МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное автономное образовательное учреждение высшего образования

«Южно-Уральский государственный университет» (национальный исследовательский университет)

Высшая школа электроники и компьютерных наук

Кафедра «Информационно-измерительная техника»

РАБОТА ПРОВЕРЕНА

Рецензент, доцент

_____/ Н.М. Япарова/

«_____» _____2020 г.

ДОПУСТИТЬ К ЗАЩИТЕ

Заведующий кафедрой, д.т.н., проф.

_____/ А.Л. Шестаков /

«_____» _____2020г.

Алгоритм коррекции динамической погрешности с помощью нейронных сетей

(наименование темы работы (проекта)

ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА

ЮУрГУ – 12.03.01.2020.308-609.ВКР

(код направления/специальности, год, номер студенческого)

Руководитель, доцент кафедры ИнИТ

_____/ А.С. Волосников/

«_____» _____2020 г.

Автор

студент группы КЭ – 225

_____/ В.С. Лисовский/

«_____» _____2020 г.

Нормоконтролер, доцент кафедры ИнИТ

_____/ А.С. Волосников /

«_____» _____2020 г.

Челябинск 2020

АННОТАЦИЯ

Лисовский В.С. Алгоритм коррекции динамической погрешности с помощью нейронных сетей – Челябинск: ЮУрГУ, КЭ-225; 2020, 115 с., 24 ил., библиогр. Список 31 – наим., 6 прил..

В выпускной квалификационной работе был разработан алгоритм коррекции динамической погрешности, с помощью которого получилось уменьшить погрешность динамических измерений в присутствии аддитивного шума. Была создана и обучена нейронная сеть, способная отфильтровать шумы и восстановить входной сигнал.

Были решены следующие задачи:

1. Информационный поиск источников. Изучены современные методы, алгоритмы и подходы восстановления входного сигнала. Выделены достоинства и недостатки существующих методов;
2. Были рассмотрены существующие нейронные сети, их структура и особенности. В качестве наиболее подходящей нейронной сети была выбрана динамическая нейронная сеть с обратной связью;
3. Разработан адаптивный алгоритм коррекции динамической погрешности с использованием нейронной сети;
4. Приведены результаты моделирования алгоритма, как при заданных сигналах, так и экспериментальных данных, полученных со стенда датчика температуры.

Наиболее качественно восстанавливается гармонический сигнал. Ступенчатый и импульсный сигналы также хорошо восстанавливаются, но всё же не так идеально как гармонический сигнал. Восстановление сигнала при случайном шуме сильно усложняет работу алгоритма. Из-за особенностей Белого Гауссового шума сигнал хуже фильтруется, чем при синусоидальном шуме.

ОГЛАВЛЕНИЕ

ВВЕДЕНИЕ

Зачастую при обработке результатов измерений человек сталкивается проблемой искажения, вызванного многими внутренними и внешними факторами, такими как шумы, перебои питания и так далее. При этом возникает погрешность, которая оказывает сильное влияние на качество сигнала. В таких случаях становится актуальна, задача коррекции погрешности. Но не всегда, получается, уменьшить погрешность в силу множества факторов.

Существует множество подходов, методов, алгоритмов коррекции подходящие только для конкретных случаев или условий, они не универсальны и во многих ситуациях качественно реализовать задуманное не получается. Для решения этой проблемы стали разрабатываться уже более новые и совершенные алгоритмы, они получили название адаптивных методов. Суть их заключается в обучении и подстройке к новым условиям, то есть необходимость человеку в трудной и кропотливой работе становится меньше, нагрузка снижается, а значит, растёт производительность.

Одним из способов коррекция погрешности, является восстановление входного сигнала с помощью выходного. Можно использовать методы, способствующие более точному восстановлению входного сигнала, но существуют и определенные трудности, и так просто ими воспользоваться не получается. Достаточно часто при работе с сигналами измерительной информации возникают искажения в данных, вызванные прерыванием питания, сбоями системы измерений и другими различными причинами. Это усложняет последующий анализ таких сигналов, что создает определенные сложности для определения интегральных количественных характеристик таких сигналов и для принятия обоснованных решений на их основе. Как правило, такой анализ возможен только после оцифровки сигнала измерительной информации. Невозможность корректного восприятия данных по причине их неполноты снижает их объективную ценность. Проблема восстановления искаженных

участков сигнала является актуальной задачей, поскольку позволяет получить хотя бы приближенные результаты в ситуациях, когда альтернативой является отсутствие каких-либо результатов вообще. Искажения и пропуски в данных могут возникнуть в задачах, когда сигнал измерительной информации подвергается сильным внешним воздействиям (например, шумы в полосе частот сигнала кратковременно обеспечивают неприемлемое соотношение «сигнал/шум»), когда носитель информации в автономно работающей системе подвержен воздействиям, приводящим к частичной потере данных. Существуют методы, позволяющие восстановить с некоторой точностью и достоверностью искажения непериодического сигнала. Для корректного восстановления данных требуется выполнение определенных условий или допущений о свойствах сигнала (например, таких как ограниченность полосы частот или положительность сигнала за все время наблюдений). Отличительной особенностью этих подходов является их итерационный характер и необходимость задания начального приближения, от удачного выбора которого зависит качество конечного результата. В той или иной ситуации разные методы могут оказаться более эффективными, чем другие.

Значительный вклад в разработку теории и алгоритмов коррекции динамической погрешности методом восстановления входного сигнала принадлежит как отечественным ученым Ю.Е. Воскобойникову, Д.Ю. Иосифову, Н.Е.Карповой, К.А. Королёвой, А.В. Меркушевой, Э.П. Тихонову, А.Л. Шетсакову В.А. Котельникову, так и зарубежным М. Маркусу, А. Популису, Л. Рабинеру, Б. Голду, А. Оппенгейму.

В настоящее время нейросетевые методы выходят на первый план, область их применения досочно сильно увеличилась в последние десять лет за счёт нелинейных алгоритмов обработки сигналов. Сейчас уже никого не удивишь использованием нейронных сетей в прогнозировании и в транспортной индустрии, в задачах восстановлении изображений и обработке цифровых

сигналов (ЦС). Существует множество программ, которые могут распознать или идентифицировать образ, основанных на нейронных сетях. У истоков разработки и дальнейшего прогресса нейронных сетей стоят такие научные деятели, как Т. Кохонен, Ф. Розенблатт, С. Гроссберг, М. Минский. В настоящее время многие российские ученые также активно работают в направлении НС одни из них В.И. Горбаченко, В.В. Золотарева, Н.И. Червякова, А.И. Галушкина, А.Н. Горбаня, А.Н. Балухто.

Нейронная сеть имеет множество преимуществ, которые достаточно сильно помогают при обработки ЦС. Можно добить высокой производительности при не самых высоких затратах, как по времени, так и по денежным вложениям. НС отлично подходят в тех ситуациях, когда информация присутствует в неполной форме. Также нейросети достаточно популярны в задачах обработки различного рода сигналов.

Основной целью работы является уменьшение погрешности динамических измерений путём адаптивного восстановления входного сигнала измерительного преобразователя в присутствии аддитивного шума на его выходе с использованием нейросетей.

Основной целью работы является разработка нейросетевого алгоритма для уменьшение погрешности динамических измерений путём адаптивного восстановления входного сигнала измерительного преобразователя в присутствии аддитивного шума на его выходе.

Для достижения указанной цели в диссертационной работе решаются следующие задачи:

- Рассмотреть существующие подходы, методы и алгоритмы восстановления измерительных сигналов, отметить их недостатки и достоинства;
- Изучить структуру и виды нейросетей, с целью найти наиболее подходящую для решения восстановления входного сигнала;

- Разработать алгоритм корректировки погрешности, используя нейронные сети;
- Реализовать метод коррекции в одном из математических пакетов и выполнить качественное сравнение эффективности применения метода при восстановлении конкретных измерительных сигналов.

Научная новизна работы:

Применение нейронных сетей как метода искусственного интеллекта в задачах оценки и коррекции погрешности динамических измерений.

Практическая ценность работы:

Путем дополнительной математической обработки выходного сигнала измерительного преобразователя, без изменения конструкции прибора уменьшить его динамическую погрешность.

Апробация:

Лисовский, В.С. Адаптивный алгоритм коррекции погрешности динамических измерений / В.С. Лисовский, А.С. Волосников // Проблемы получения, обработки и передачи измерительной информации: сб. науч. тр. – Уфа: РИК УГАТУ, 2019. – С. 370-375.

1. АНАЛИТИЧЕСКИЙ ОБЗОР МЕТОДОВ ВОССТАНОВЛЕНИЯ ИЗМЕРИТЕЛЬНЫХ СИГНАЛОВ

1.1Динамическая погрешность

Под понятием динамическая погрешность понимается – погрешность результата измерения, обусловленная условиями динамического измерения [5].

Согласно ГОСТ 8.009-84 [4] инструментальная составляющая погрешности измерений определяется лишь как сумма определенных составляющих, так для общего случая принято использовать четыре составляющие погрешности измерений, то есть четыре составляющие инструментальной составляющей погрешности измерений:

1) Основная погрешность средства измерения;

2) Дополнительная погрешность средства измерения;

3) Динамическая погрешность средства измерения;

4) Погрешность, обусловленная взаимодействием СИ и объекта измерений.

“Погрешность, обусловленная реакцией средства измерения на скорость или частоту изменения входного сигнала. Эта составляющая, определяющая динамическую погрешность измерений, зависит как от динамических свойств средства измерения, так и от частотного спектра входного сигнала, она называется динамической погрешностью средства измерения.”[4]

Следовательно, динамическим режимом измерения является режим измерения, где динамическая составляющая погрешности оказывает существенное влияние на общую погрешность при измерении [19, с. 1–9].

“Динамические характеристики, которые регламентированы в ГОСТ 8.009-84 [5], отражают динамические свойства средства измерения. Эти свойства принято представлять двумя характеристиками:

1) Временем установления показаний (выходного сигнала);

2) Амплитудно-частотной (редко фазово-частотной) характеристикой.”[4]

При расчете динамической погрешности первая характеристика не допускается при использовании, поскольку при помощи её возможно только время, нужное для осуществления единичного измерения.

Частотные характеристики выражаются как максимальные допустимые изменения амплитуды или фазы синусоидального выходного сигнала средства измерения при изменении частоты входного синусоидального сигнала, с постоянными амплитудой и фазой. Эти характеристики позволяют оценить максимальную динамическую составляющую погрешности измерений при граничных значениях частоты входного сигнала, но только если входной сигнал имеет синусоидальную форму. Поэтому частотные характеристики в таком виде являются лишь качественными характеристиками и для расчета динамической составляющей погрешности измерений непригодны.

В ГОСТ 8.009-84 [4] также предусмотрены другие характеристики, которые более полно отражают динамические свойства средства измерений. Также стоит отметить, что по этим динамическим характеристикам достаточно легко определить указанные выше характеристики, нормируемые традиционно. Предусмотрено нормирование таких динамических характеристик средств измерений, которые позволяют оценить искажения средств измерений любых изменяющихся сигналов, поступающих на вход – исследуемых при измерениях динамической составляющей погрешности измерений и статической погрешности средств измерений.

1.2 Классические методы

1.2.1 Метод интерполяции

Показан алгоритм полиномиальной интерполяции для АЦП с длительностью калибровки 1мс.

Решением проблемы сохранения данных со спутника является восстановление потерянных значений путем их интерполяции по ближайшим известным [8].

Суть метода заключается в изучении возможности интерполяции пропущенных значений в ходе калибровки данных полученных из сигнала, при помощи интерполяции многочлена третьего порядка и нахождение среднеквадратической ошибки интерполяции.

С точки зрения математики, задача восстановления аналоговых сигналов по его дискретным отсчётам, это всем известная задача интерполяции непрерывной функции F(x) по конечному числу N её точек. Допустим, что в координатах $x0, x1, \dots xi, \dots xN$, таких, что $a \leq x0 \leq x1 \leq \dots \leq xN \leq b$, известны значения функции y=f(x), то есть в промежутке [a, b] задана табличная функция. Эти координаты, в которых значение функции задано, называются узлами интерполяции.

Значения аппроксимирующих функций в самих узлах должны быть идентичны заданным значениям исходной функции. При интерполяции тригонометрических функций наиболее подходящим представлением будет аппроксимация гладкой функцией [9]. Такое решение даёт, например, метод полиномиальной интерполяции. Мысль метода заключается в том, что исходная функция аппроксимируется многочленом третьей степени, то есть функцией вида:

$$y=a_0+a_1x+a_2x_2+a_3x_3. \qquad (1.2.1.1)$$

При этом значение интерполирующей функции совпадает со значениями рассматриваемой функции в узлах интерполяции [9]. На основании данного условия составляется система линейных уравнений вида:

$$\begin{cases} a_0 + a_1 x_1 + a_2 x_1^2 + \ldots + a_{n+1} x_1^n = y_1 \\ a_0 + a_1 x_2 + a_2 x_2^2 + \ldots + a_{n+1} x_2^n = y_2 \\ a_0 + a_1 x_3 + a_2 x_3^2 + \ldots + a_{n+1} x_3^n = y_3 \\ \ldots \\ a_0 + a_1 x_{n+1} + a_2 x_{n+1}^2 + \ldots + a_{n+1} x_{n+1}^n = y_{n+1} \end{cases} . \quad (1.2.1.2)$$

При реализации интерполяционной процедуры коэффициенты полинома a_k находятся решением (1.2.1.2) каким-либо численным методом.

1.2.2 Алгоритмы на основе методов проекций на выпуклые множества

Суть алгоритмов на основе методов проекций на выпуклые множества заключается в следующем.[20] На восстанавливаемую функцию накладывается ряд ограничений S_i, каждое из которых представляет собой замкнутое выпуклое подмножество Гильбертова пространства. Тогда решение содержится в пересечении этих подмножеств:

$$S_0 = \bigcap_{i=1}^{n} S_i. \quad (1.2.2.1)$$

Замкнутым называется такое подмножество X, которое содержит все свои предельные точки . Выпуклым называется такое подмножество X, которое для любых двух элементов f, g ∈ X , содержит отрезок, соединяющий их:

$$\alpha f + (1- \alpha)g \in X, \quad (1.2.2.2)$$

где $0 \leq \alpha 1 \leq 1$. Задача восстановления сводится к построению оператора (композиции операторов ограничений), позволяющего найти решение, принадлежащее пересечению этих подмножеств S_0.

1.2.3 Итерационный метод

Ограничения, накладываемые на функции, представляют собой некоторые множества функций. Существует несколько наиболее часто применяемых операторов проектирования функций на множества ограничений [24]:

1) S_1 – множество функций, ограниченных по протяженности. Такие функции отличны от нуля на некотором пространстве $S_1 \subset \Omega$, где Ω – некоторое подмножество Гильбертова пространства. Тогда оператор ограничения имеет вид:

$$C_1 x = \begin{cases} x(t), & \text{если } t \in S_1 \\ 0, & \text{если } t \notin S_1 \end{cases}. \qquad (1.2.3.1)$$

2) S_2 – множество функций, для которых результат преобразования Фурье G(ω) попадает в замкнутую области θ Фурье-плоскости. Пусть x(t) – функция без пропусков. Оператор ограничения имеет вид:

$$C_2 x = F^{-}[H(\omega)], \qquad (1.2.3.2)$$

где F^- –обратное преобразование Фурье.

$$H(\omega) = \begin{cases} G(\omega), & \text{если } \omega \in \theta \\ X(\omega), & \text{если } \omega \notin \theta' \end{cases} \qquad (1.2.3.3)$$

где $X(\omega) = F^{+}[x(t)]$ прямое преобразование Фурье.

3) S_3 – множество, содержащее все неотрицательные функции в H с ограниченной энергией:

$$\int |x(t)|^2 \, dt \le E, x(t) \ge 0, \qquad (1.2.3.4)$$

$$C_3 x = \begin{cases} 0, & если \quad x_1 < 0 \\ x_1^+, & если \quad E_1^+ \le E, \\ \sqrt{\dfrac{E}{E_1^+}} x_1^+, & если \quad E_1^+ > E \end{cases} \qquad (1.2.3.5)$$

где $x_1^+ = \begin{cases} \mathrm{Re}(x) & \text{если} \quad x \ge 0 \\ 0, & \text{если} \quad x < 0 \end{cases}$, а энергия $E_1^+ = \int_{\Omega} (x_1^+)^2 dt$.

4) S_4 – множество всех неотрицательных функций x(t) в H. Принадлежность к этому множеству ограничивает область возможных значений функции. Проекция на это множество представляет собой следующее выражение:

$$C_4 x = \begin{cases} \mathrm{Re}(x), & \text{если } x \ge 0 \\ 0, & \text{если } x < 0 \end{cases}, \qquad (1.2.3.6)$$

5) S_5 – множество, содержащее вещественные функции в H, область значений которых ограничена. Интервал [a, b] является неотрицательным – a ≥ 0, b > a. Проекция на данное множество имеет вид:

$$C_5 x = \begin{cases} a, & \text{если } x(t) < a \\ x\ (t), & \text{если } a \le x(t) \le b. \\ b, & \text{елси } x(t) > b \end{cases} \qquad (1.2.3.7)$$

Представленные операторы и их комбинации могут быть использованы для восстановления сигналов. [23]

1.2.4 Алгоритм Гершберга-Папулиса

Итерационный алгоритм Гершберга-Папулиса (далее алгоритм ГП) может применяться для восстановления сигнала в том случае, если известна информация о восстанавливаемом сигнале и его спектре. Обязательным условием, выполнение которого требуется для возможности восстановления сигнала, является ограниченность спектра исходного сигнала. Информация о виде сигнала и его спектре должна быть описана ограничениями, наложенными на восстанавливаемый сигнал в виде множеств ограничений S_1 и S_2 (п. 1.2.3.1), а также дополнительным ограничением о конечности области протяженности [27]. Тогда с учетом (1.10) итерационный алгоритм для задачи восстановления может быть представлен в следующей компактной форме:

$$x_{k+1} = C_2 C_1 x_k. \qquad (1.2.4.1)$$

На рисунке 1 наглядно проиллюстрирована работа алгоритма ГП. Рисунок 1а показывает конфигурацию исходного сигнала (слева) и его преобразование Фурье (справа). Далее описано применение алгоритма к фрагменту исходного сигнала. Изначально к восстанавливаемой функции применяется оператор C_1. Затем вычисляется прямое преобразование Фурье от полученной величины. Производится ограничение (фильтрация) значения спектра полученной функции. Далее производится обратное преобразование Фурье. Фрагмент полученного

сигнала в области пропуска заменяется на исходный, после чего итерация повторяется. После некоторого количества итераций исходный сигнал восстанавливается с некоторой точностью.[27]

Кроме основного алгоритма ГП можно выделить некоторые его модификации: обобщенный алгоритм ГП, обобщенный релаксационный алгоритм ГП. Каждый из этих методов также может быть применен для восстановления искажений в цифровых сигналах.

1.2.4.1 Обобщённый алгоритм Гершберга-Папулиса

Обобщённый алгоритм ГП впервые был рассмотрен в [1.2.4]. Его суть заключается в следующем:

$$x_{k+1} = Cx_k, \qquad (1.2.4.2)$$

где $C = C_5 * C_2 * C_3 * C_1$ – композиция операторов проектирования, заданных в п.1.2.4.

1.1.4.2 Обобщённый релаксационный алгоритм Гершберга-Папулиса

Данный алгоритм является обобщением рассмотренных ранее алгоритмов и может быть определен следующим рекуррентным соотношением:

$$x_{k+1} = Tx_k, \qquad (1.2.4.3)$$

где $T = T5 * T2 * T3 * T1$ и $Ti = I + \lambda i(Ci - I), 0 < \lambda i < 2 (i = 1, \dots, 5)$. Параметры λ_i выбираются для улучшения скорости сходимости итерационного процесса. Несмотря на то, что T_i не являются операторами проектирования, они имеют те же неподвижные точки, что и C_i. Эти точки и являются решениями поставленной задачи (находятся в области пересечения (1.2.4.2)). [24]

1.2.5 Алгоритм Ховарда

В отличие от алгоритма ГП, где ограничивается полоса частот сигнала, в итерационном алгоритме Ховарда (Howard) [29] накладывается ограничение следующего вида:

$$Cx = \begin{cases} x(t), & \text{если } x(t) < 0 \\ 0, & \text{если } x(t) \geq 0 \end{cases}. \qquad (1.2.5.1)$$

Таким образом, алгоритм представляет собой следующую последовательность действий. Сначала восстанавливаемый сигнал подвергается преобразованию Фурье, затем применяется ограничение (1.2.4.3). Производится обратное преобразование Фурье. Фрагмент полученного сигнала заменяется на исходный, как в случае алгоритма ГП. Далее последовательность действий повторяется.

1.2.6 Восстановление сигнала по дискретным отсчетам Котельникова

Данное восстановление сигнала производится в соответствии с рядом Котельникова:[6]

$$\tilde{S}(t)=\sum_{k=-\infty}^{\infty}(k\Delta t)\frac{\sin\omega_в(t-kdt)}{\omega_в(t-kdt)}. \qquad (1.2.6.1)$$

Базисными функциями являются уже рассмотренные в первой главе функции отсчётов:

$$\varphi k(t) = \frac{sin\omega_в(t-k\Delta t)}{(t-k\Delta t)}. \qquad (1.2.6.2)$$

Если на вход идеального фильтра нижних частот с полосой пропускания 0... $\omega_в$, подать последовательность δ-функций (δ(t-kΔt)), умноженных на значения s(kΔt), то можно восстановить непрерывный сигнал в соответствии с (1.2.6.1). Но, ни сигнал в виде δ-функции, ни идеальный фильтр нижних частот физически не реализуемы. Вследствие чего, на в практических задачах взамен δ-функций применяют короткие импульсы, а вместо идеального фильтра – реально реализуемый фильтр нижних частот, что, естественно, приводит к погрешности восстановления. Так же, многие реальные сигналы конечны во времени и имеют неограниченный по частоте спектр, что также увеличивает погрешность восстановления. После прохождения дискретизированного сигнала s(kΔt) через

фильтр нижних частот, с частотой среза π/Δt, повторяющиеся реплики в спектре будут подавлены, а сигнал на выходе такого фильтра будет с точностью до постоянного множителя совпадать с исходным аналоговым сигналом s(t), что подтверждает справедливость выводов Котельникова.[6] Если же спектр непрерывного сигнала не ограничен диапазоном π/Δt, то соотношение между спектрами оказывается более сложным ввиду наложения спектров. В полосу пропускания фильтра будет попадать часть энергии от лепестков-реплик, и сигнал на выходе фильтра будет отличаться от исходного аналогового сигнала. Наложения можно избежать, повышая частоту дискретизации.

