – 400с

6. Синтез адаптивных нейросетевых регуляторов нелинейных динамических объектов URL : <https://www.math.spbu.ru/user/gran/sb1/terechov.pdf> (Дата обращение 04.05.2021)

7. Калачев Ю.Н. “Векторное регулирование” URL : <http://www.efo-power.ru/BROSHURES_CATALOGS/KALACHEV.pdf> (Дата обращения 06.04.2021)

8. Андрей Карпатский “Блог о различных структурах и методики применения рекуррентных нейронных сетей” URL : <http://karpathy.github.io/2015/05/21/rnn-effectiveness/>(Дата обращения 19.04.2021)

9. Jeffrey L.Elman Cognitive Science Volume 14, Issue 2, April–June 1990, Pages 179-211

10. Пишем нейронную сеть на Python с нуля URL: <https://proglib.io/p/pishem-neyroset-na-python-s-nulya-2020-10-07> (Дата обращения 21.02.2021)

11. Герман-Галкин, С.Г. MATLAB&Simulink. Проектирование мехатронных систем на ПК. С.Г. Герман-Галкин. – СПб.: КОРОНА-Век, 2008. – 368 с.

12. Рудаков, В.В. Асинхронный электропривод с векторным управлением / И. М. Столяров, В. А. Дартау. – Л.: Энергоатомиздат, Ленинг. отд. – 1992. – 296 с.

13. Новиков В.А., Савва С.В., Татаринцев Н.И. Электропривод в современных технологиях. Под ред. В.А. Новикова. Учебник для студентов вузов — М: Издательский центр «Академия», 2014. — 400 с.

14. Дацковский, Л. Х. Современное состояние и тенденции в асинхронном частотно-регулируемом электроприводе (краткий аналитический обзор) / Л. Х. Дацковский [и др.] // Электротехника. – 1996. – № 10. – С. 18-28.

15. Белов М.П., Носиров И.С., Белов А.М. Синтез нейросетевого регулятора двухмассовой электромеханической системы привода подачи токарного станка СПБГЭТУ ”ЛЭТИ” .2018.№8.С.70-76.

16. А. А. Попов. Эргономика пользовательских интерфейсов в информационных системах: Учебное пособие. М.: РУСАЙНС 2016.

17. ГОСТ Р ИСО 9241-110-2016 Эргономика взаимодействия человек-система. Часть 110. Принципы организации диалога. М.: Стандартинформ,2016. – 65 с.