1.3 Методы ТАУ

1.3.1 Восстановление измерительного сигнала измерительных преобразователй тока электроэнергетических систем

На выобр алгоритма может повлиять несколько факторов, один из которых является метод обработки получаемой информации. Восстановление сигнала цифровым корректирующим устройстве осуществляется по завершению процесса регистрации или в темпе процесса. Последнее ощутимо ограничивает выбор, и из множества наборов алгоритмов применение выявляют только такие. которые имеют решение на основе упрощенного разностного уравнения. [3]

Для линейного одноступенчатого трансфарматора тока, описываемого системой уравнений 1.4.1.1, алгоритма цифровой коррекции в зависимости от метода интегрирования будут иметь вид:

$$\begin{aligned} i_1(jh) &= (1+\frac{L_2}{L_0})i_2(jh)+\frac{R_2 h}{L}\sum_{\mu=1}^{j} i_2(\mu h), \\ i_1(jh) &= (1+\frac{L_2}{L_0}+\frac{R_2 h}{2L_0})i_2(jh)+\frac{R_2 h}{2L_0}(i_2(0)/2+\sum_{\mu=1}^{j} i_2(\mu h),)), \\ i_1(jh) &= (1+\frac{L_2}{L_0}+\frac{R_2 h}{3L_0})i_2(jh)+\frac{R_2 h}{3L_0}(i_2(0)+\sum_{\mu=2,k=1}^{j=1} 4i_2(\mu h)+\sum_{\mu=2k}^{j=2} 2i_2(\mu h)), \end{aligned} \qquad (1.3.1.1)$$

где h — шаг квантования;

j — текущий номер дискретного значения сигнала;

μ =i,j.

Цифровому корректирующему устройству, реализующему приведенные алгоритмы, за время мемду очередными считываниями значения с аналого-цифрового преобразователя необходимо осуществить одну операцию умножения, два сложения и несколько, в зависимости от формулы интегрирования, операция сдвига. Каждый последующий алгоритм в (1.4.1.1) имеет меньшую» погрешность. При этом количество операция при переходе от одной формулы к другой существенно не возрастает. В работе приводятся сравнительные характеристики программ цифровой коррекции (ПЦК), реализующих различные методы построения алгоритмов. Показано, что в ряде случаев другие составляющие погрешности преобразования сигнала могэт значительно превышать методическую погрешность ПЦК.

Построение зависимости тока намагничивания от потокосиепления Ψ_0 для пик, работающих в темпе процесса, осуществлялось на базе таблично-аналитических методов. При этом исходная функция $i_0(\Psi_0)$ в рабочем диапазоне $[\Psi_{0min},\Psi_{0max}]$ аппроксимировалась функцией:

$$I_0(\psi_{0i}) = I_0(\psi_{0k}) + \Delta I_0(\psi_{0i}), \qquad (1.3.1.2)$$

Устойчивость при длительных режимах работы обеспечивается применением Для опроделекия потокосиепления рекурсивного цифрового фильтра, критерии выбора которого рассмотрены в предыдущей главе. [3]

Погрешности разработанных ПЦК исследовались с учетом порождающих факторов и особенностей проявления. При этом оценивались погрешности метода обработки, дискретизации сигнала по времени, смещения и квантования по уровню в АЦП. вычислительные погрешноси и погрешности квантования констант. Показано, что погрешность отображения входного сигнала ТТ в основном определялась точностью задания параметров схемы замещения ТТ в исходных данных для ПЦК.

1.3.2 Метод восстановления в скользящем режиме

В системах управления очень часто используются скользящие режимы. Благодаря обратным связям можно управлять системой в скользящем среднем. Существуют осложнения, которые мешают управлению динамическими характеристиками. И введение скользящего режима становится невозможным, поскольку измерительная система не имеет возможности охвата обратными связями первичного ИП.[2] Поэтому, разработка динамических моделей ИС на основе теории скользящих режимов и алгоритмов восстановления динамически искаженных сигналов с помощью этих методов является весьма актуальной. Разработка динамических моделей и алгоритма восстановления поспособствует значительному улучшению метрологических характеристик и качеству существующих дорогостоящих наземных испытательноизмерительных комплексов, а также повысит эффективность денежным и метериальных вложениий. Это будет достигнуто с помощью детальной и более полной математической обработки данных, полученных при измрении.

В измерительной системе в скользящем режиме с линейной частью превышающий второй порядок автоколебания, обычно, существуют. Устранить автоколебания возможно с помощью редукции первоначальной модели до второго порядка или структурным преобразованием системы на каскады, содержащие замкнутые нелинейные контуры с линейной частью, не превьппающей второго порядка.

В данном пункете рассмотрен метод редукции, позволяющий добиться соответствия в низкочастотной области первоначальной и редуцированной моделей. Принято, что исходная модель датчика описывается выражениями:

$$\dot{x} = Ax + Bu\ ,\ \ y = Cx + Du\ , \qquad (1.3.2.1)$$

где x − n-мерный вектор состояния датчика;

y − m−мерный вектор выходного сигнала датчика;

u − скалярное измеряемое входное воздействие;

A− матрица размерностью $n_{1x}n_2$;

B − матрица размерностью n_x1;

C − матрицы размерностью 1_x n;

D − нулевая матрица.

Редуцированная модель датчика рассматривалась в следующем виде:

$$x_r=A_rx_r+B_ru_r,\ y_r=C_rx_r+D_ru_r, \qquad (1.3.2.2)$$

где x_r - n_i-мерный вектор состояния редуцированной модели датчика ($n_i< n$),

y - m_i -мерный вектор вьпсодного сигнала исходной модели датчика;

u - скалярное измеряемое входное воздействи;

A_r - матрица размерностью n_i, x n_i;

B_r - матрица размерностью n_ix1;

C_r - матрица размерностью 1 x n_i;

D_r - нулевая матрица.

Редуцированные матрицы определены следующим образом:

$$A_r = LAT,\ B_r = LB,\ C_r = CT, \qquad (1.3.2.3)$$

где L и T матрицы преобразования размерностью n x i и i x n соответственно (i - порядок редуцированной системы).

Редуцированные модели датчиков, полученные этим методом, определяются некоторым выбором матриц L и T. В результате основным становится вопрос выбора этих матриц преобразования. В практике измерений необходимо соответствие частотных характеристик первоначальной и редуцированной модели в низкочастотной области, в которой расположен спектр измеряемого сигнала. Для этого в работе использован частотный момент:

$$M_i(w) = C(A-j\ wI)^{-1}\ B \qquad (1.3.2.4)$$

где $i \geq 1$, $0 \leq w < \infty$, и при w=0 достигается необходимое соответствие в низкочастотной области. Матрицы преобразования имеют следуюпщй вид:

$$L = \mathrm{Re}\begin{matrix}(A-jwI)^{-(i-1)}C\\(A-jwI)^{-(i-2)}C\\\vdots\\C\end{matrix} + \mathrm{Im}\begin{matrix}(A-jwI)^{-(i-1)}C\\(A-jwI)^{-(i-2)}C\\\vdots\\C\end{matrix} \qquad (1.3.2.5)$$

$$W = \mathrm{Re}[A^{-1}B \cdot A^{-(i-1)}B \cdots A^{-1}B] + \mathrm{Im}[A^{-1}B \cdot A^{-(i-1)}B \cdots A^{-1}B], T = W(LW)^{-1}, \qquad (1.4.2.6)$$

где W - промежуточная матрица преобразования размерностью i x n.

Данный метод является полностью работоспособным, поскольку влияние помех незначительно на результаты работы. Но по-прежнему остаётся разница между амплитудой измеряемым и восстановленным сигналом.[3] При существующем различии первоначального и редуцированного модуля, применение метода редукции для восстановления динамически искаженных сигналов, приводит к ощутимой погрешности. Чем больше порядок первичного ИП , тем пошрешность редуцированной системы увеличивается. Уменьшить динамическую погрешность и устранить автоколебания возможно, при использовании метода структурного преобразования, который позволит сохранить соответствие модели его датчика.

1.3.3 Применение методов восстановления сигналов в системах защиты информации

Для того чтобы, оценить информационную безопасность системы необходимо выполнить определённый порядок действий: [7]

- выбирают различные методы и средства измерения величины;
- формулируют определенные требования по достижению необходимого уровня безопасности;
- результаты измерений интерпретируются и определяется соответствие измеренных параметров необходимому уровню безопасности.

Чтобы восстановить входное значение x(t) по выходному сигналу y(t) и имеющейся информации об аппаратной функции (операторе), необходимо решить обратную задачу коррекции динамической погрешности. Необходимо ввести

корректирующее звено КЗ, передаточную функцию (ПФ) которая обратна передаточной функции ИП, обрабатывающее сигнал y(t) и формирующее сигнал z(t) на выходе КЗ. Если ИП описывается ПФ апериодического звена первого порядка с постоянной времени измерительного преобразователя T:

$$W(p) = \frac{y(p)}{x(p)} = \frac{K}{Tp + 1}, \qquad (1.3.3.1)$$

Разумеется, что при K = 1 ПФ корректирующего звена должна иметь следующий вид:

$$W^{-1}(p) = \frac{x(p)}{y(p)} = Tp + 1, \qquad (1.3.3.2)$$

Что в свою очередь соответствует дифференциальному уравнению, которое реализуется корректирующим звеном:

$$T\frac{dy(t)}{dt} + y(t) = z(t). \qquad (1.3.3.3)$$

Следовательно, этому звену необходимо реализовать функцию дифференцирования сигнала y(t) и сложения его производной с самим сигналом. В решения данной задачи необходимо воспользоваться цифровым фильтром, который реализует параболическую сплайн–аппроксимацию сигнала y(t), а также его первой производной по дискретным значениям сигнала y(t) [7]. При использовании параболической сплайн-аппроксимации, аппроксимируемый сигнал y(t) описывается параболической функцией на n-м дискретном участке:

$$y_n(t) = a_2[n]t^2 + a_1[n]t + a_0[n], \qquad (1.3.3.4)$$

где $a_2[n]$, $a_1[n]$, $a_0[n]$ — коэффициенты аппроксимации.

Для сплайн–аппроксимации первой производной сигнала y(t) справедливо аналогичное соотношение:

$$y_n'(t) = b_2[n]t^2 + b_1[n]t + b_0[n], \qquad (1.3.3.5)$$

где $b_2[n]$, $b_1[n]$, $b_0[n]$ — коэффициенты сплайн–аппроксимации первой производной сигнала y(t).

“Коэффициенты параболической сплайн–функции, аппроксимирующей сигнал y(t) и его производную, определяются с помощью весовой функции, которая определена на пяти дискретных отсчетах.”[8] Таким образом, сигнал на выходе корректирующего звена z(t) имеет вид:

$$z(t) = y(t) + Ty'(t). \qquad (1.3.3.6)$$

При t=0, то есть на границах интервалов дискретизации,

$$z(t)|_{t=0} = z_0[n] = a_0[n] + b_0[n]T. \qquad (1.3.3.7)$$

Формула (1.4.3.7) описывает сигнал z(t) на выходе КЗ. При отсутствии погрешности и самом наилучшем случае сигнал z(t) должен повторять очертания сигнала x(t). В реальном случае на выходе корректирующего фильтра сигнал z(t) формируется с задержкой в 2 дискретных интервала. Пропустим в качестве наглядного примера единичный сигнал Гауссовой формы через корректирующую цепь (рисунок 1.1).

$$x(t) = \exp\left[-\frac{(t-5)^2}{1.5}\right]. \qquad (1.3.3.8)$$

Согласно формулам (1.4.3.6), (1.4.3.7) и (1.4.3.8) выходной сигнал:

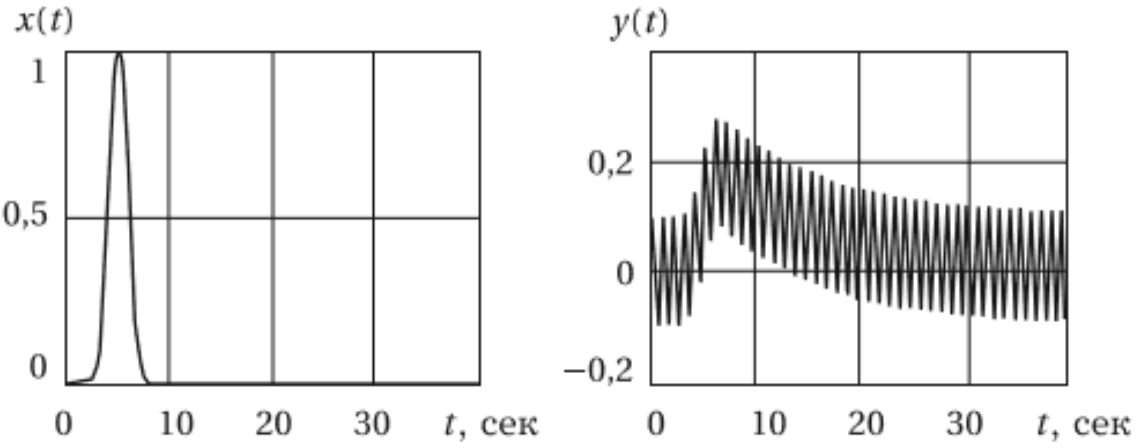


Рисунок 1.1 - Сигнал Гауссовой формы на входе ИП и сигнал на выходе ИП в сумме с помехой.

ИП определяется следующим образом:

$$y(t) = \frac{1}{T}\int_0^t \exp\left[\frac{-(t-\tau-5)^2}{1.5}\right]\exp(\frac{-\tau}{T})\cdot 1(t)d\tau. \qquad (1.3.3.9)$$

На сигнал полученный на рисунке 1.1, добавляется случайная аддитивная равномерно распределенная помеха. Тогда сигнал приобретает немного другой вид, показанный на рисунке 1.2. Попробуем оценить каким эффектом обладает корректирующий фильтр. Постоянная времени ИП составляет 10 секунд T=10 с, сигнал x(t) на входе измерительного преобразователя, выходной сигнал с измерительного преобразователя y(t) и сигнал на выходе корректирующего фильтра z(t) приведены на рисунке 1.2.

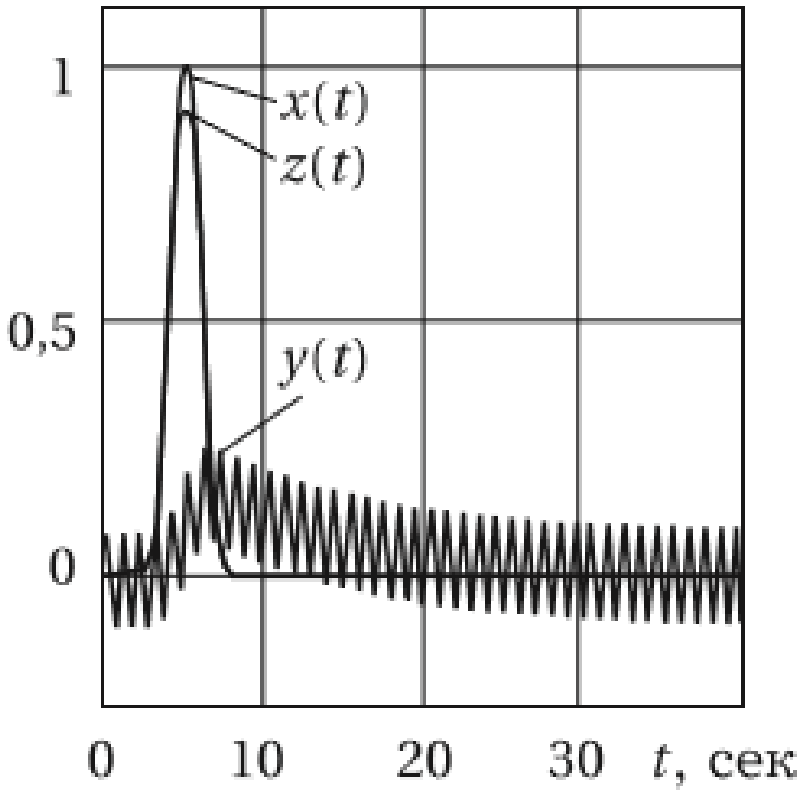


Рисунок 1.2 - Преобразование сигнала Гауссовой форм.

При рассмотрении рисунка 1.2 видно, что разработанный цифровой фильтр эффективно восстанавливает входной сигнал x(t) и подавляет случайный аддитивный равномерно распределенный шум.

1.4 Методы с использованием нейросетей

1.4.1 Рекуррентные сети

С помощью нейросетевого (НС) алгоритма [12], возможно, восстановить форму сигнала. Базовое формирование данного алгоритма было осуществлено Комоном, Джутен и Херолдом, так же был предложен общий подход к решению задачи восстановления и построен алгоритм для НС простой структуры. Но существуют у этого алгоритма и недостатки, один из которых устойчивость, она сопровождается нарушением при значительной неравномерности масштаба отдельных сигналов и/или при плохой обусловленности матрицы смешивания, а также имеет временные срывы даже в благоприятных ситуациях относительно объективных характеристик, которые могут осложнять решение задачи.

Анализ методической схемы приведен ниже и представлены две модификации алгоритма, одна из которых подстроена на нейронную сеть с прямым распространением сигнала, а другая — на рекуррентную НС (с обратными связями). Первый вариант алгоритма (частично) и второй (полностью) лишены упомянутых недостатков: они практически независимы от большой неравномерности масштабов восстанавливаемых сигналов и от степени сингулярности матрицы смешивания.[14]

Считается, что неизвестные сигналы $s_j(t)$, $j = 1,\ldots, n$, имеющие различные математические и физические модели, совместно воздействуя на датчики ИИС, порождают на их выходах измеряемые (системой) сигналы $x_1, \ldots, x_n$, которые представляют некоторую линейную комбинацию (для каждого датчика свою) первичных сигналов $s_j(t)$:

$$x_i(t) = \sum_{j=1}^{n} a_{ij}(t) s_j(t), \qquad (1.4.1.1)$$

Или

$$\underset{n}{x(t)} = \underset{nxn}{A} \cdot \underset{n}{s(t)}, \qquad (1.4.1.2)$$

где a_{ij} (i,j = 1,…,n) — коэффициенты линейных комбинаций; второе соотношение является векторно-матричной формой первого;

A — фиксированная или медленно меняющаяся матрица с определителем det A ≠ 0; $s(t) = [s_1(t),\ldots,s_n(t)]^T$; $x(t) = [x_1(t),\ldots,x_n(t)]^T$;

Размерности переменных указаны под формулой, и для простоты приняты одинаковые размерности сигналов s и x, а A – называется матрицей смешивания, и она неизвестна.

Восстановление состоит в получении сигналов $y_1(t),\ldots,y_n(t)$, или вектора $y(t) = [y_1(t),\ldots,y_n(t)]^T$, который в определенном смысле аппроксимирует сигнал s(t). Остаточная неполнота в решении задачи состоит в неопределенности масштабирования y(t) и порядка его компонент $\{y_j(t)\}$, в котором система оценивает $\{s_i(t)\}$. Эта неопределенность может быть выражена соотношением :

$$\underset{n}{y(t)} = \underset{nxx}{D} \cdot \underset{nxn}{P} \cdot \underset{n}{s(t)} = \underset{nxn}{\tilde{P}} \cdot \underset{n}{s(t)}, \qquad (1.4.1.3)$$

где P — матрица перестановок;

D — диагональная масштабирующая матрица;

$\tilde{P}$ — обобщенная матрица перестановок, объединяющая перестановку и масштабирование.

Значительным ограничение является непределенность, поскольку на практике её влияние несущественное, так как наибольшая доля информации заключается в форме сигналов (как например, в биомедицинских системах), а не в величине и порядке компонент. Обзор методов восстановления линейно смешанных сигналов, формирования структуры НС и алгоритма ее обучения лучше начинать с подробного анализа первоначального подхода к решению данной задачи и некоторых результатов, содержащихся в работах.

1.4.2 Восстановление изображений

В данной статье речь идет о модели искажения(восстановления) изображений, модель представлена на рисунке 1.3.

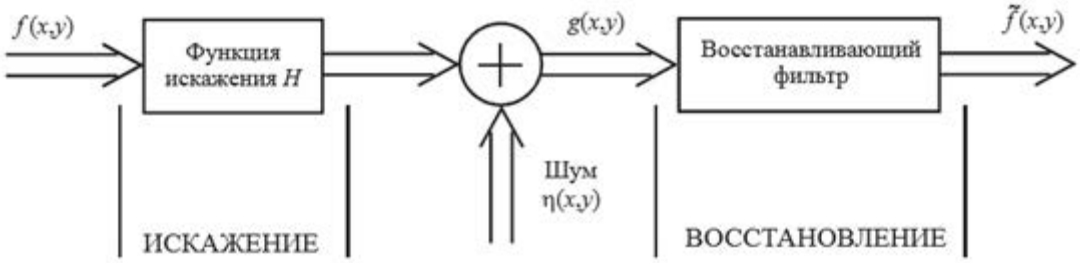


Рисунок 1.3 – Схема восстановления после искажения изображения

где $f(x,y)$ – изображение на входе,

$g(x,y)$ –зашумленное изображение,

$\tilde{f}(x,y)$ – восстановленное изображение,

H – функция искажения,

$\mu(x,y)$ — накладываемый шум (аддитивный).

Задача восстановления изображения тесно связана с восстановлением сигналов и принцип решения в большей части совпадает. В данной статье нужно восстановить $f(x,y)$ по изображению $g(x,y)$, которое подверглось искажению, и имеющимся данным H, а также присутствующему шума $\mu(x,y)$.

Для оценки точности работы алгоритма берется отношение сигнала к шуму (ОСШ)[12].

$$\text{ОСШ}=10\times\log_{10}\left(\sum_{x=1}^{L}\sum_{y=1}^{D}[f(x,y)]^2 / \sum_{x=1}^{L}\sum_{y=1}^{D}[f(x,y)-\tilde{f}(x,y)]^2\right), \qquad 1.4.2.1)$$

$$\sigma^2 = \frac{1}{LD}\sum_{x=1}^{L}\sum_{y=1}^{D}(f(x,y)-\tilde{f}(x,y))^2, \qquad (1.4.2.2)$$

а также, с помощью визуальных оценок восстановленного изображения. L и D обозначают размерность пикселей в формулах (1.4.2.1) и (1.4.2.2).

Для того, чтобы обработать цифровое изображение, представляющее собой двумерную последовательность $g(x,y)$, необходимо построчно разложить в одномерную последовательность дискреты времени, продолжительностью $M = LD$. Это необходимо для того, чтобы ограничить рассмотрение лишь в одномерном случае восстановления.

Далее необходимо решить задачу восстановления цифрового сигнала по его фазовым спектрам. Наиболее известным алгоритмом для такого случая является алгоритм восстановления по фазовому спектру Фурье-образа. Для решения цели восстановления есть несколько способов, одним из таких является итерационный подход, разработанный В. Закстоном и Р. Герхбергом. Но данный метод имеет недостатки. Необходимо производить множество итераций, для того чтобы алгоритм сходился. Также нельзя получить частное решение с заданными характеристиками. Поэтому всё же нобходим другой алгоритм для восстановления цифровых сигналов по их фазовому спектру.

Рассмотрим нейросетевой алгоритм для решения поставленной задачи. [13]

Это необходимо для того, чтобы ограничить рассмотрение только в одномерном случае восстановления.

Далее необходимо решить задачу восстановления цифрового сигнала по его фазовым спектрам. Наиболее известным алгоритмом для этого случая является алгоритм восстановления по фазовому спектру изображения Фурье. Существует несколько способов решения задачи восстановления, одним из которых является итеративный подход, разработанный В. Закстоном и Р. Герхбергом. Но у этого метода есть недостатки. Необходимо выполнить много итераций, чтобы алгоритм сходился. Также невозможно получить конкретное решение с заданными характеристиками. Следовательно, для восстановления цифровых сигналов из их фазового спектра все еще необходим другой алгоритм.

Рассмотрим алгоритм нейронной сети для решения проблемы.[13]

Восстановление сигнала по фазе его дискретного преобразования Фурье (ДПФ) в этом случае рассматривается как задача аппроксимации, когда необходимо получить отклик (сигнал времени) на входное действие (фаза ДПФ с N-точками). Когда необходимо получить отклик (сигнал времени) на входной эффект (фаза N-точечного ДПФ), необходимо решить проблему аппроксимации, то есть восстановить сигнал из фазы его дискеты Фурье-преобразования.

Необходимо настроить M входных нейронов на скрытом слое нейронов и входов, чтобы настроить трехслойный персептрон для решения проблемы. Как показывают эксперименты с нейронной сетью, желательно выбрать N, соответствующее наименьшему целому числу формы, большему или равному 4M. Сигмовидная функция используется для каждого нейрона как нелинейная функция активации. Значения энергии ошибки сигнала, которые генерируются сетью из самого сигнала из фазового спектра ДПФ текстового сигнала, указывают на эффективную работу сети при обучении по среднему и максимальному значению. Алгоритм Левенберга-Марквардта показывает лучшие значения из самых проверенных алгоритмов обучения. Требуется меньше времени на обучение, менее сложный расчет и меньше тренировочных циклов. Этот алгоритм работает намного быстрее, чем алгоритм итерации Герхберга-Закстона, который позволяет получить реконструкцию визуального изображения высочайшего качества; Эти выводы сделаны при тестировании нейронной сети при отсутствии шумов во входных выборках фазового спектра. Но все же стоит отметить, что алгоритм не является оптимальным, поскольку при аппроксимации решения алгоритма нейронной сети возникают ошибки. На рисунке 1.4 показан разработанный и протестированный комплексный метод решения этой проблемы.

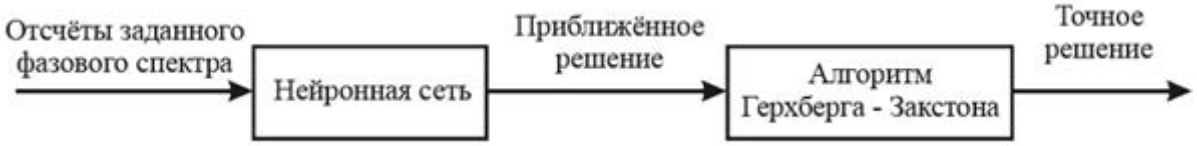


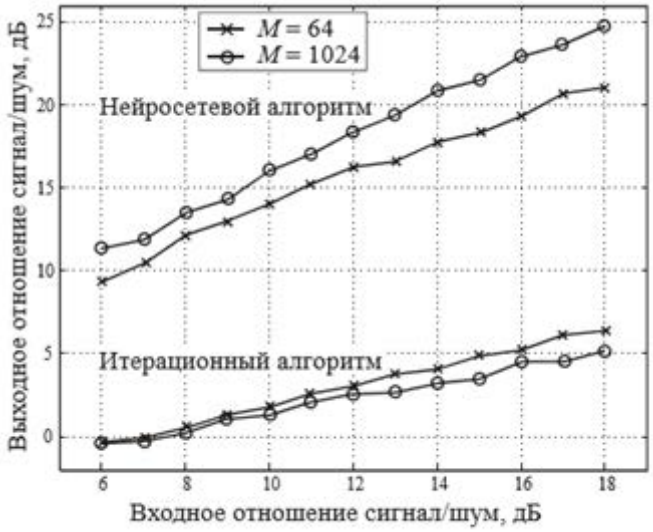
Рисунок 1.4 – Комплексный алгоритм восстановления

В описанной выше процедуре восстановления входные отчёты фазового спектра направляются непосредственно на вход нейронной сети. Результат инициализация метода Герхберга-Закстона полученного выше используется как начальная итерация. При инициализации алгоритма случайных сигналов в среднем в два раза больше, чем количество итераций, которые необходимы при сходимости алгоритма. Параметр останова принимается равным значениям ε =

0.001 и ε = 0.01,который характеризует минимальное допустимое расхождение между текущими и предыдущими итерациями для комплексного и итерационного подхода. Время получения решения при комплексном алгоритме в сравнение с итерационным методом уменьшается в два раза при то, что

Анализ результатов показывает, что время получения решения комплексным алгоритмом по сравнению с итерационным сокращается примерно в 2 раза при сохранении одинаковой ошибки восстановления. Далее производится восстановление сигнала по отсчетам фазового спектра, с накладываемым белым Гауссовым или релевский шум с заданным отношением сигнала и шума, это производится для того, чтобы проверить устойчивость нейросетевого метода в присутствие на входе шума. Итерационный алгоритм Герхберга-Закстона сравнивается с работой нейросетевого алгоритма, точнее их результаты при использовании. Далее представлены зависимости выходного отношения сигнала и шума от входного для белого Гауссова и релеевского шумов (приведены средние значения, полученные из рассмотрения 100 сигналов длительностью M = 64 и 1024) на рисунке 1.5. Анализ результатов показывает, что нейронная сеть гораздо более устойчива к присутствию шума во входных данных, чем итерационный алгоритм.

а)



б)

Рисунок 1.5 – Результат сравнения нейросетевого и итерационного алгоритмов при зашумленных входных данных: белый Гауссов шум (а) и релеевский шум (б)

Можно сделать вывод о работе данного метода:

Итерационный алгоритм с точки зрения отношения сигнала к шуму на выходе при тестировании уступает при наличии трех типов шума нейросетевому алгоритму. Также разработанный алгоритм при отсутствии шума на входе выборок фазового спектра требует существенно меньшего объёма вычислений при сравнение с итерационным методом. Итерационный алгоритм требует в почти два раза больше итераций, чем разработанный комплексный алгоритм.

1.4.3 Восстановление сигнала с использованием нейронных сетей

В этой статье исследуется нелинейная аппроксимация, в которой используется двухсторонняя нейронная сеть подачи вперед для восстановления исходного сигнала из его искаженной версии. Эта нейронная сеть была обучена с помощью управляемого алгоритма обучения. Набор данных обучения использовался из известного сегмента исходных данных и его искаженной версии. Предложенный подход был протестирован в случае, когда исходный сигнал является случайной бинарной временной последовательностью, и искажение получается путем

передачи исходного сигнала через фильтр нижних частот плюс дополнительный белый шум. Показано, что этот подход обеспечивает превосходное восстановление исходного сигнала. Предлагаемый подход обеспечивает нелинейный метод декодирования. Ему не нужна априорная информация о процедуре искажения. Требуется только набор ранее полученных данных обучения. Этот подход также может быть использован для систем, импульсная характеристика которых известна. В этом случае данные обучения могут быть генерируемый возбуждением этой системной функции с белым шумом. Обученная нейронная сеть затем может использоваться для оценки исходного сигнала и может использоваться для приложений реального времени.

Предположим, что искажение канала может быть описано линейной системой с импульсной характеристикой $h_c(n)$ или системной функцией $H_c(\omega)$, тогда сигнал, достигший конца приемника, обозначаемый y(n), может быть задан:

$$y(n)=h(n)*x(n)+\omega(n), \qquad (1.4.3.1)$$

или, что то же самое:

$$Y(\omega)=H_c(\omega)X(\omega)+W(\omega), \qquad (1.4.3.2)$$

где обозначение * обозначает операцию свертки, ω(n) определяет шум канала, который считается белым. Y(ω), X(ω) и W(ω) - преобразования Фурье от y(n), s(n) и ω(n) соответственно.

Задача, изученная в этой статье, связана с задачей восстановления неискаженного цифрового сигнала из набора принятых искаженных сигналов без знания канальной характеристической функции $h_c(n)$ или $H_c(\omega)$. То есть, чтобы создать обратную систему, чтобы отменить эффекты искажения или деконвертировать неискаженный сигнал от принятого сигнала. Пусть g(n) обозначает импульсный ответ такой инверсной системы, то мы имеем:

$$\dot{x}(n)=g(n)*y(n), \qquad (1.4.3.3)$$

где, $\dot{x}(n)$ обозначает восстановленный сигнал, который является оценкой исходный сигнал x(n).

Хорошо известно, что решение этой проблемы во многом зависит от характеристик искажения канала. Когда канал ведет себя как фильтр нижних частот или полосовой фильтр, нам будет очень сложно для восстановления исходного сигнала в условиях шума даже системная функция $H_c(\omega)$ или импульсная характеристика $h_c(n)$ известны. Однако теоретические исследования показывают, что решение существует, когда значения исходного сигнала ограничены некоторыми отдельными значениями. То есть, если $S \subset R$ обозначает набор действительных чисел, а через x(n) обозначает цифровой сигнал, который должен быть передан, в любое время n существует:

$$x(n) \in S. \qquad (1.4.3.4)$$

Нелинейная деконволюция – предлагаемый в настоящем документе подход не требует априорной информации о процедуре искажения системой. Требуется только набор ранее полученных данных, которые включают кусок правильного сообщения и его искаженную версию на конце приемника. Часто не так много, чтобы предоставлять такой набор данных во многих системах связи. С помощью этих данных может быть обучена нелинейная обратная система на основе нейронной сети для автоматического восстановления неискаженного сигнала из его искаженной версии. Эта система реализуется стандартной двухслойной подающей нейронной сетью. Входной вектор представляет собой сегмент искаженного сигнала y(n), а выход представляет собой оценку сегмента оригинала сигнал x(n). Длина отрезка равна m. При восстановлении сигнала в реальном времени искаженная временная последовательность преобразуется в последовательность сегментов длины m:

$$[y(1),y(2),\cdots,y(m)],[y(m+1),y(m+2),\cdots,y(2m)],\cdots \qquad (1.4.3.5)$$

Эти сегменты затем вводятся в обучаемую обратную систему, а выход системы, собранной вместе, используется в качестве оценки исходного сигнала.

Предположим, что N – длина части известного правильного сообщения и его искаженной версии, тогда пары обучения могут быть построены следующим

образом, где X обозначает входной вектор, а Y обозначает цель. Для того чтобы, сократить время обучения, был использован эффективный алгоритм [2,3] поиска по нескольким направлениям поиска. Если известно, что механизм искажения известен, то есть системная функция $H_c(\omega)$ или импульсная характеристика $h_c(n)$ известны, то мы можем сгенерировать набор данных для обучения обратной системы.

Производительность нелинейной обратной системы, как описано ранее, может быть продемонстрирована экспериментом с имитируемыми данными. В моделируемых данных неискаженный сигнал s(n) выбирался как случайная двоичная последовательность. Вероятности появления 0 и 1 в этой последовательности одинаковы. Затем эту последовательность пропускали через низкочастотную футеровку с масляным фильтром 4-го порядка с частотой среза fs = 0,25. Выходной сигнал фильтра, добавленного с помощью последовательности белого шума, использовался как искаженный сигнал y (n). Отношение сигнал / шум составляло 20 дБ. Длина, N, данных обучения и длина, м, сегментации были выбраны равными 300 и 8 соответственно. Таким образом, была использована сеть из 8 входных и 8 выходных сигналов, в которых количество узлов в скрытом слое было выбрано равным 12. Достигнутая ошибка среднего квадрата обучения составила 0,1. Обученная система затем тестируется другими случайными последовательностями, генерируемыми таким же образом. Ошибка тестирования составляет около 0,2. Типичный результат показан на следующих рисунках. Ошибки обучения и тестирования сильно зависят от отношения сигнал / шум.[19]

Выводы по первой главе

По результатам аналитического обзора можно сделать вывод, что существующие методы, алгоритмы восстановления измерительного сигнала имеют ряд недостатков и погрешность, которую можно уменьшить. Цель выпускной квалификационной работы можно сформулировать как: разработка

алгоритма коррекции погрешности путем адаптивного восстановления входного сигнала с использованием нейронных сетей.

Для достижения цели необходимо решить следующие задачи:

- Рассмотреть структуру нейронных сетей и выбрать подходящую модель;
- Разработать алгоритм корректировки погрешности, используя нейронные сети;
- Реализовать метод коррекции в одном из математических пакетов и выполнить качественное сравнение эффективности применения метода при восстановлении конкретных измерительных сигналов.

2. НЕЙРОННЫЕ СЕТИ

2.1 Постановка задачи

Целью второй главы является рассмотреть структуру нейронных сетей, их разновидности и модели, а также выбрать модель, которая будет оптимальной для решения задачи коррекции погрешности путём восстановления измерительного сигнала.

Для того чтобы правильно использовать нейронные сети необходимо изучить базовые понятия и структуры нейронной сети. В процессе разбора темы можно понять, какие параметры той или ной сети лучше подходят для каких-то отдельных областей. Нашей задачей является уменьшение погрешности, следовательно, необходимо сделать акцент на сетях, которые точно вычисляют определенные параметры и не вносят искажений. Также необходимо рассмотреть процесс обучения нейронных сетей, с целью выявить вообще, как проходит процесс обучения и какую сеть будет обучить эффективней для решения нашей задачи. В итоге необходимо выбрать ту модель, которая будет соответствовать всем пунктам и дальше опираться на возможности выбранной НС непосредственно уже в разработке алгоритма.

2.2 Общие сведения о нейронных сетях

Нейронные сети представляют собой глубокое обучение с использованием искусственного интеллекта. Некоторые прикладные задачи слишком тяжелы или находятся вне области применения традиционных алгоритмов машинного обучения. Нейронная сеть разбивает такие блоки на подзадачи и заполняет пробел.

Работают нейроны по следующему принципу. Когда входные узлы снабжены информацией, каждому узлу присваивается значение (в числовой форме). Узлы с более высокими номерами имеют большее значение активации, и на основе передаточной функции и силы соединения узлы передают значение активации.

Как только узлы получают значение активации, оно вычисляет всю сумму и модифицирует ее в соответствии с передаточной функцией. Следующим шагом в этом процессе является применение функции активации, которая помогает нейрону решить, нужно ли передавать сигнал. После процесса активации веса назначаются синапсам для проектирования искусственной нейронной сети. Вес имеет решающее значение для обучения ANN, как функционировать. Веса также могут быть отрегулированы, чтобы определить степень, в которой сигналы могут быть переданы. Веса активации часто меняются во время тренировки искусственного нейронная сеть. После процесса активации сеть достигает выходных узлов. Это шаг, который действует как интерфейс между пользователем и системой. Выходной узел интерпретирует информацию, понятную пользователю. Функции стоимости сравнивают ожидаемый и реальный результат, чтобы оценить производительность модели. В зависимости от ваших требований вы можете выбрать из ряда функций стоимости, чтобы уменьшить функцию потерь. Функция с меньшими потерями приведет к более точному выводу.

Обратное распространение или обратное распространение - это метод расчета градиента функции ошибки в соответствии с весами нейронной сети. Этот процесс обратного расчета помогает устранить неправильные веса и достичь желаемой цели.

Прямое распространение, с другой стороны, является кумулятивным методом достижения конечного результата. В этом методе входные слои обрабатывают информацию и распространяют ее по сети. Как только ожидаемые результаты сравниваются со значениями результатов, вычисляются ошибки и информация распространяется в обратном направлении. После корректировки весов для достижения оптимального уровня сеть можно проверить на конечный результат.

Вкратце, каждый нейрон получает умноженную версию входных данных и случайных весов, которые затем добавляются со значением статического смещения (уникальным для каждого слоя нейрона), а затем передаются в соответствующую функцию активации, которая решает, какое окончательное значение будет выдано из нейрона.

Искусственные нейронные сети основаны на биологических нейронах человеческого тела, которые активируются при определенных обстоятельствах. Они состоят из различных слоев взаимосвязанных искусственных нейронов, приводимых в действие функциями активации, которые помогают включать или же выключать их. Подобно традиционным машинным алгоритмам, здесь также есть определенные значения, которые нейронные сети изучают на этапе обучения. [22]

Существуют различные функции активации, доступные в зависимости от характера вводимых значений. После того, как выходные данные сгенерированы из конечного слоя нейронной сети, вычисляется функция потерь (входные и выходные данные) и выполняется обратное распространение, где веса корректируются, чтобы свести потери к минимуму.

Поиск оптимальных значений весов - это то, на чем сфокусирована вся операция обучения.

Одними из самых важных компонентов в нейронных сетях являются:

- Веса - это числовые значения, которые умножаются на входные данные. При обратном распространении они модифицируются для уменьшения потерь. Проще говоря, веса - это машинные значения, полученные из нейронных сетей. Они саморегулируются в зависимости от разницы между прогнозируемыми результатами и тренировочными.
- Функция активации - это математическая формула, которая помогает нейрону включать / выключать.
- Входной слой представляет размеры входного вектора.

- Скрытый слой представляет собой промежуточные узлы, которые делят входное пространство на области с (мягкими) границами. Он принимает набор взвешенного ввода и производит вывод через функцию активации.
- Выходной слой представляет собой выход нейронной сети.

Массивная параллельная распределённая структура и способности учиться позволяют нейронным сетям находить хорошие приближенные решения сложных (масштабных) задач, которые трудно решить. На практике, однако, нейронные сети не могут обеспечить решение, работая индивидуально. Скорее, они должны быть интегрированы в последовательный системный подход. В частности, сложная интересующая проблема разлагается на ряд относительно простых задач, и нейронным сетям назначается подмножество задач, которые соответствуют их врожденным возможностям. Однако важно признать, что нам предстоит пройти долгий путь (если вообще когда-либо), прежде чем мы сможем построить компьютерную архитектуру, которая имитирует человеческий мозг.

Основные модели НС, которые наиболее востребованы:

1. Сеть прямого распространения;
2. Радиально-базисные нейронные сети ;
3. Рекуррентные сети;
4. Сети Кохонена;
5. Динамические нейронные сети.

Рассмотрим эти нейронные сети поближе.

2.3 Сети прямого распространения

Сети прямого распространения как правило имеют определенно последовательное направление от входных нейронов к выходным, такие как простейший персептрон и многослойный персептрон.[23]

Модель персептрона, предложенная Мински-Папертом, является одной из самых простых и старых моделей нейрона. Это наименьшая единица нейронной сети, которая выполняет определенные вычисления для обнаружения функций

или бизнес-аналитики во входных данных (рисунок 2.1). Он принимает взвешенные входные данные и с помощью функции активации получает входные данные в качестве конечного результата.

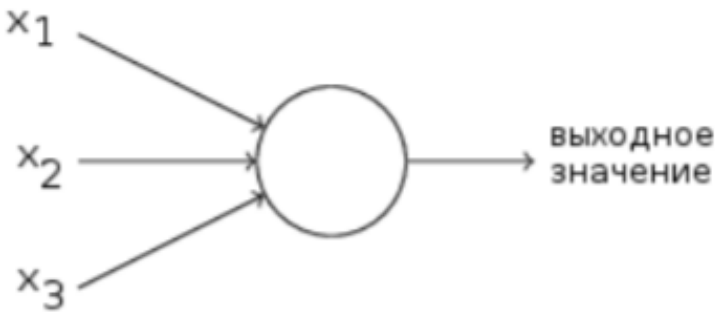


Рисунок 2.1 — Схема персептрона

Персептрон – это контролируемый алгоритм обучения, который классифицирует данные на две категории, таким образом, это двоичный классификатор. Персептрон разделяет входное пространство на две категории с помощью гиперплоскости, представленной следующим уравнением:

$$W^T x + b_i = 0 . \qquad [2.3.1]$$

Используемые веса — числа, выражающие важность вклада каждого входа в конечный результат. Взвешенная сумма (или веса) сравнивается с пороговым значением (threshold), и по результатам определяется, будет ли выдан 0 или 1. Пороговое значение также является параметром нейрона.

$$\begin{cases} 0, IF \sum_j \omega 0 x 0 \le threshold \\ 0, IF \sum_j \omega 0 x 0 > threshold \end{cases} \qquad (2.3.2)$$

Обучение персептрона состоит в изменении матрицы весов в процессе обучения. Существуют 4 исторически сложившихся видов персептронов:

- Персептрон с одним скрытым слоем;
- Однослойный персептрон: входные элементы напрямую соединены с выходными с помощью системы весов. Является простейшей сетью прямого распространения (feedforward network);
- Многослойный персептрон (по Розенблатту): присутствуют

дополнительные скрытые слои;

- Многослойный персептрон (по Румельхарту): присутствуют дополнительные скрытые слои, а обучение проводится по методу обратного распространения ошибки (backpropagation algorithm).

Недостатком обычного персптрона является, то что они предназначены для решения линейных задач. Для нелинейных задач, таких как булева задача, они не совсем подходят.

Многослойные перссепртоны используются для решения задач распознавания речи, машинном переводе. Они представляют собой сложную структуру, где каждый отдельный узел связан со всеми нейронами в следующем слое, что делает его полноценной нейронной сетью. Имеются входные и выходные слои, имеющие несколько скрытых слоев, т.е. в общей сложности не менее трех или более слоев. Они имеет двунаправленное распространение, то есть прямое распространение и обратное распространение.

Входы умножаются на весовые коэффициенты и подаются на функцию активации, а при обратном распространении они модифицируются для уменьшения потерь. Проще говоря, веса - это машинные значения, полученные из нейронных сетей. Они саморегулируются в зависимости от разницы между прогнозируемыми результатами и тренировочными. Используются нелинейные функции активации, за которыми следует softmax в качестве функции активации выходного слоя.

Преимуществом многослойных персептронов является их глубокое обучение, из-за наличия плотных полностью связанных слоёв и обратного распространения. Их достаточно сложно проектировать, что является недостатком.

2.4 Радиально-базисные нейронные сети

Радиально-базисная нейронная сеть состоит из лишь одного скрытого слоя и одного выходного слоя. В различной литературе входной слой сети радиально-базисных функций может как учитываться, так и опускаться. Это связано с тем,

что входной слой такой сети, по сути, является вектором. Иными словами, входной слой можно рассматривать как набор нейронов, каждый из которых является одной из координат такого вектора, но точно так же можно опускать данное допущение. Суть работы радиально-базисной нейронной сети заключается в следующем. Входной вектор поочередно «показывается» каждому из нейронов скрытого слоя, в которых заранее установлены эталонные вектора. Далее входной вектор сравнивается с эталонным. Для сравнения можно использовать любую метрику, например, расстояние Манхэттана, но чаще используется евклидово расстояние. Полученный результат сравнения обрабатывается активационной радиально-базисной функцией. Далее полученный результат умножается на вес синапса нейрона и передается выходному слою. Нейроны выходного слоя суммируют полученные сигналы и выдают ответ, на этом работу сети можно считать оконченной.[18]

Общая структура радиально-базисной сети представлена на рисунке 2.2.

Выходной сигнал каждого элемента определяется функцией Гаусса. Однако, в зависимости от решаемых искусственной нейронной сетью задач, функция Гаусса может незначительно изменяться.

В случае решения задач прогнозирования количество нейронов в выходном слое должно быть равно количеству классов, и основное отличие от задач регрессии состоит в том, что у каждого нейрона выхода будет свое значение веса нейрона смещения.

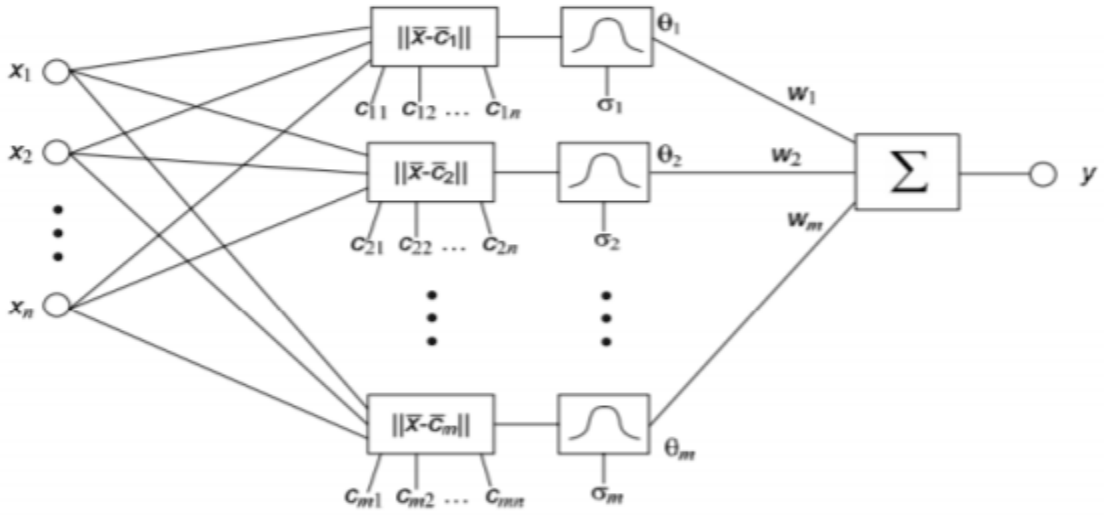


Рисунок 2.2 – Общая структура радиально-базисной сети

Использованы обозначения:

1. n – количество элементов в первом слое;
2. x1, x2, …, xb – координаты входного вектора;
3. b – количество нейронов во 2 слое;
4. *ci*1, *ci*2, ..., *cib* – координаты центра i-го элемента;
5. σij – ширина радиальной функции i-го элемента;
6. θi – выходной сигнал i-го элемента;
7. *wi* – весовой коэффициент выходной связи i-го элемента;
8. w0-вес нейрона смещения;
9. y – выходной сигнал сети.

Для обучения радиально-базисной сети используется алгоритм обратного распространения ошибки, который основан на минимизации целевой функции ошибки сети.[18]

Радиально-базисные нейронные сети имеют ряд преимуществ над другими архитектурами сетей. Во-первых, как было упомянуто выше, они имеют всего один скрытый слой, что позволяет освободить разработчика от решения вопроса о количестве скрытых слоев, и облегчает реализацию. Во- 17 вторых, полученные линейные комбинации можно оптимизировать с помощью известных методов

линейной оптимизации. В связи с этим радиально-базисные нейронные сети быстро обучаются.

Однако у радиально-базисных сетей есть и недостатки. Так, они получаются весьма громоздкими при большой размерности вектора входов. Также к недостаткам можно отнести тот факт, что заранее должно быть известно число эталонов

2.5 Рекуррентные нейронные сети

Рекуррентная нейронная сеть возвращается к входным данным, чтобы помочь в прогнозировании исходов слоя. Первый уровень обычно представляет собой нейронную сеть с прямой связью, за которой следует рекуррентный уровень нейронной сети, где некоторая информация, которая была у него на предыдущем временном шаге, запоминается функцией памяти. Прямое распространение осуществляется в этом случае. Он хранит информацию, необходимую для его будущего использования. Если прогноз неверен, скорость обучения используется для внесения небольших изменений. Следовательно, постепенно увеличивая его в сторону правильного прогноза во время обратного распространения.

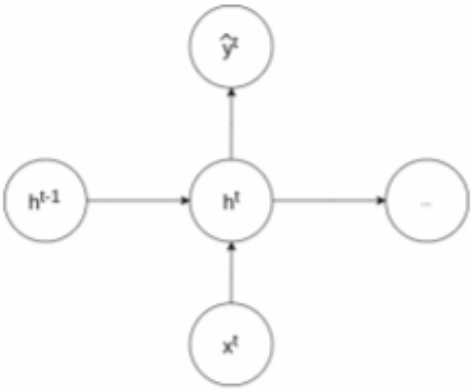


Рисунок 2.3 –Нейрон рекуррентной сети

Преимущества рекуррентных нейронных сетей:

1. Последовательные данные модели, где можно предположить, что каждая выборка зависит от предыдущей выборки, является одним из преимуществ;

2. Используется со сверточными слоями для увеличения эффективности обработки.

Недостатки рекуррентных нейронных сетей:

1. Проблемы градиента исчезновения и сбоя;
2. Обучение повторяющихся нейронных сетей может быть трудной задачей;
3. Трудно обрабатывать длинные последовательные данные, используя ReLU в качестве функции активации.

2.7 Сети Кохонена

Нейронная сеть Кохонена использует обучение без учителя, что является её отличительной четртой.Чаще всего данная нейронная сеть выполняет задачи кластеризации или визуализации. Помимо этого часто применяется моделирования, прогнозирования, поиска закономерностей в больших массивах данных Впервые была предложена финским ученым Теуво Кохоненом в 1989 году. По своей сути является методом проецирования многомерного пространства на пространство с более низкой размерностью. Сеть Кохонена достаточно проста по своей структуре – она представляет всего два слоя: входной и выходной. Такая сеть называется картой. Элементы карты располагаются в n-мерном пространстве, чаще всего двумерном. Сеть Кохонена обучается методом последовательных приближений.[17] Как и в случае с любой искусственной нейронной сетью, в процессе обучения используются входные данные, но подстройка сети идёт под закономерность входных сигналов, а не под значение эталона с выхода. Со случайного выбора выходного расположения центров начинается обучение. С определения наиболее близкого нейрона, у которого скалярное произведение весов и поданного на вход вектора минимально, с этого и начинается последовательная подача обучающих примеров на вход сети. Центральным при подстройке весов у соседних нейронов является победивший нейрон.

Иначе говоря, в данном методе присутствует “состязательное” обучение, если учитывать расстояние нейронов такое правило обучения предполагает "соревновательное" обучение с учетом расстояния нейронов от победившего

“нейрона”. Не в минимизации ошибки при этом заключается обучение, как в случае с рядом других искусственных нейронных сетей, а в подстроичных весах (внутренних параметров нейронной сети) для наибольшего совпадения с входными данными. К преимуществам самоорганизующейся карты Кохонена можно отнести устойчивость к зашумленным данным, возможность упрощения многомерных входных данных с помощью визуализации и быстрое и неуправляемое обучение.

К недостаткам стоит отнести сильную зависимость от первоначальных настроек сети – специалисту, работающему с такой сетью, необходимо глубоко погрузится в объект исследования, и пройти серьезное обучение самому, перед тем как подвергнуть обучению непосредственно самообучающуюся карту Кохонена.

2.8. Динамические нейронные сети

Рассмотрим динамическую структуру нейронной сети с непрерывным временем.

Уравнение состояния:

$$\frac{dx(t)}{dt} = f(x(t), u(t), \omega). \qquad (2.8.1)$$

Уравнение выхода:

$$y(t) = h(x(t), \omega), \qquad (2.8.2)$$

где $x \in \Re^n$ представляет вектор состояния, $u \in \Re^m$ - внешний входной вектор, а $\omega \in \Re^l$ - вектор нейронных параметров, который содержит веса синаптического соединения и соматические операционные параметры; f(.) - это функция, которая представляет структуру нейронной сети, а h(.) - это функция, которая представляет связь между вектором состояния x(t) и выходным вектором $y(t) \in \Re^p$.

Динамически нейронная структура с дискретным временем может быть определена как:

Уравнение состояния:

$$x(k+1) = f(x(k), u(k), \omega). \qquad (2.8.3)$$

Уравнение выхода:

$$y(k) = h(x(k), \omega), \qquad (2.8.4)$$

Эти динамические нервные структуры обладают тремя общими характеристиками, на которые указал Пинеда (1988).

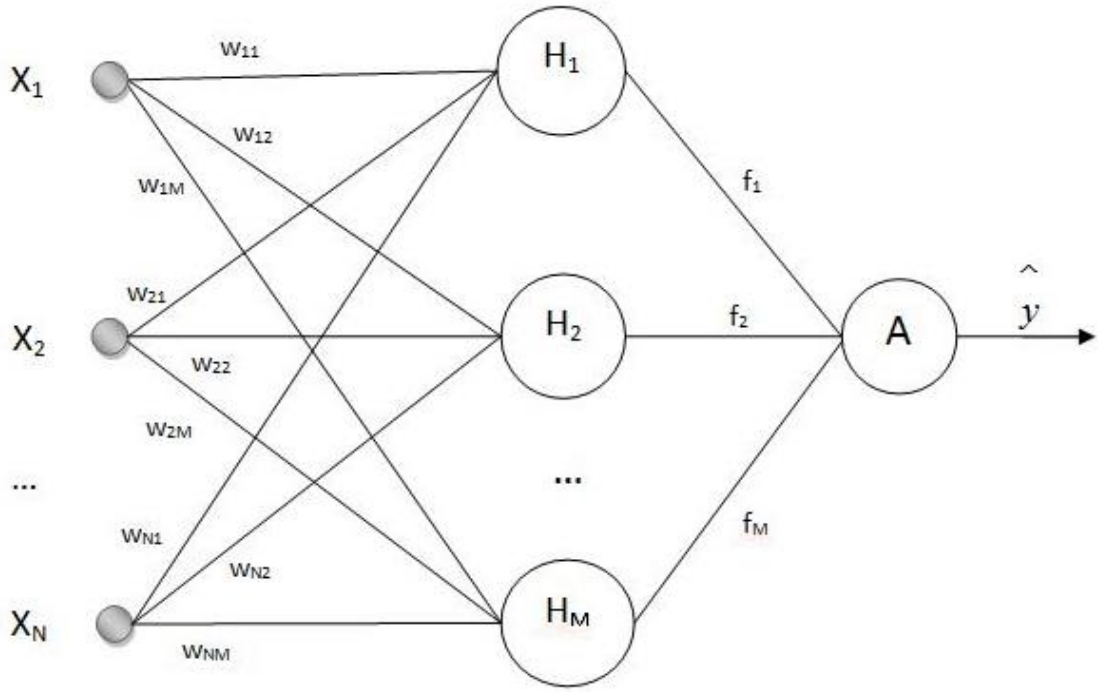


Рисунок 2.5 – Динамическая нейронная сеть

Во-первых, они обычно имеют много степеней свободы. Считается, что человеческий мозг в качестве основы этих динамических моделей содержит от 10^{11} до 10^{13} биологических нейронов. Состояние каждого из этих нейронов может быть смоделировано различными динамическими уравнениями. Обычно считается, что вычислительная мощность и отказоустойчивость нейронных систем являются результатом коллективной динамики нейронных сетей.

Коллективные эффекты объясняют свойства многих физических систем, включая магнетизм, сверхпроводимость и динамику жидкости. Эти статические нейронные сети тривиальны в некоторых отношениях. Все они могут характеризоваться только одной или двумя константами связи. Динамические нейронные системы, с другой стороны, характеризуются многими синаптическими связующими весами.

Вторая общая характеристика заключается в том, что динамические нейронные структуры являются нелинейными. Линейные динамические системы характеризуются тем, что любые два решения системы могут быть добавлены вместе для получения третьего решения. Соответственно, линейные динамические системы могут выполнять только линейные отображения и, следовательно, ограничены в своих вычислительных возможностях. Фактически, нелинейность является обязательным свойством в ассоциативной памяти, если они хотят различать два сохраненных шаблона.

Третья характеристика динамических нейронных систем заключается в том, что они диссипативны. Диссипативная система характеризуется сходимостью потока к многообразию меньшей размерности по мере развития системы. Общие диссипативные системы, как показано на рис. 9.1, могут демонстрировать сложное поведение. Например, они могут сходиться на многообразия с дробными измерениями (аттракторы) или на одномерное многообразие с периодическими орбитами.

Параллельно с разработкой статических нейронных сетей с прямой связью динамические рекуррентные нейронные сети были впервые предложены в контексте проблем ассоциативной или адресно-ориентированной памяти (CAM) (Хопфилд 1982, 1984; Кохонен 1988) для распознавания образов. Неповрежденная модель используется в качестве точки устойчивого равновесия, а ее шумные версии - в качестве бассейна притяжения. Таким образом, создается динамическая нейронная система, связанная с набором паттернов. Если все рабочее пространство правильно разделено такой памятью с адресацией содержимого

(CAM), то система должна иметь устойчивое решение, соответствующее неиспорченному шаблону для любого начального условия, которое представляет образец шаблона. Динамика нейронной сети такого классификатора служит фильтром.

Хорошо известная модель динамических рекуррентных нейронных сетей с некоторыми полезными коллективными вычислительными свойствами принадлежит Хопфилду (1982, 1984). Эта динамическая нейронная структура состоит из большого количества динамических нейронов, которые были введены.

Во второй главе были изучены нейронные сети. Кратко были описаны модели и структура НС. В работе будут использоваться динамические нейронные сети с возможностью персональной настройки параметров сети, которые будут использованы для разработки адаптивного алгоритма коррекции динамической погрешности.

Задача третей главы является непосредственная разработка алгоритма коррекции динамической погрешности с использованием динамической нейронной сети.

3. РАЗРАБОТКА АЛГОРИТМА КОРРЕКЦИИ ПОГРЕШНОСТИ С ПОМОЩЬЮ НЕЙРОННОЙ СЕТИ

3.1 Постановка задачи

Целью данной главы является разработка алгоритма корректирования динамической погрешности путём адаптивного восстановления входного сигнала измерительного преобразователя в присутствии аддитивного шума на его выходе с использованием нейронных сетей.

На выход измерительного преобразователя W(p) накладывается гармонический шум V, зашумленный сигнал проходит через нейросетевой блок, который фильтрует и восстанавливает входной сигнал.

Структурна схема восстановления сигнала, приведена на рисунке 3.1.

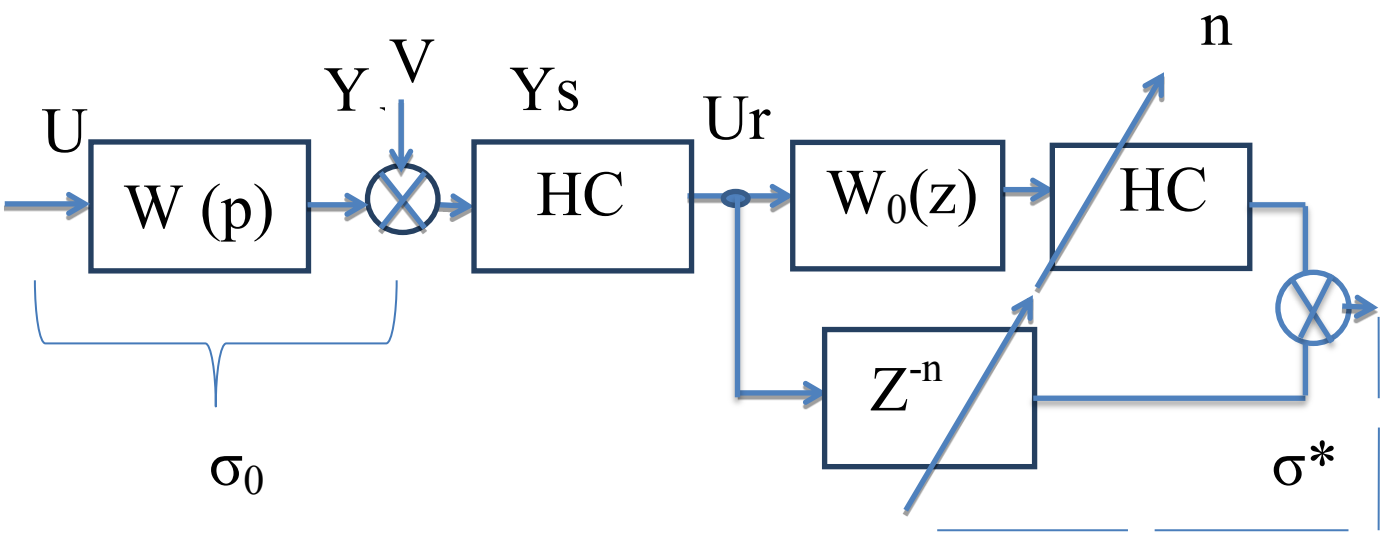


Рисунок 3.1 – Функциональная схема коррекции динамической погрешности

где U – входной сигнал;

W(p) – передаточная функция измерительного преобразователя;

Y − выходной сигнал с измерительного преобразователя;

V − шум;

Ys-зашумленный сигнал;

σ_0 – (Y-U) недоступная оценка;

σ* – (Uff-Ufd) расчетная оценка;

НС – нейронная сеть;

Z^{-1}-задержка;

$W_0(z)$- датчик.

Задачей алгоритма является нахождение приближенной оценки, с помощью которой, можно найти оптимальный параметр нейронной сети, для оптимального восстановления.

3.2 Возможности среды MATLAB

С помощью среды MATLAB можно осуществлять анализ данных, моделировать алгоритмы и разрабатывать приложения

MATLAB широко применяется задачах:

- обработки сигналов;
- обработки различных изображений и видео;
- систем управления;
- автоматизация различных измерений;
- финансового дела;
- органики.

MATLAB сильно упрощает работу человека, позволяет быстро и качественно произвести необходимый анализ и расчёты. Многие учёные, студенты используют эту среду для своих локальных задач, поскольку этим приложением не сложно пользоваться. [19]

3.3 Основные функции в MATLAB при создании и обучении нейронных сетей

С набором инструментов Matlab можно проектировать, обучать, визуализировать и моделировать нейронные сетей. “Neural Network Toolbox” разработан, чтобы учесть многие виды сетей.

Для реализации нейронной сети (процесс проектирования) необходимо выполнить 7 шагов:

1. Сбор данных (Загрузить источник данных);

2. Создание нейронной сети;

3. Настройте сеть (выбор сетевой архитектуры);

4. Инициализируйте веса и уклоны;

5. Тренируй сеть;

6. Проверьте сеть (Тестирование и оценка производительности);

7. Используйте сеть.

Существует несколько этапов для разработки многослойных сетей, например с персептроном:

• Сначала определяется структура сети, выбираются функции активации (веса и уклоны инициализируются).

• Затем настраиваются параметры алгоритма обучения, такие как цель ошибки, максимальное количество эпохи (итерации) и т. д. определены.

• Запустите алгоритм обучения.

• Имитировать выход нейронной сети с помощью измеренных входных данных (сравнивается с измеренными выходами).

• Окончательная проверка должна быть проведена с независимыми данными.

Команды MATLAB, используемые в процедуре: newff, train и sim:

1) newff создает сетевой объект обратной связи и также автоматически инициализирует сеть.

Функция могут принимать следующие параметры;

- P - матрица RxQ1 представляются в виде векторов входных элементов R-элемента Q1;
- T - матрица SNxQ2 представляются в виде целевых векторов SN -элемента Q2;
- S - Размеры N-1 скрытых слоев, от S1 до S (N-1), по умолчанию;
- PR - = Rx2 матрица минимальных и максимальных значений для входных элементов R;
- Si - количество нейронов (размер) в i-м слое, i = 1,…, N_l ;

- N_l - количество слоев;
- TFi - передаточная функция i-го слоя.

По умолчанию «tansig» для скрытых слоев, и «purelin» для выходного слоя. Передаточные функции TF{i} может быть любой дифференцируемой передаточной функцией, такой как TANSIG, LOGSIG или PURELIN;

- BTF - функция обучения обратного распространения, по умолчанию «traingdx»;
- BLF - функция обучения обратного распространения, по умолчанию 'learnngdm';
- PF - исполнительная функция, по умолчанию = «mse» .

Если нейроны должны иметь разные передаточные функции, то они должны быть расположены в разных слоях.

Параметры обучения сети (net.trainParam) должны содержать параметры:

- trainParam - это свойство определяет параметры и значения текущей тренировочной функции;
- net.trainParam - поля этого свойства зависят от текущей тренировочной функции;
- Наиболее часто используемые из этих параметров (составляющие trainParam):

1. net.trainParam.epochs, который сообщает алгоритму максимум количество эпох для обучения;

2. net.trainParam.show, который сообщает алгоритму, сколько эпох должна быть между каждой презентацией спектакля.

Обычно одна эпоха обучения определяется как единое представление всех задаваемых векторов в сеть. Затем сеть обновляется в соответствии с результатами всех этих презентаций.

• Каждый вес и смещение обновляются в соответствии с его функцией обучения после каждой эпохи (один проход через весь набор входных векторов).

Функции исполнения может означать абсолютную погрешность-производительность.

Тренировка прекращается при выполнении любого из следующих условий:

- Максимальное количество эпох (повторений) достигнуто.
- Производительность была сведена к минимуму.
- Максимальное количество времени было превышено.

3.4 Процесс обучения нейронных сетей

Важным элементом конфигурирования искусственной нейронной сети является подбор так называемых гиперпараметров. Гиперпараметры искусственной нейронной сети в общем случае можно разделить на две группы: глобальные и локальные (узловые). К глобальным гиперпараметрам относятся количество скрытых слоев, количество нейронов в каждом слое, уровень обучения и момент, инициализация весов нейронов. Локальные гиперпараметры — тип слоя, функция активации и другие параметры регуляризации. Так как большая часть локальных гиперпараметров, по сути, определено выбранной структурой нейронной сети, а именно сети радиально-базисных функций, то в ходе данной главы будут рассматриваться глобальные гиперпараметры, а именно:

- коэффициент обучения;
- момент обучения m;
- количество нейронов в скрытом слое искусственной нейронной сети.

В данной работе обучение нейронной сети производилось в три этапа:

1. Задавался временной промежуток;
2. В качестве обучающей последовательности использовалась выборка с косинусоидальным сглаживанием:
3. Далее задавались параметры для обучения нейронной сети.

3.5 Обучение нейронной сети с учителем

Обучение с учителем также называется контролируемым обучением. Концептуально мы можем представить себе учителя как обладающего знаниями

об окружающей среде, и это знание представлено рядом примеров ввода-вывода. Окружающая среда, однако, неизвестна нейронной сети. Теперь предположим, что учитель и нейронная сеть подвергаются воздействию вектора обучения (т. е., например), взятого из одной и той же среды. Благодаря встроенным знаниям учитель может обеспечить нейронную сеть желаемой реакцией на этот вектор обучения. Действительно, желаемый ответ - это "оптимальное" действие, которое должно быть выполнено нейронной сетью. Параметры сети настраиваются под комбинированным воздействием обучающего вектора и сигнала ошибки. Сигнал ошибки определяется как разница между желаемым ответом и фактическим ответом сети. Эта настройка выполняется итеративно шаг за шагом с целью в конечном итоге заставить нейронную сеть имитировать учителя; предполагается, что эмуляция является оптимальной в некотором статистическом смысле. Таким образом, знания об окружающей среде, доступные учителю, передаются в нейронную сеть посредством обучения и хранятся в виде" фиксированных " синаптических шкал, представляющих собой долговременную память. Когда это условие достигнуто, мы можем обойтись без учителя и позволить нейронной сети полностью справиться с окружающей средой. [16]

3.6 Обучение без учителя

В контролируемом обучении учебный процесс происходит под опекой учителя. В данном методе нет учителя, который бы контролировал процесс обучения. То есть, нет маркированных примеров функции, которую должна изучать сеть. В рамках этой второй парадигмы определены две подкатегории:

1. Укрепляющее обучение. В процессе укрепляющего изучение отображения ввода-вывода выполняется посредством постоянного взаимодействия со средой, чтобы минимизировать скалярный индекс производительности.Укрепляющее обучение необходимо для минимизации функции затрат на выполнение.

2. Неконтролируемое обучение

При неконтролируемом или самоорганизованном обучении нет внешнего учителя или критика, который бы контролировал процесс обучения. Исходя из того, что сеть должна учиться и свободные параметры сети оптимизируются с учетом меры.

Скорее, предусмотрена независимая от задачи мера качества представления, что сети необходимо учиться, и свободные параметры сети оптимизируются с учетом этой меры.

3.7 Архитектура нейронной сети

В среде MATLAB с помощью toolbox возможно настроить свою персональную сеть, которая бы наиболее лучше подходила для решения задач. Возможно задать связи и построить обратные связи, которые необходимы для корректной работы алгоритма. В структуре разработанной нейронной сети задана одна обратная связь у первого слоя. Она способствует лучшей обработки сигнала. Нейронная сеть должна фильтровать и восстанавливать входной сигнал, при этом не должна вносить искажений в обработку сигнала.

Как уже говорилось раннее в работе используется динамическая нейронная сеть, которая имеет два слоя. Второй слой необходим для того, чтобы сеть лучше фильтровала сигнал. Но не всегда наличие второго слоя способствует лучшей фильтрации. Также играет большое значение, как была обучена нейронная сеть и на каких сигналах производилось это обучение. На рисунке 3.3 представлена структура нейронной сети, которая будет использоваться при моделировании

алгоритма коррекции погрешности. В MATLAB возможно посмотреть структуру сети , для этого необходимо ввести команду view(net).

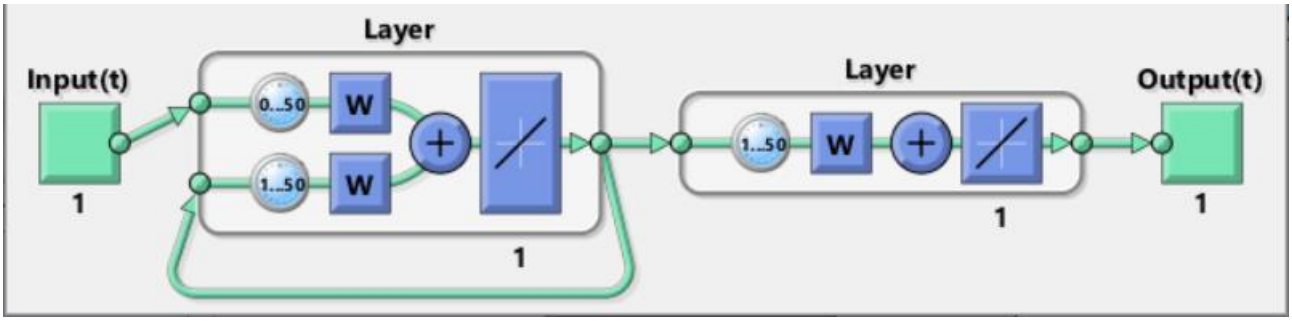


Рисунок 3.3 – Созданная структура нейронной сети

3.8 Алгоритм коррекции

В данном разделе подробно остановимся на описании алгоритма коррекции погрешности. На рисунке 3.5 приведена блок схема коррекции динамической погрешности.


Начало
ПФ 1-го порядка
Задаем обучающую выборку с использованием косинусоидального сглаживания
Настройка нейронной сети (задание связей и параметров)
Обучение нейронной сети
Формирование цикла
Восстановление сигнала с использованием нейронной сети
Расчет СКО
$\sigma_{\hat{S}} = \sqrt{\frac{1}{N-1}\sum_{k=0}^{N-1}(\hat{S}(\omega_k) - \hat{S}_{ср})^2},$
Нахождение оптимального порядка
Уменьшение погрешности при восстановлении сигнала
Вывод всех графиков
Конец

Рисунок 3.4 – Блок-схема алгоритма коррекции

Алгоритм реализован с помощью среды MATLAB, в приложении А предоставлен код программы.

Линейное звено первого порядка называется апериодическим или инерционным звеном, является наиболее распространённым в системах. Описывается дифференциальным уравнением первого порядка, формула (3.8.1):

$$T\frac{dx_2}{dt} + x_2 = Kx_1, \qquad (3.8.1)$$

где K – коэффициент усиление звена;

T – постоянная времена, характеризующая инерционность звена, то есть чем больше постоянная времени, тем дольше длится переходный процесс. [16]

Запишем передаточную функцию линейного звена первого порядка, формула (3.8.2):

$$W(s) = \frac{K}{T * s + 1}, \qquad (3.8.2)$$

где s – комплексная переменная.

3.9 Проблема переобучения и недообучения

Существуют определенные проблемы как для мелких нейронных сетей, так и для глубоких нейронных сетей, такие как переоснащение и время вычислений. На DNN влияет переоснащение, потому что используются дополнительные уровни абстракции, которые позволяют им моделировать редкие зависимости в обучающих данных.

Методы регуляризации, такие как выбывание, ранняя остановка, увеличение данных, передача обучения, применяются во время тренировок для борьбы с перегрузкой. Выпадающая регуляризация случайно пропускает юниты из скрытых слоев во время обучения, что помогает избежать редких зависимостей. DNN принимают во внимание несколько параметров обучения, таких как размер, то есть количество слоев и количество единиц на слой, скорость

обучения и начальные веса. Поиск оптимальных параметров не всегда практичен из-за высокой стоимости времени и вычислительных ресурсов. Несколько хаков, таких как пакетная обработка, могут ускорить вычисления. Большая вычислительная мощность графических процессоров значительно помогла процессу обучения, поскольку требуемые матричные и векторные вычисления хорошо выполняются на графических процессорах.

Dropout - популярный метод регуляризации для нейронных сетей. Глубокие нейронные сети особенно подвержены переоснащению.

Давайте теперь посмотрим, что такое отсев и как он работает.

По словам Джеффри Хинтона, одного из пионеров глубокого обучения: «Если у вас есть глубокая нейронная сеть и она не переоснащается, вам, вероятно, следует использовать более крупную и использовать метод отсева».

Удаление - это метод, при котором во время каждой итерации градиентного спуска мы отбрасываем набор случайно выбранных узлов. Это означает, что мы игнорируем некоторые узлы случайным образом, как будто они не существуют.

Каждый нейрон хранится с вероятностью q и случайно падает с вероятностью 1-q. Значение q может быть различным для каждого слоя в нейронной сети. Значение 0,5 для скрытых слоев и 0 для входного слоя хорошо подходит для широкого круга задач.

Во время оценки и прогноза отсев не используется. Выход каждого нейрона умножается на q, так что вход на следующий слой имеет то же ожидаемое значение.

Идея Dropout заключается в следующем: в нейронной сети без регуляризации отсева нейроны развивают взаимозависимость друг с другом, что приводит к переоснащению.

Идея ранней остановки интуитивна; мы прекращаем обучение, когда ошибка начинает увеличиваться. Здесь под ошибкой мы подразумеваем ошибку, измеренную в проверочных данных, которая является частью обучающих данных,

используемых для настройки гиперпараметров. В этом случае гиперпараметр является критерием остановки.

Процесс увеличения данных, в котором мы увеличиваем количество имеющихся у нас данных или увеличиваем их, используя существующие данные и применяя к ним некоторые преобразования. Точные используемые преобразования зависят от задачи, которую мы намерены достичь. Более того, преобразования, которые помогают нейронной сети, зависят от ее архитектуры.

Например, во многих задачах компьютерного зрения, таких как классификация объектов, эффективный метод увеличения данных добавляет новые точки данных, которые являются обрезанными или переведенными версиями исходных данных.

Когда компьютер принимает изображение в качестве входных данных, он принимает массив значений пикселей. Допустим, что все изображение смещено влево на 15 пикселей. Мы применяем много разных сдвигов в разных направлениях, в результате чего расширенный набор данных во много раз превосходит исходный набор данных.

Процесс передачи процесса отбора предварительно обученной модели и «точной настройки» модели с помощью нашего собственного набора данных называется трансферным обучением. Есть несколько способов сделать это. Несколько способов описаны ниже -

- Мы обучаем предварительно обученную модель на большом наборе данных. Затем мы удаляем последний слой сети и заменяем его новым слоем со случайными весами.
- Затем мы замораживаем веса всех остальных слоев и обучаем сеть в обычном режиме. Здесь замораживание слоев не меняет вес во время градиентного спуска или оптимизации.

Концепция этого заключается в том, что предварительно обученная модель будет действовать как экстрактор признаков, и только последний уровень будет обучен текущей задаче.

По мере того как нейронная сеть учится, она медленно корректирует многие веса, чтобы они могли правильно отображать значение сигнала. Соотношение между Сетевой ошибкой и каждым из этих весовых коэффициентов является производной, dE / dw, которая вычисляет степень, в которой небольшое изменение веса вызывает небольшое изменение ошибки.

Каждый вес является лишь одним фактором в глубокой сети, которая включает в себя множество преобразований; Сигнал веса проходит через активации и суммируется в нескольких слоях, поэтому мы используем цепное правило исчисления, чтобы вернуться к активациям и выходам сети. Это приводит нас к рассматриваемому весу и его связи с общей ошибкой.

Данные две переменные, ошибка и вес, опосредованы третьей переменной, активацией , через которую передается вес. Мы можем рассчитать, как изменение веса влияет на изменение ошибки, сначала рассчитав, как изменение активации влияет на изменение ошибки, и как изменение веса влияет на изменение активации.

Основная идея глубокого обучения - это не более чем: корректировка весов модели в ответ на ошибку, которую она производит, до тех пор, пока вы больше не сможете уменьшить ошибку.

Глубокая сеть обучается медленно, если значение градиента мало, и быстро, если значение велико. Любые неточности в обучении приводят к неточным результатам. Процесс обучения сетей от выхода до входа называется обратным распространением или обратным ходом. Мы знаем, что прямое распространение начинается с ввода и работает вперед. Back prop выполняет обратный / обратный расчет градиента справа налево.

Каждый раз, когда мы вычисляем градиент, мы используем все предыдущие градиенты до этой точки.

Давайте начнем с узла в выходном слое. Край использует градиент в этом узле. Когда мы возвращаемся в скрытые слои, это становится все более

сложным. Произведение двух чисел от 0 до 1 дает вам меньшее число. Значение градиента продолжает уменьшаться, и в результате обратная тренировка занимает много времени, и точность снижается.

3.10 Расчёт СКО

“ Среднее квадратичное отклонение — это квадратный корень из среднего арифметического всех квадратов разностей между данными величинами и их средним арифметическим. “ [19]

В среде MatLab несмещенная и смещенная оценки СКО вычисляются соответственно с помощью функций: std(x) и std(x,1), где x это исходная последовательность длины N. [19]

Несмещенная оценка СКО $\sigma\hat{S}$ вычисляется по формуле:

$$\sigma\hat{S} = \sqrt{\frac{1}{N-1}\sum_{k=0}^{N-1}(\hat{S}(\omega_k) - \hat{S}_{ср})^2}, \qquad (3.16)$$

а смещенная − по формуле:

$$\sigma\hat{S} = \sqrt{\frac{1}{N}\sum_{k=0}^{N-1}(\hat{S}(\omega_k) - \hat{S}_{ср})^2}. \qquad (3.17)$$

В третьей главе был разработан алгоритм коррекции динамической погрешности с использованием динамической нейронной сети.

В четвёртой главе будут представлено моделирование алгоритм, а также проведен эксперимент.

4 РЕЗУЛЬТАТЫ МОДЕЛИРОВАНИЕ

4.1 Постановка задачи

Целью данной главы является, наглядное представление результатов моделирования, т.е. получение результатов работы, по уменьшению динамической погрешности. Алгоритм, разработанный в 3 главе, был реализован в пакете MATLAB в виде кода, код предоставлен в приложении.

На вход измерительного преобразователя подаётся гармонический, ступенчатый и импульсный сигнал с частотой 0.2 Гц (т.е. период равен 5 с). На выход измерительного преобразователя накладывается случайный шум. Далее с помощью корректирующего блока нейронной сети происходит фильтрация и восстановление входного сигнала.

Чтобы нейронная сеть эффективно фильтровала и восстанавливала сигнал её необходимо обучить. Для этого подаются обучающие входные сигналы в виде синусоидального сигнала, на который накладывается случайный шум. Затем необходимо задать количество нейронных слоёв сети, задать задержку и выбрать подходящую функции для обучения.

4.2 Гармонический входной сигнал

4.2.1 Гармонический входной сигнал при синусоидальном шуме

Для начало постоим график восстановленного сигнала с произвольным параметром. Обученная нейронная сеть будет восстанавливать и фильтровать зашумлённый сигнал, но для того, чтобы точно восстановить сигнал необходимо использовать алгоритм. На рисунке 4.1 представлен график восстановленного входного сигнала без поиска оптимального значения, т.е. с произвольным значением параметра нейронной сети.

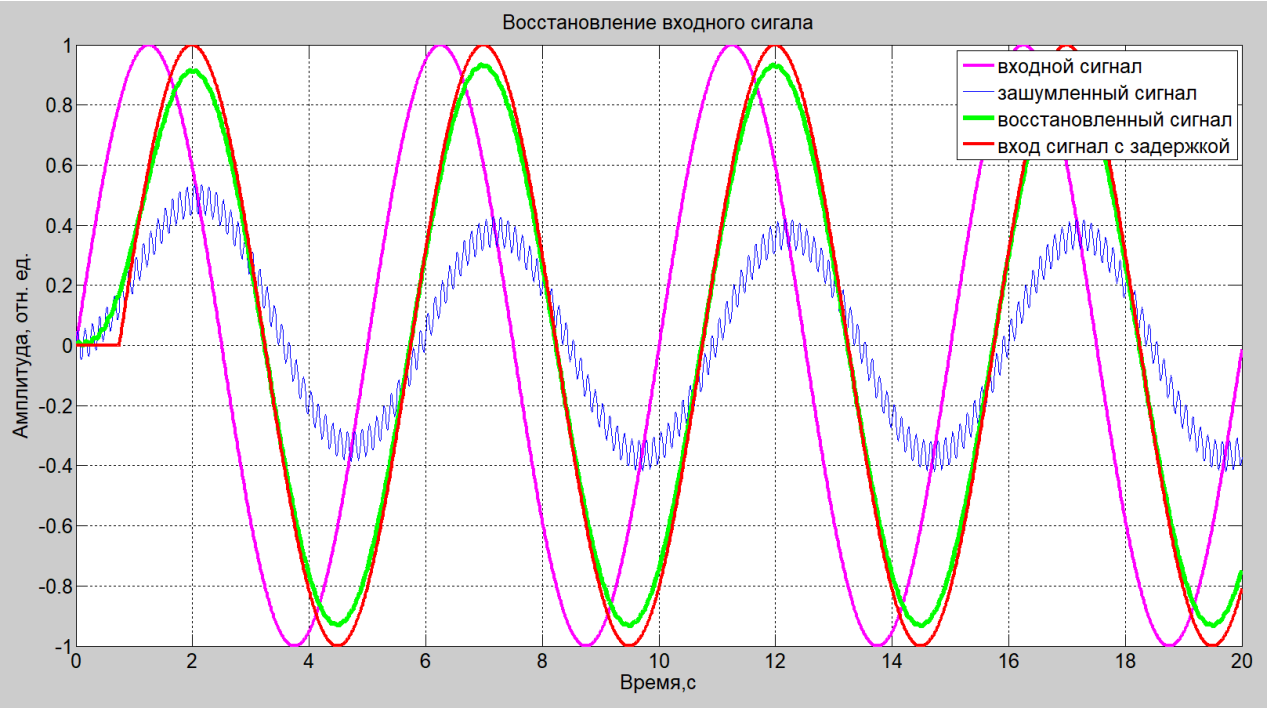


Рисунок 4.1− График восстановленного сигнала при произвольном параметре

Как видно из рисунка 4.1 сигнал восстанавливается неточно, необходимо подобрать параметр нейронной сети, который будет оптимально восстанавливать входной сигнал.

Построим график СКО истинной оценки и СКО приблизительной оценки. Важно понимать, что сигнал Uf с измерительного преобразователя нам недоступен, а доступен только Uff. Необходимо также пропустить сигнал Uf через задержку, чтобы получить информацию с учётом фазовой задержки.

Теоретическая оценка СКО находится по формуле:

$$\sigma_0 = \sqrt{\frac{1}{N\text{-}1}\sum_{k=0}^{N-1}(U_{f_k} - U_{d_k})^2}. \qquad (4.1)$$

Расчётная оценка СКО находится по формуле:

$$\sigma = \sqrt{\frac{1}{N\text{-}1}\sum_{k=0}^{N-1}(U_{k\,ff} - U_{k\,fd})^2}. \qquad (4.2)$$

На рисунке 4.2 показан график СКО востановленного сигнала, зависящий от параметра нейронной сети.

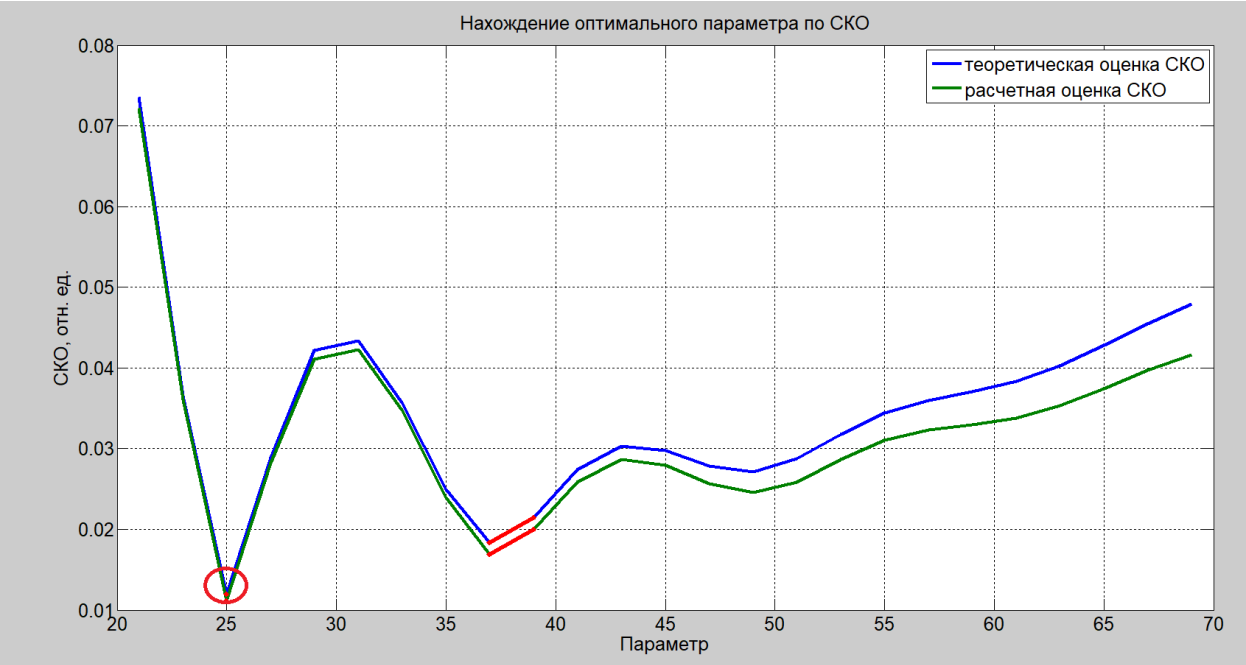


Рисунок 4.2− График СКО восстановленного сигнала

По рисунку 4.2 видно, что наименьшее значение СКО приобретает в отметке 25.

Приведем значения теоретической оценки СКО и расчётной оценки СКО в Таблице 1.

Таблица 1 – СКО оценок динамической погрешности

Значение коэффициента	Теоретическая оценка	Расчётная оценка
21	0.0736	0.0722
23	0.0365	0.0360
25	**0.0120**	**0.0112**
27	0.0289	0.0282
29	0.0422	0.0411

По результатам Таблицы 1 один видно, что самый оптимальный параметр нейронной сети 25.

На рисунке 4.3 построен восстановленный сигнал с уже уточнённым порядком.

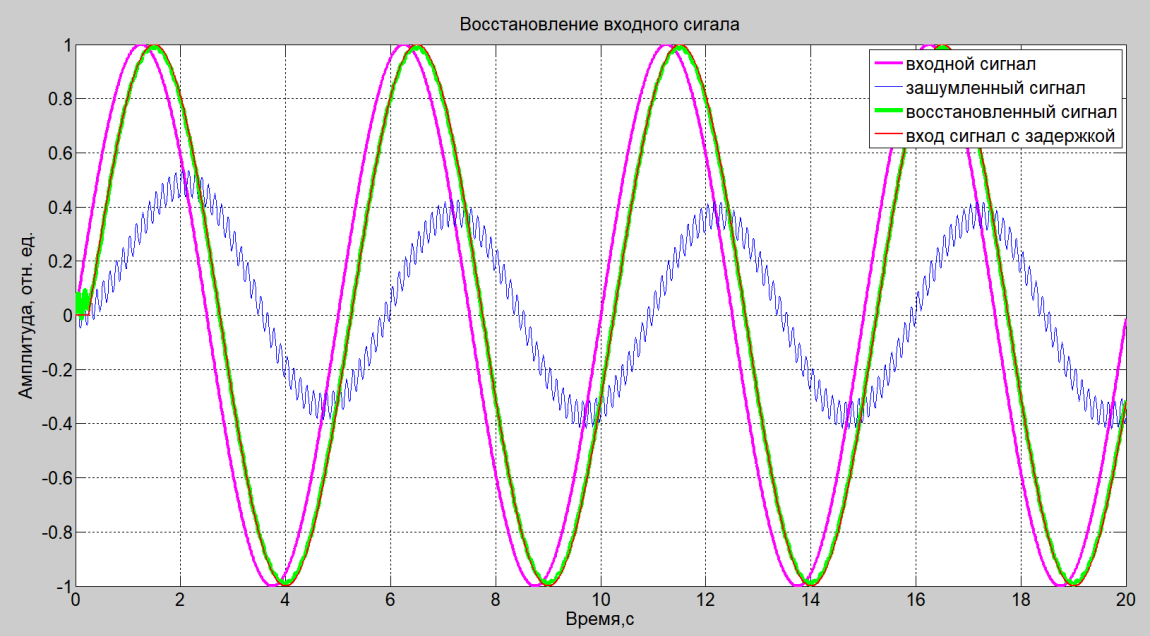


Рисунок 4.3− График восстановленного сигнала с оптимальным параметром

Произведем сравнение полученных оценок по формуле:

$$\frac{\sigma_0}{\sigma} = \frac{0{,}0120}{0{,}0112} = 1{,}07. \qquad (4.3)$$

Из полученного значения можно сделать вывод о том, что оценка СКО уменьшилось в 1.07 раза, это говорит о том, что мы достигли решения поставленной цели выпускной квалификационной работы.

4.2.2 Гармонический входной сигнал при случайном шуме

Для начало постоим график восстановленного сигнала с произвольным параметром. Обученная нейронная сеть будет восстанавливать и фильтровать зашумлённый сигнал, но для того, чтобы точно восстановить сигнал необходимо использовать алгоритм. На рисунке 4.4 представлен график восстановленного входного сигнала без поиска оптимального значения, т.е. с произвольным значением параметра нейронной сети.

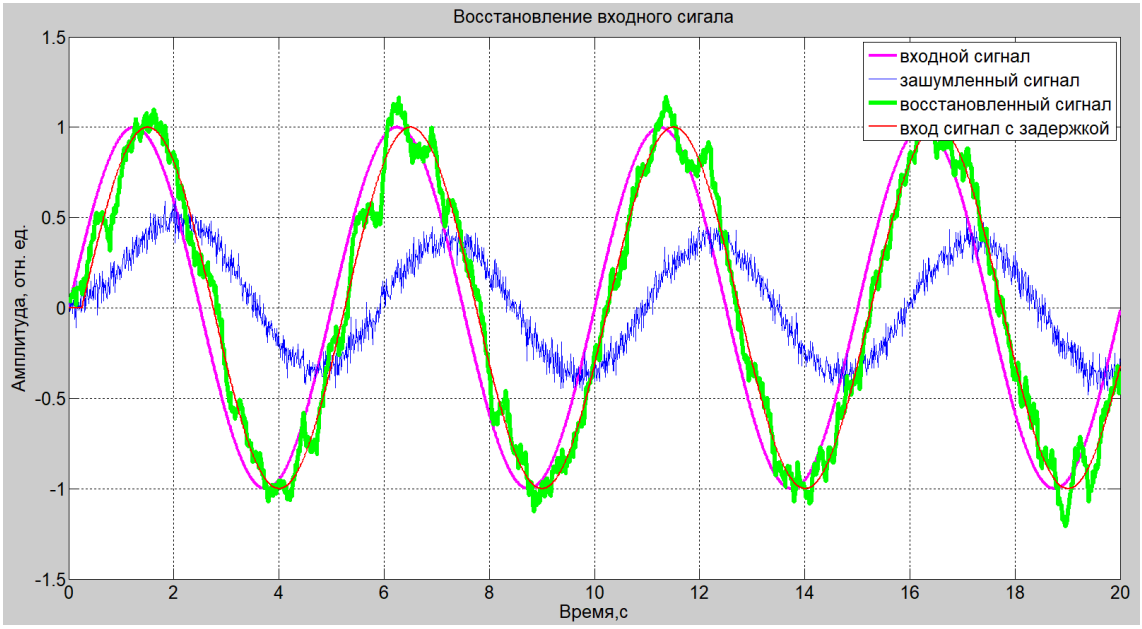


Рисунок 4.4− График восстановленного сигнала при произвольном параметре

Как видно из рисунка 4.4 сигнал восстанавливается неточно, необходимо подобрать параметр НС, который будет оптимально восстанавливать входной сигнал.

Теоретическая оценка СКО находится по формуле (4.1). Расчетная оценка СКО находится по формуле (4.2).

На рисунке 4.5 показан график СКО востановленного сигнала, зависящий от параметра НС.

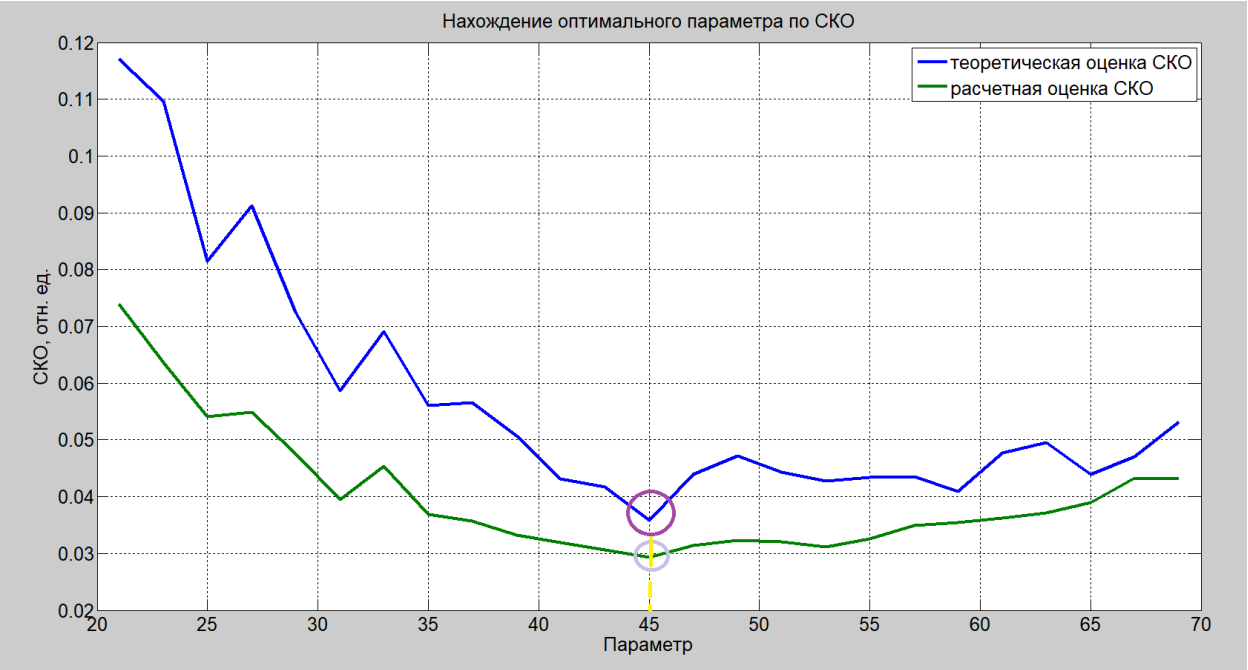


Рисунок 4.5− График СКО восстановленного сигнала

По рисунку 4.5 видно, что наименьшее значение СКО приобретает на интервале от 41 до 49.

Приведем значения теоретической оценки СКО и расчётной оценки СКО в Таблице 2.

Таблица 2 – СКО оценок динамической погрешности

Значение коэффициента	Теоретическая оценка	Расчётная оценка
41	0.0432	0.0320
43	0.0412	0.0306
45	**0.0358**	**0.0293**
47	0.0439	0.0314
49	0.0471	0.0323
51	0.0443	0.0320

По результатам таблицы 2 один видно, что самый оптимальный параметр НС 45.

На рисунке 4.6 построен восстановленный сигнал с уже уточнённым параметром.

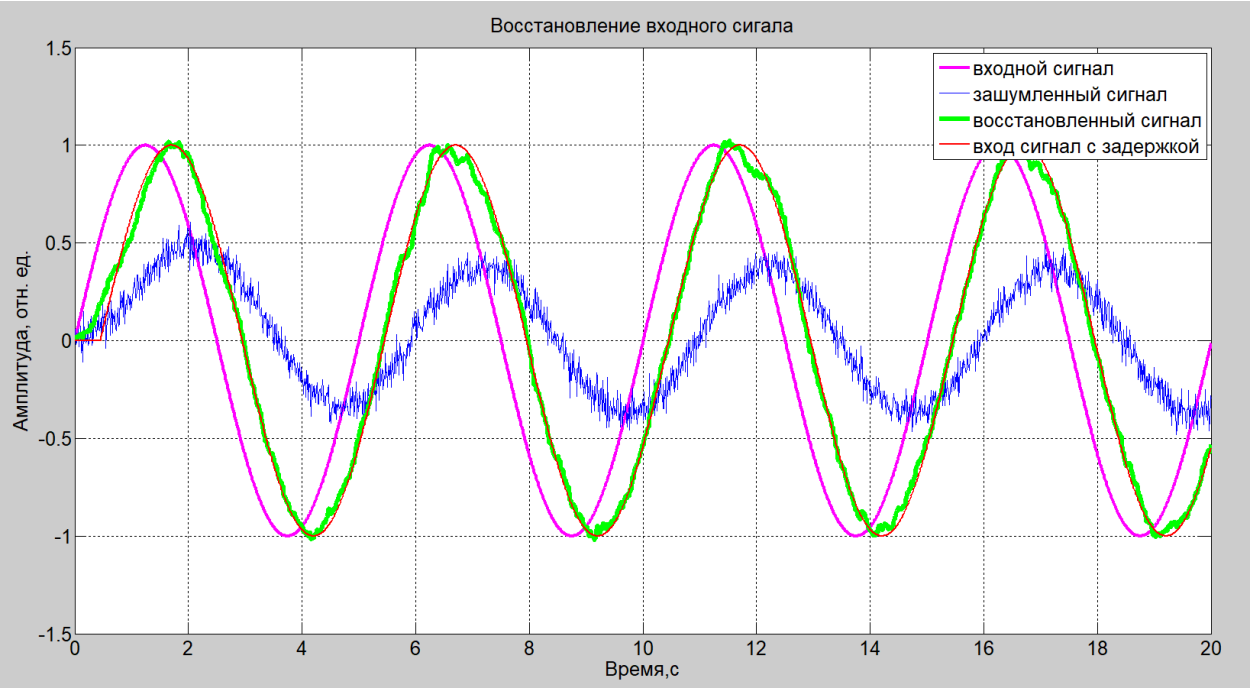


Рисунок 4.6– График восстановленного сигнала с оптимальным параметром

Из графика 4.6 видно, что восстановленный сигнал достаточно точно повторяет входной сигнал. Полностью устранить погрешность, разумеется, не получится, но с помощью разработанного алгоритма удалось её уменьшить.

Произведем сравнение полученных оценок по формуле:

$$\frac{\sigma_0}{\sigma} = \frac{0{,}0358}{0{,}0293} = 1{,}22. \qquad (4.4)$$

Из полученного значения можно сделать вывод о том, что оценка СКО уменьшилось в 1.22 раза, это говорит о том, что мы достигли решения поставленной цели выпускной квалификационной работы.

4.3 Ступенчатый входной сигнал

4.3.1 Ступенчатый входной сигнал с синусоидальным шумом

Для начало постоим график восстановленного сигнала с произвольным параметром. Обученная нейронная сеть будет восстанавливать и фильтровать зашумлённый сигнал, но для того, чтобы точно восстановить сигнал необходимо использовать алгоритм. На рисунке 4.7 представлен график восстановленного входного сигнала без поиска оптимального значения, т.е. с произвольным значением параметра нейронной сети.

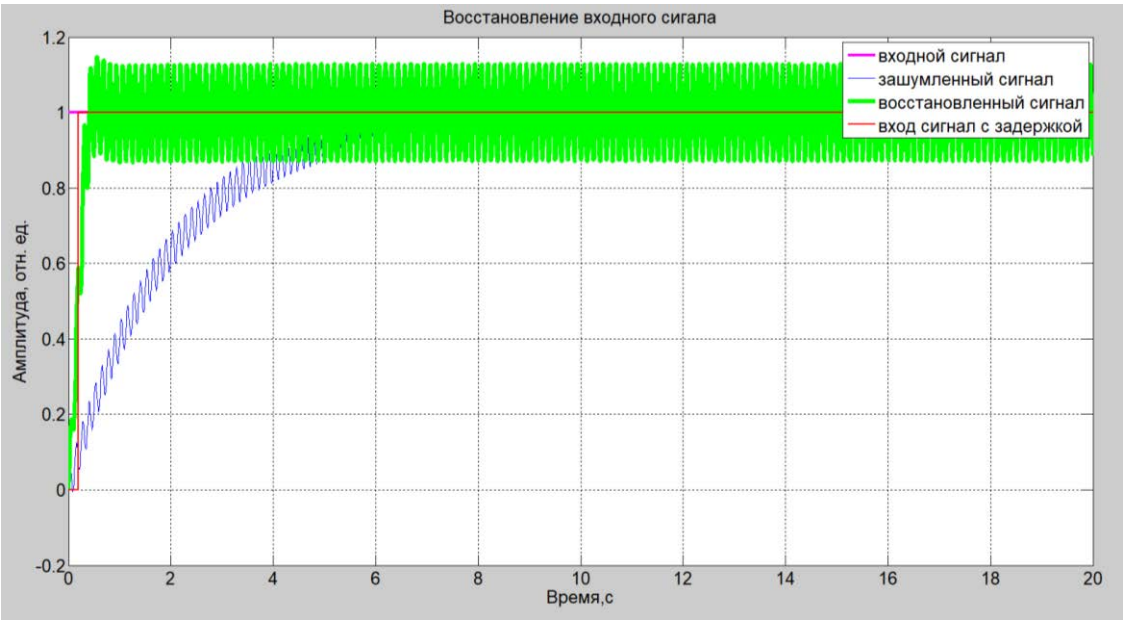


Рисунок 4.7 − График восстановленного сигнала при произвольном параметре

Как видно из рисунка 4.7 сигнал восстанавливается неточно, необходимо подобрать параметр НС, который будет оптимально восстанавливать входной сигнал.

Теоретическая оценка СКО находится по формуле (4.1). Расчетная оценка СКО находится по формуле (4.2).

На рисунке 4.8 показан график СКО востановленного сигнала, зависящий от параметра НС.

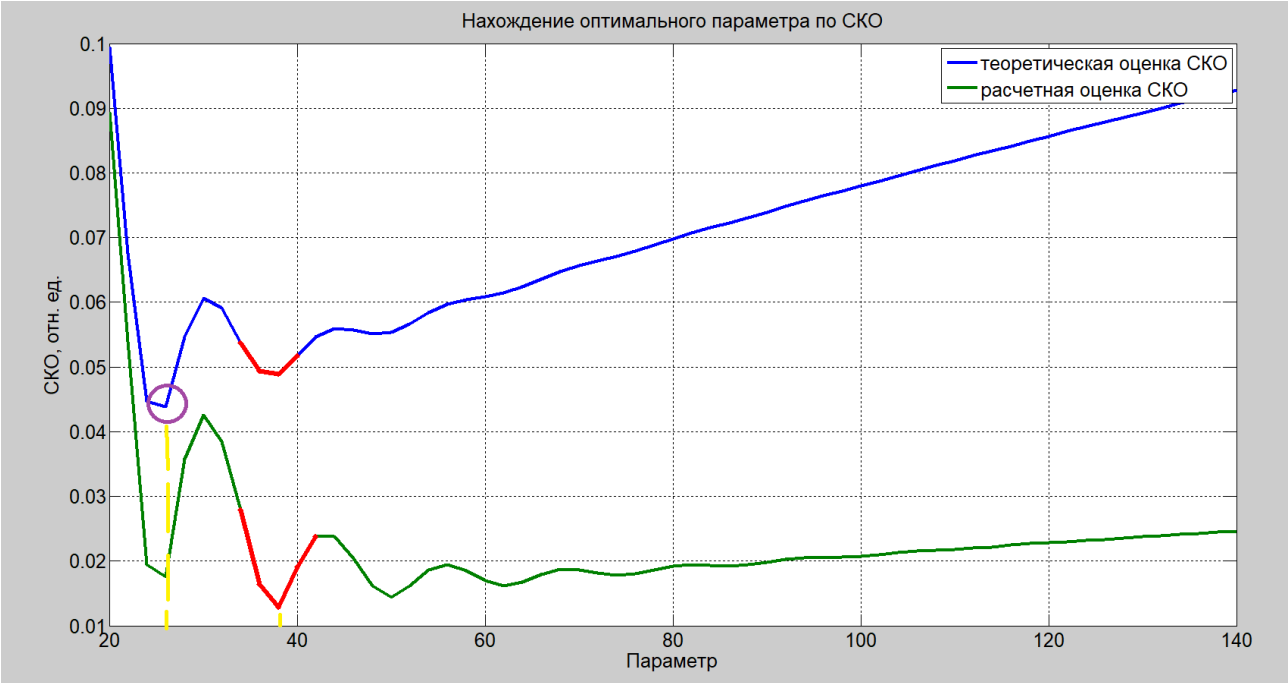


Рисунок 4.8 − График СКО восстановленного сигнала

По рисунку 4.8 видно, что наименьшее значение СКО приобретает на интервале от 22 до 40

Приведем значения теоретической оценки СКО и расчётной оценки СКО в Таблице 3.

Таблица 3 – СКО оценок динамической погрешности

Значение коэффициента	Теоретическая оценка	Расчётная оценка
24	0.0446	0.0195
26	0.0438	0.0176
28	0.0546	0.0358
30	0.0607	0.0426
32	0.0591	0.0385
34	0.0537	0.0279
36	0.0493	0.0163
38	**0.0489**	**0.0129**
40	0.0518	0.0191
42	0.0547	0.0238

По результатам таблицы 3 видно, что самый оптимальный параметр НС 38.

На рисунке 4.9 построен восстановленный сигнал с уже уточнённым параметром НС.

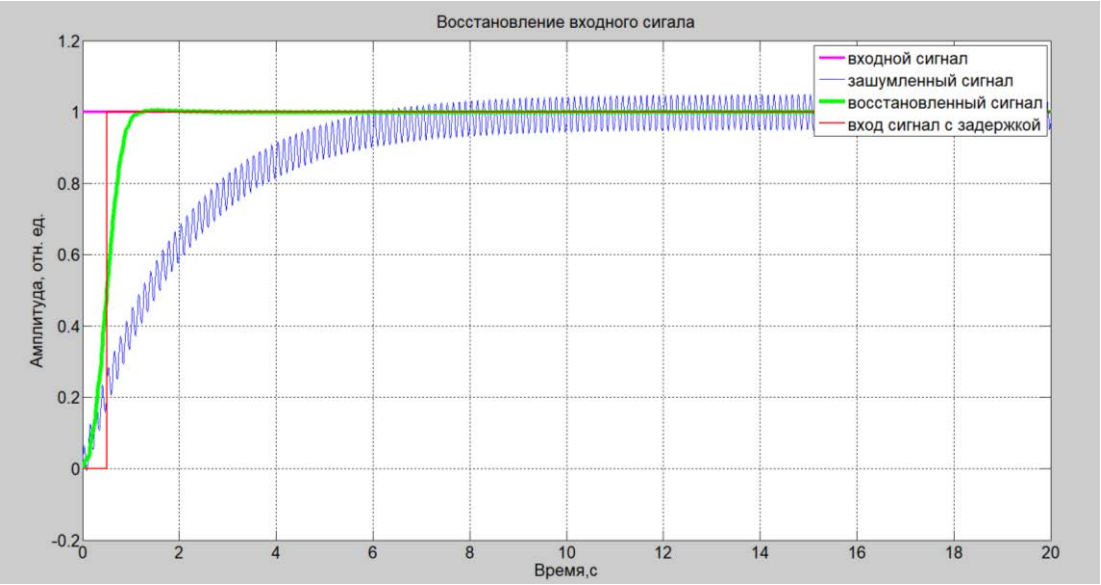


Рисунок 4.9 – График восстановленного сигнала с оптимальным параметром

Произведем сравнение полученных оценок:

$$\frac{\sigma_0}{\sigma} = \frac{0{,}0489}{0{,}0129} = 3{,}8. \qquad (4.11)$$

Из полученного значения можно сделать вывод о том, что СКО уменьшилось в 3,8 раза, это говорит о том, что мы достигли решения поставленной цели выпускной квалификационной работы.

4.3.4. Ступенчатый входной сигнал со случайным шумом

Для начало постоим график восстановленного сигнала с произвольным параметром. Обученная нейронная сеть будет восстанавливать и фильтровать зашумлённый сигнал, но для того, чтобы точно восстановить сигнал необходимо использовать алгоритм. На рисунке 4.10 представлен график восстановленного входного сигнала без поиска оптимального значения, т.е. с произвольным значением параметра нейронной сети.

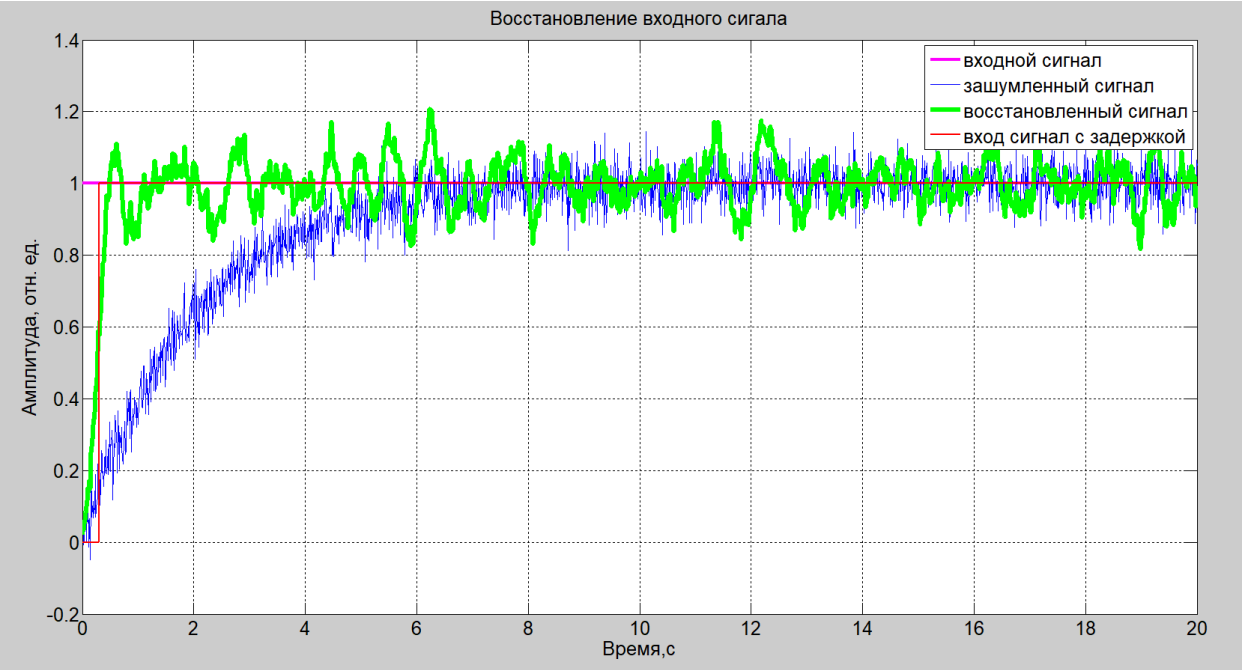


Рисунок 4.10− График восстановленного сигнала при произвольном параметре

Как видно из рисунка 4.10 сигнал восстанавливается неточно, необходимо подобрать параметр НС, который будет оптимально восстанавливать входной сигнал.

Теоретическая оценка СКО находится по формуле (4.1). Расчетная оценка СКО находится по формуле (4.2).

На рисунке 4.11 показан график СКО востановленного сигнала, зависящий от параметра НС.

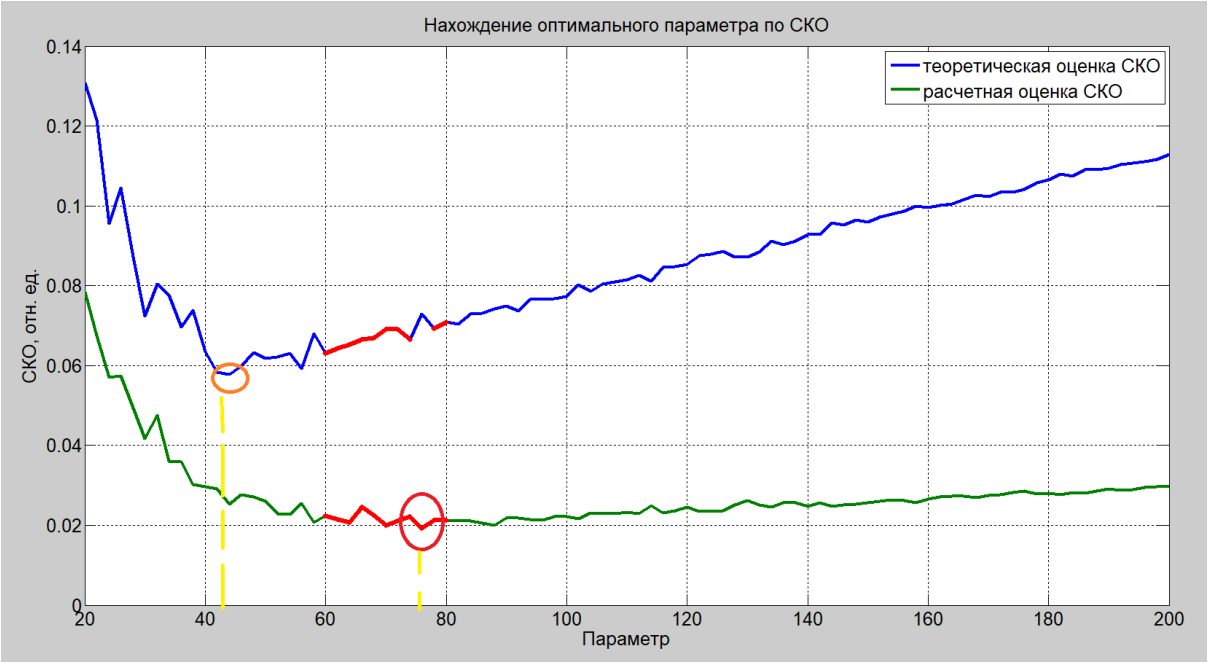


Рисунок 4.11− График СКО восстановленного сигнала

По рисунку 4.11 видно, что наименьшее значение СКО приобретает на интервале от 60 до 80.

Приведем значения теоретической оценки СКО и расчётной оценки СКО в Таблице 4.

Таблица 4 – СКО оценок динамической погрешности

Значение коэффициента	Теоретическая оценка	Расчётная оценка
60	0.0631	0.0223
62	0.0644	0.0215
64	0.0652	0.0207
66	0.0666	0.0246
68	0.0669	0.0225
70	0.0691	0.0199
72	0.0690	0.0211
74	0.0665	0.0222
76	**0.0729**	**0.0193**
78	0.0693	0.0213
80	0.0708	0.0214

По результатам Таблицы 4 видно, что самый оптимальный параметр 76.

На рисунке 4.12 построен восстановленный сигнал с уже уточнённым параметром.

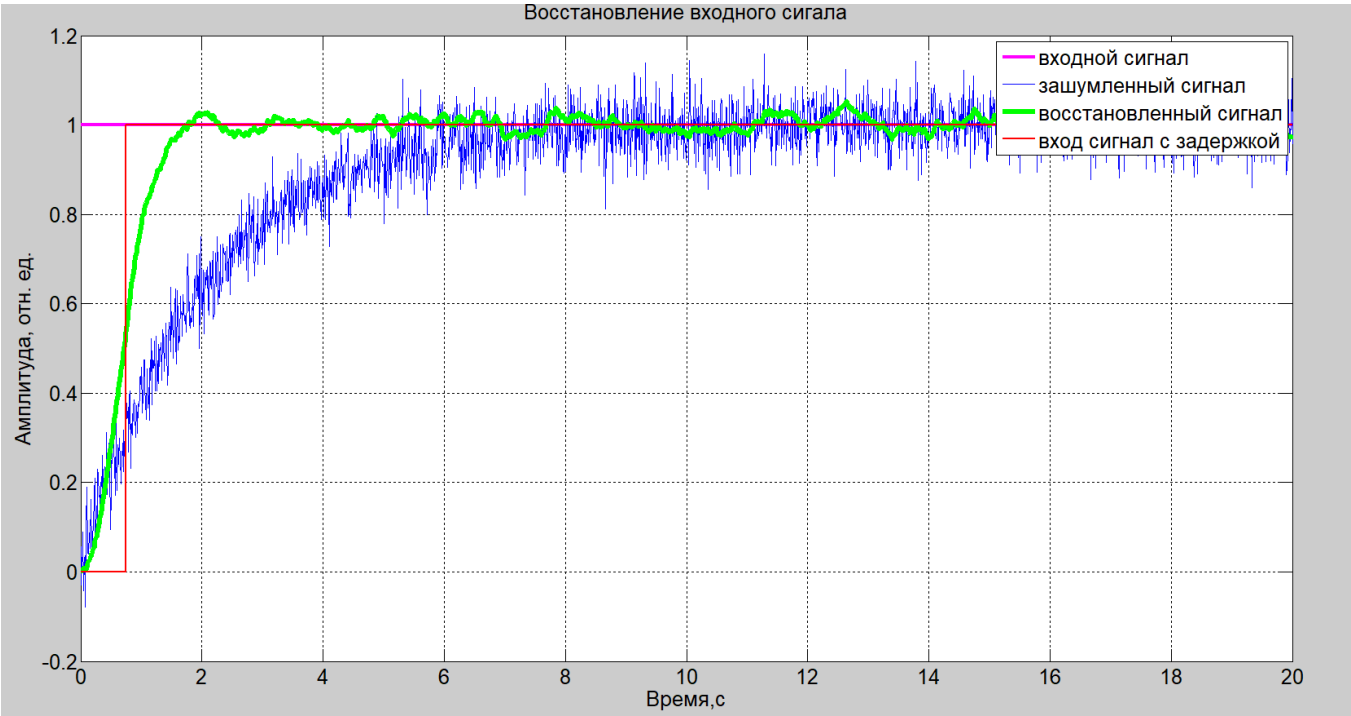


Рисунок 4.12– График восстановленного сигнала с оптимальным параметром

Произведем сравнение полученных оценок:

$$\frac{\sigma_0}{\sigma} = \frac{0{,}0739}{0{,}0193} = 3{,}82 \qquad (4.12)$$

Из полученного значения можно сделать вывод о том, что СКО уменьшилось в 3,8 раза, это говорит о том, что мы достигли решения поставленной цели выпускной квалификационной работы

4.4 Импульсный входной

4.4.1 Ступенчатый входной сигнал с синусоидальном шумом

Для начало постоим график восстановленного сигнала с произвольным параметром. Обученная нейронная сеть будет восстанавливать и фильтровать

зашумлённый сигнал, но для того, чтобы точно восстановить сигнал необходимо использовать алгоритм. На рисунке 4.13 представлен график восстановленного входного сигнала без поиска оптимального значения, т.е. с произвольным значением параметра нейронной сети.

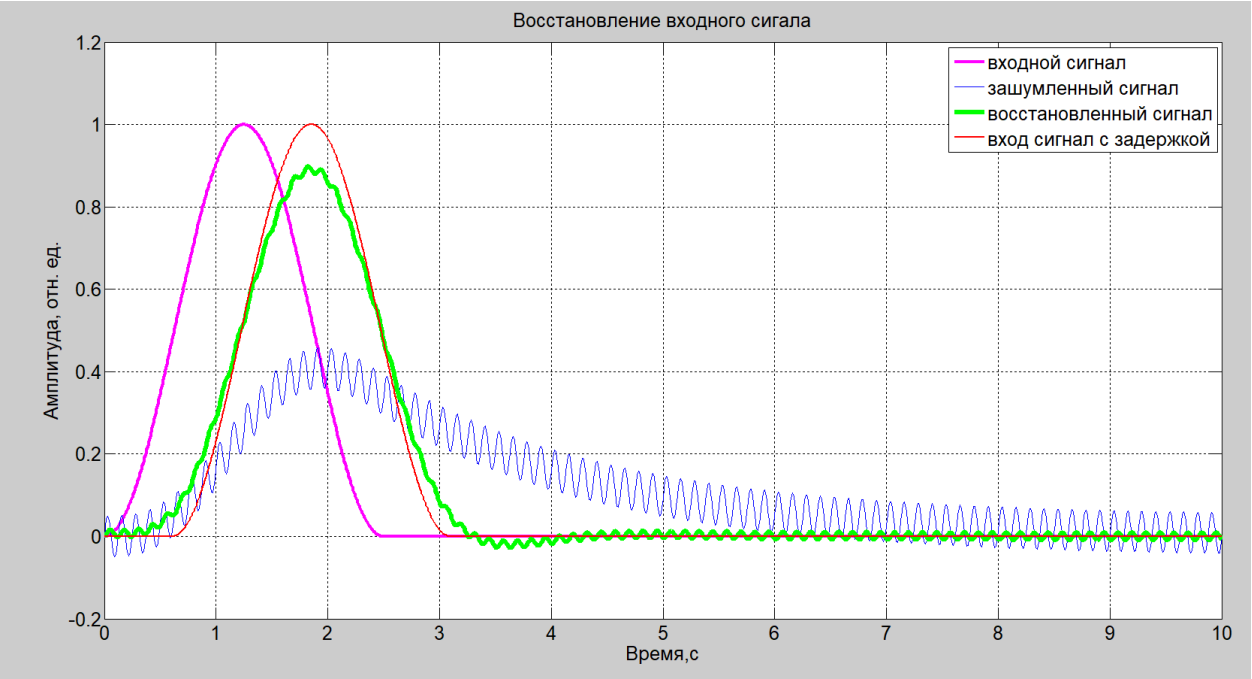


Рисунок 4.13− График восстановленного сигнала при произвольном параметре

Как видно из рисунка 4.13 сигнал восстанавливается неточно, необходимо подобрать параметра НС, который будет оптимально восстанавливать входной сигнал.

Теоретическая оценка СКО находится по формуле (4.1). Расчетная оценка СКО находится по формуле (4.2).

На рисунке 4.13 показан график СКО востановленного сигнала, зависящий от параметра НС.

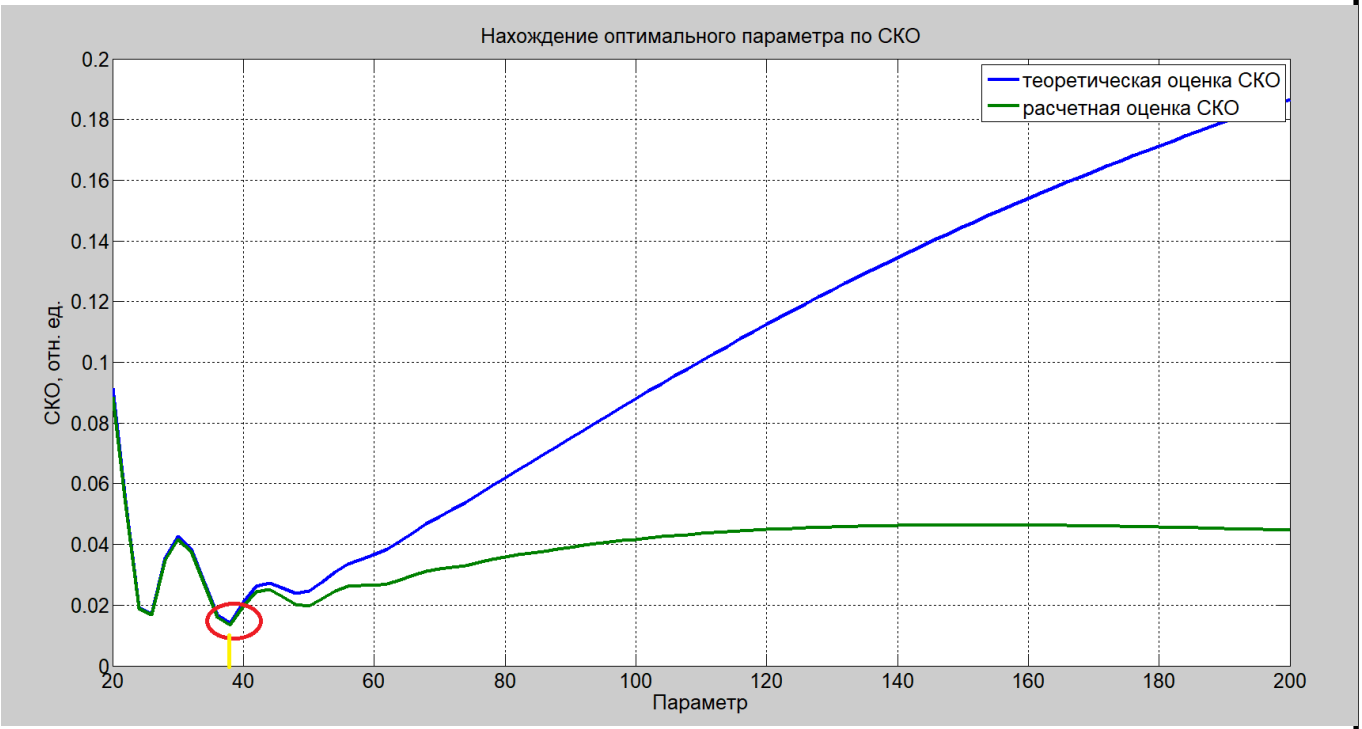


Рисунок 4.14– График СКО восстановленного сигнала

По рисунку 4.14 видно, что наименьшее значение СКО приобретает на интервале от 34 до 44.

Приведем значения теоретической оценки СКО и расчётной оценки СКО в Таблице 5.

Таблица 5 – СКО оценок динамической погрешности

Значение коэффициента	Теоретическая оценка	Расчётная оценка
34	0.0279	0.0273
36	0.0167	0.0161
38	**0.0143**	**0.0133**
40	0.0210	0.0197
42	0.0262	0.0245
44	0.0273	0.0251

По результатам таблицы 5 видно, что самый оптимальный параметр НС 38.

На рисунке 4.14 построен восстановленный сигнал с уже уточнённым параметром.

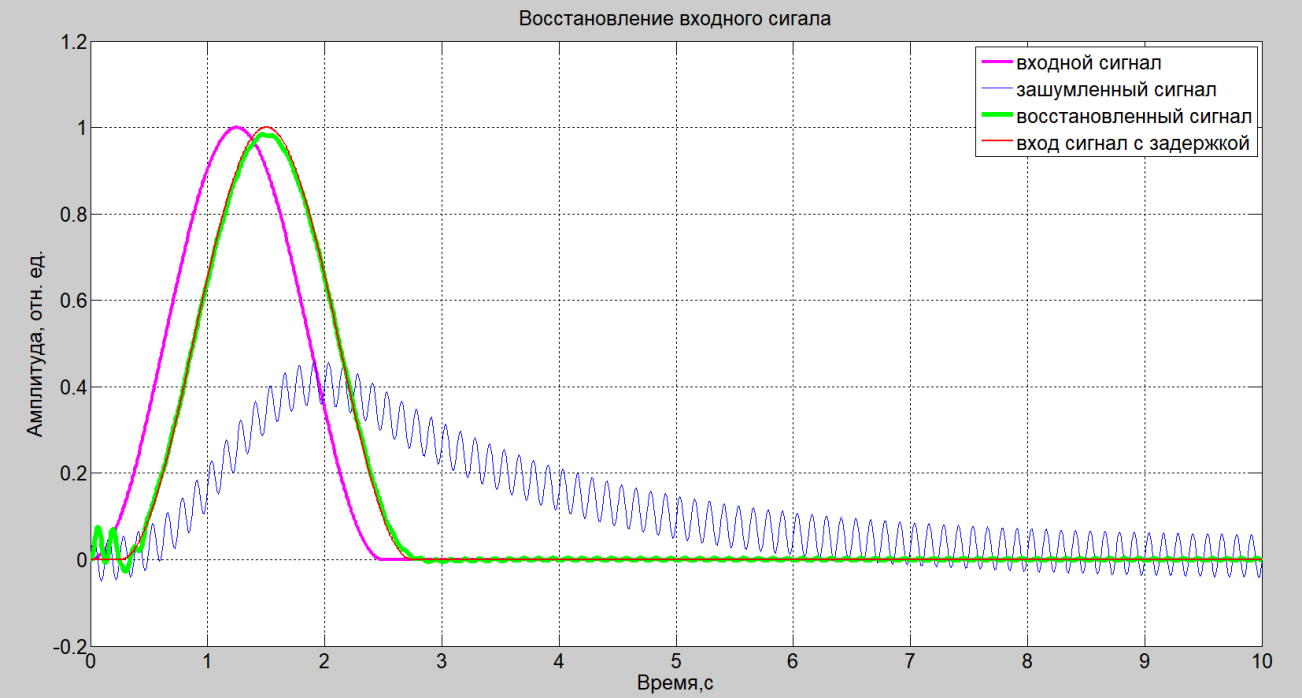


Рисунок 4.15 – График восстановленного сигнала с оптимальным параметром

Произведем сравнение полученных оценок:

$$\frac{\sigma_0}{\sigma} = \frac{0{,}0143}{0{,}0133} = 1{,}1. \qquad (4.14)$$

Из полученного значения можно сделать вывод о том, что СКО уменьшилось в 1,1 раз, это говорит о том, что мы достигли решения поставленной цели выпускной квалификационной работы.

4.4.2 Ступенчатый входной сигнал со случайным шумом

Для начало постоим график восстановленного сигнала с произвольным параметром. Обученная нейронная сеть будет восстанавливать и фильтровать зашумлённый сигнал, но для того, чтобы точно восстановить сигнал необходимо использовать алгоритм. На рисунке 4.16 представлен график восстановленного входного сигнала без поиска оптимального значения, т.е. с произвольным значением параметра нейронной сети.

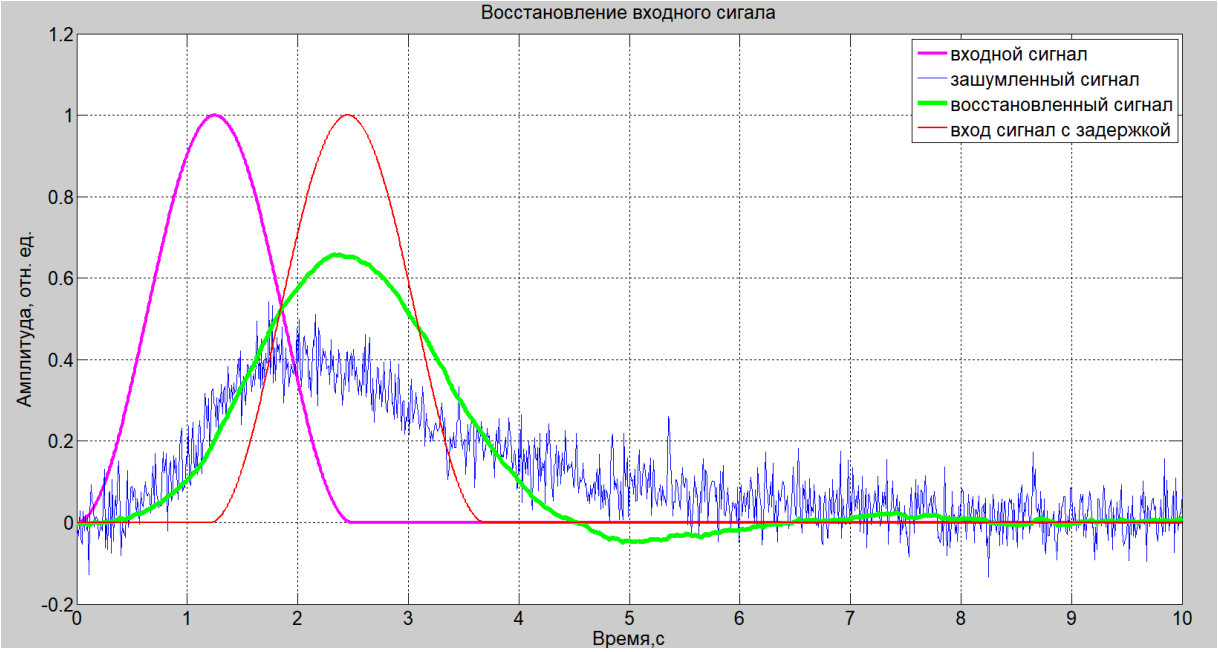


Рисунок 4.16− График восстановленного сигнала при произвольном параметре

Как видно из рисунка 4.16 сигнал восстанавливается неточно, необходимо подобрать параметр НС, который будет оптимально восстанавливать входной сигнал.

Теоретическая оценка СКО находится по формуле (4.1). Расчетная оценка СКО находится по формуле (4.2).

На рисунке 4.17. показан график СКО востановленного сигнала, зависящий от параметра НС.

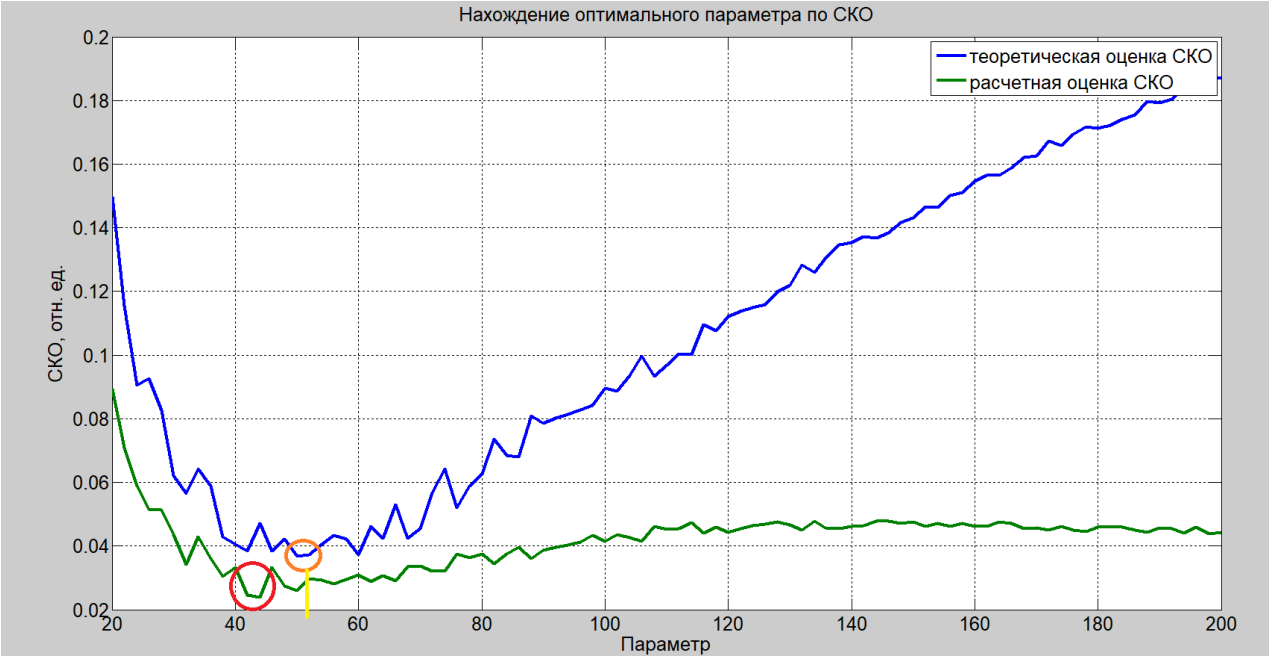


Рисунок 4.17− График СКО восстановленного сигнала

По рисунку 4.16 видно, что наименьшее значение СКО приобретает на интервале от 82 до 94.

Приведем значения теоретической оценки СКО и расчётной оценки СКО в Таблице 6.

Таблица 6 – СКО оценок динамической погрешности

Значение коэффициента	Теоретическая оценка	Расчётная оценка
36	0.0588	0.0360
38	0.0428	0.0305
40	0.0406	0.0333
42	0.0385	0.0246
44	**0.0472**	**0.0238**
46	0.0384	0.0334

По результатам таблицы 6 видно, что самый оптимальный параметр 44.

На рисунке 4.18 построен восстановленный сигнал с уже уточнённым параметром.

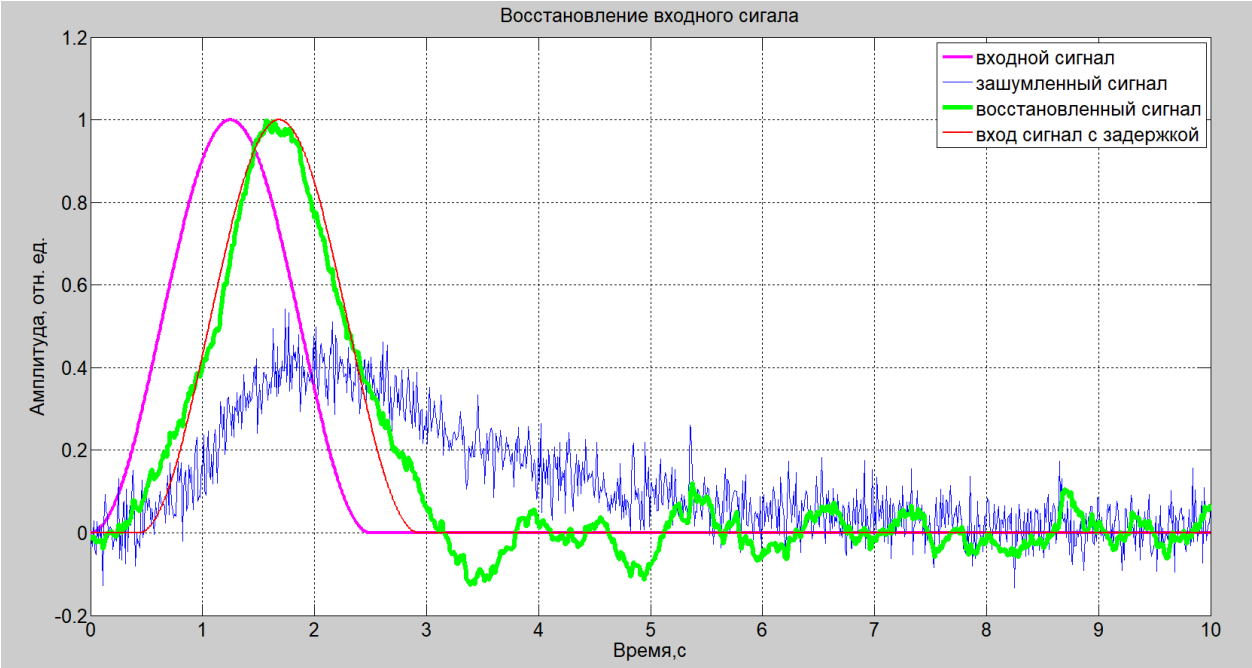


Рисунок 4.18 – График восстановленного сигнала с оптимальным параметром

Произведем сравнение полученных оценок:

Произведем сравнение полученных оценок:

$$\frac{\sigma_0}{\sigma} = \frac{0{,}0472}{0{,}0232} = 2. \qquad (4.15)$$

Из полученного значения можно сделать вывод о том, что СКО уменьшилось в 2 раз, это говорит о том, что мы достигли решения поставленной цели выпускной квалификационной работы.

4.4 Моделирование реальных данных со стенда датчика температуры

Одной из самых важных тестирований работы алгоритма является его проверка при использовании реальными данными, поскольку создание алгоритмов направлены на получения пользы в реальной жизни.

Были взяты реальные данные с эксперимента термопреобразователя Метран-281. В течение пятисот секунд чувствительный элемент преобразователя нагревали от 0 до 800 °С. Данные полученные из этого эксперимента были обработаны и использованы при моделировании алгоритма коррекции динамической погрешности.

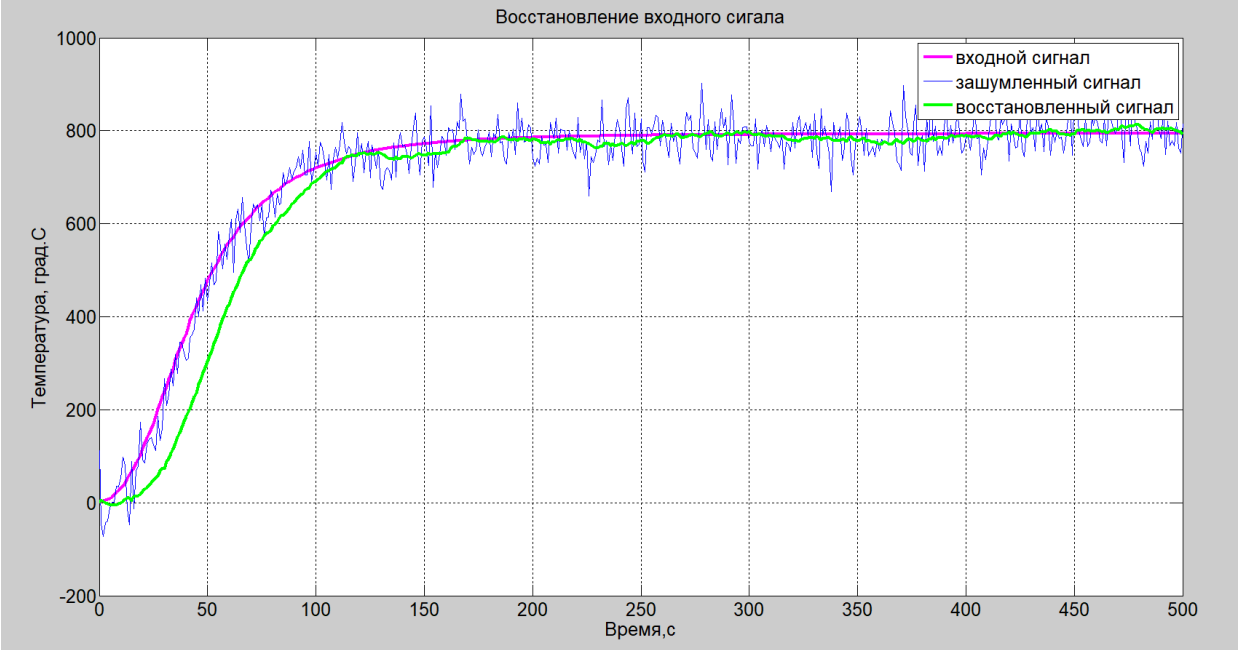


Рисунок 4.19− График восстановленного сигнала при произвольном параметре

Как видно из рисунка 4.19 сигнал восстанавливается неточно, необходимо подобрать параметр НС, который будет оптимально восстанавливать входной сигнал.

Теоретическая оценка СКО находится по формуле (4.1). Расчетная оценка СКО находится по формуле (4.2).

На рисунке 4.20 показан график СКО востановленного сигнала, зависящий от параметра НС.

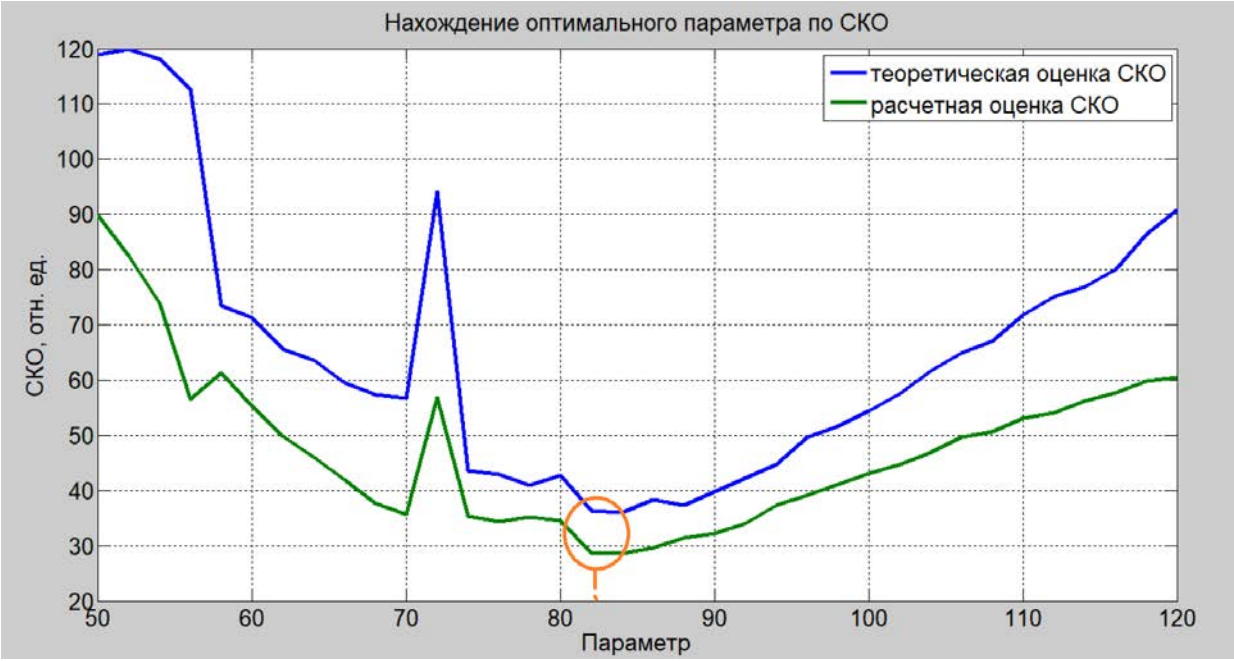


Рисунок 4.50− График СКО восстановленного сигнала

По рисунку 4.20 видно, что наименьшее значение СКО приобретает на интервале от 78 до 90.

Приведем значения теоретической оценки СКО и расчётной оценки СКО в Таблице 7.

Таблица 7 – СКО оценок динамической погрешности

Значение коэффициента	Теоретическая оценка	Расчётная оценка
78	40.9919	35.1511
80	42.7414	34.5208
82	**36.2853**	**28.6783**
84	38.3842	29.5625
86	37.3386	31.3888
88	39.8362	32.1711
90	42.2447	33.9770

По результатам Таблицы 7 видно, что самый оптимальный параметр 82.

На рисунке 4.21 построен восстановленный сигнал с уже уточнённым

параметром.

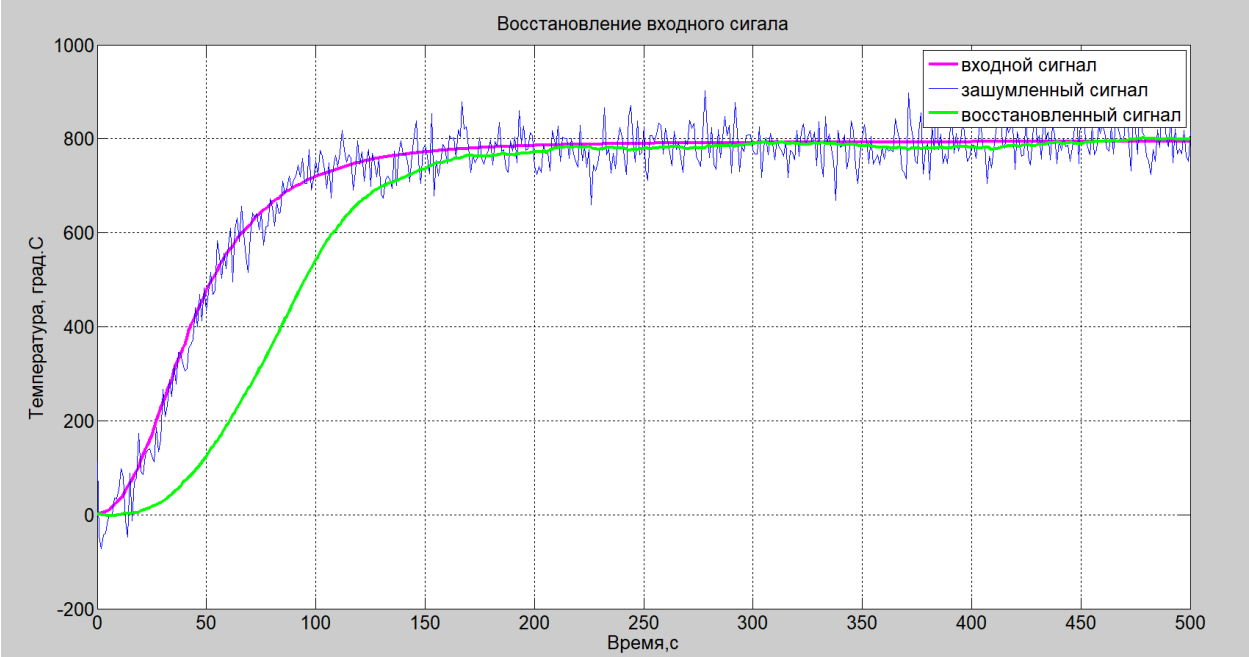


Рисунок 4.21− График восстановленного сигнала с оптимальным параметром

Произведем сравнение полученных оценок:

$$\frac{\sigma_0}{\sigma} = \frac{36{,}285}{28{,}678} = 1{,}26 \qquad (4.15)$$

Из полученного значения можно сделать вывод о том, что СКО уменьшилось в 1,26 раз, это говорит о том, что мы достигли решения поставленной цели выпускной квалификационной работы.

Таблица 8 – Результаты погрешности, полученные при моделировании

	Входной сигнал					
	Гармонический сигнал		Ступенчатый сигнал		Импульсный сигнал	
Шум	СКО расч, отн. ед	Оптим. парам.	СКО расч, отн. ед	Оптим. парам.	СКО расч, отн. ед	Оптим. парам.
Синусоидальный	0.0722	21	0.0279	34	0.0273	34
	0.0360	23	0.0163	36	0.0161	36
	0.0112	**25**	**0.0129**	**38**	**0.0133**	**38**
	0.0282	27	0.0191	40	0.0197	40
	0.0411	29	0.0238	42	0.0245	42
Случайный	0.0320	41	0.0199	70	0.0360	36
	0.0306	43	0.0211	72	0.0305	38
	0.0293	**45**	0.0222	74	0.0333	40
	0.0314	47	**0.0193**	**76**	0.0246	42
	0.0323	49	0.0213	78	**0.0238**	**44**
	0.0320	51	0.0214	80	0.0334	46

Выводы по 4 главе

По результатам таблицы 8 можно сделать вывод, что динамическую погрешность удалось уменьшить за счёт работы алгоритма коррекции в два, три раза, значит что, разработанный алгоритм эффективен. При случайном шуме соответственно погрешность будет больше, поскольку Белый Гауссов шум вносит большее искажение, что затрудняет фильтрацию и восстановление сигнала.

ЗАКЛЮЧЕНИЕ

В ходе выполнения выпускной квалификационной работы были получены следующие результаты:

1. В первой главе были рассмотрены методы, подходы и алгоритмы восстановления измерительных сигналов. Выявлены положительные качества и недостатки. В процессе аналитического обзора были изучены книги, статьи, монографии и другие источники методов восстановления измерительных сигналов.
2. Во второй главе были нейронные сети. Были рассмотрены модели НС, разобраны их достоинства и недостатки. Выбрана динамическая нейронная сеть для выполнения поставленной задачи.
3. В третьей главе разработан алгоритм коррекции динамической погрешности, реализованный в виде кода в среде MATLAB. Разработана блок-схема данного алгоритма и найдены оптимальные параметры нейронной сети.
4. В четвёртой главе показаны результаты моделирования в программе MATLAB. Составлена сводная таблица результатов, с помощью которой можно сказать о коррекции погрешности каждого сигнала.

Главная цель работы, а именно, уменьшение динамической погрешности выполнена.

С помощью алгоритма удалось наиболее качественно восстановить гармонический сигнал. Ступенчатый и импульсный сигналы также хорошо восстанавливаются, но всё же не так идеально как гармонический сигнал. Восстановление сигнала при случайном шуме сильно усложняет работу алгоритма. Из-за особенностей Белого Гауссового шума сигнал хуже фильтруется ,чем при синусоидальном шуме.

Таким образом, цель выпускной квалификационной работы, можно считать достигнутой.

БИБЛИОГРАФИЧЕСКИЙ СПИСОК

1. Бизяев, М.Н. Динамические модели и алгоритмы восстановления динамически искаженных сигналов измерительных систем в скользящем режиме: дис. … канд. техн. наук / М.Н. Бизяев. – Челябинск, 2004. – 177 с.

2. Бизяев, М.Н. Динамические модели и алгоритмы восстановления динамически искаженных сигналов измерительных систем в скользящем режиме / М.Н. Бизяев // Энергетика. Серия «Математика». – 2004. – №6. С.119.

3. Воскобойников, Ю.Е. Восстановление реализаций входных сигналов измерительной системы /Ю.Е. Воскобойников, Я.Я. Томсон // Электродиффузионная диагностика турбулентных потоков. — Новосибирск: Ин-т теплофизики СО АН СССР, 1973.—С. 66-96.

4. ГОСТ 8.009-84. Нормируемые метрологические характеристики средств измерений. – М.: Изд-во стандартов, 1986. – 27 с.

5. Грановский, В.А. Динамические измерения: Основы метрологичекого обеспечения / В.А. Грановский. – Л.: Энергоатомоиздат, 1984. – 224 с.

6. Иосифов, Д.Ю. Динамические модели и алгоритмы восстановления сигналов измерительных систем с наблюдаемым вектором координат состояния: дис. … канд. техн. наук / Д.Ю. Иосифов. – Челябинск, 2007 —164 с.

7. Карпов, Н.Е. Применение методов восстановления сигналов в системах защиты информации / Н.Е. Карпов // Общероссийского математического портал. MathNet.Ru. – 2006. Часть 4. – С.51– 54.

8. Королёва, К.А. Восстановление пропущенных значений сигнала во время калибровки измерительных систем / К.А. Королёва //Омский государственный университет путей и сообщений. Омский научный вестник. – 2013. – Вып.2. – №1(127) – С.188-192.

9. Кохонен, Т. Самоорганизующеся карты. / Т. Кохонен; пер.с анг. В. Агеева; ред. Ю.Тюменцев. — М.: Бином. Лаборатория знания, 2017. — 656с. — ISBN 978-5-94774-352-4

10. Меркушева, А.В. Восстановление линейно смешанных сигналов на основе адаптивного алгоритма рекуррентной сети / А.А. Меркушева, Г.Ф. Малыхина // Научное приборостроение. – 2005. – Том 15. – №3 – С. 94-107.

11. Нейронные сети. MATLAB 6: учебное пособие / В. С. Медведев, В. Г. Потемкин; под ред. В. Г. Потемкина. – 2-е изд., перераб. И доп. – М.: ДИАЛОГМИФИ, 2002. – 296 с.

12. Тихонов А.Н. Методы решения некорректных задач / А.Н. Тихонов, В.А. Арсенин. — М.: Наука, 1974.—222 с.

13. Хрящев, В.В. Анализ нейронных сетей в некоторых задачах цифровой обработки изображений: автореферат дис. …– д-ра техн. наук / В.В. Хрящев. – Ярославль: Изд-во. Ярослаль, 2004. – 15 с.

14. Хрящев В.В., Соколенко Е.А., Приоров А.Л. Нейросетевое восстановление амплитуды дискретного сигнала по его фазовому спектру // В.В. Хрящев, Е.А. Соколенко, А.Л. Приоров / Докл. 5-ой междунар. конф. и выставки «Цифровая обработка сигналов и ее применение». М:, 2003. – 622-625 с.

15. Шестаков, А.Л. Методы теории автоматического управления в динамических измерениях / А.Л. Шестаков. – Челябинск: Изд. центр ЮУрГУ, 2013. – 257 с.

16. Ян, Г. Глубокое обучение/ Б. Иошуа, Г.Ян, К. Аарон. – М.: ДМКПресс, 2018. — 652с. — ISBN 978-5-97060-618-6

17. Cohen, A.S. The Gaussian watermarking game / A.S. Cohen, A. Lapidoth // IEEE Trans. Info. Theory. – 2002. – V.48. – P. 1639-1667.

18. Frank, E. Fully supervised training of Gaussian radial basis function networks in WEKA (Computer Science Working Papers, 04/2014) / E. Frank // Hamilton, NZ: Department of Computer Science, The University of Waikato. – 2014. – P. 1-5.

19. Gamiy, V.A. Measuring Transducer of Dynamic Parameters / V.A. Gamiy, V.A. Koshcheev, A.L. Shestakov // Discoveries and inventions. – 1991. – №12. P. 191.

20. Interpolation and the Discrete PapoulisGerchberg Algorithm. – http://citeseerx.ist.psu.edu.html.

21. Li, X. Signal recovery using neural networks/ X. Li, M.Bodruzzman, C.Wang // N.: Department of Electrical Engineering. – 2007. – P.75-78.

22. Marques, M. The Papoulis-Gerchberg algorithm with unknown signal bandwidth / M. Marques // Image Analysis and Recognition. – Springer Berlin Heidelberg. – 2006. – P. 436-445.

23. McCulloch, W.S. A logical calculus of the ideas immanent in nervous activity / W.S. McCulloch, W. Pitts // Bulletin of Mathematical Biophysics. – 1943 – V. 5 – P. 115–133.

24. McCulloch, W.S. A logical calculus of the ideas immanent in nervous activity / W.S. McCulloch, W. Pitts // Bulletin of Mathematical Biophysics. – 1943 – V. 9 – P. 127–147.

25. Minsky M. Perceptrons / M. Minsky, S. Papert. – MIT Press, 1969.

26. Papoulis, A. A new algorithm in spectral analysis and band – 307 limited extrapolation / A.Papoulis // IEEE Trans. circuits. syst. – 1975. – CAS22. – 742 p.

27. Robert J. Marks 11. Convergence of Howard's minimum-negativityconstraint extrapolation algorithm / F. Kwan Cheung, Robert J. Marks 11, Les E. Atlas // Journal of the Optical Society of America A, vol.5. – 1988. – p. 208-209. – P 215.

28. Signal processing toolbox. – https://uk.mathworks.com/help/signal/index.html.

29. Schmidhuber, J., Deep learning in neural networks: An overview// Neural Networks. — 2015. —№61. — P.85-117.

30. Shestakov, A.L. Measuring Transducer of Dynamic Parameters / A.L. Shestakov // Discoveries and inventions. – 1990. – №22. P. 192.

31. Que, Q. Back to the future: Radial basis function networks revisited. In AISTATS /Q. Que, M. Belkin // Department of Computer Science and Engineering. – 2016.– P. 1376-1383.

ПРИЛОЖЕНИЕ А

Листинг программы

```
clear; clc;
t=0:0.01:20;
rng default; %инициализация генератора случайных чисел
W=tf(1,[2 1]);
U=1*sin(2*3.14*0.2*t);% синусоида как ВХОД
Av = 0.03;
V=3*Av*randn(1,length (t));% белый гауссовкий шум


A=NaN(3,80);
i=1;
for p=20:2:120
A(1,i)=p;
r=p;
yp = ones(size(t)); yp(1:2*r+1) = 0.5-0.5*cos(pi*(0:1/(2*r):1));
u = step(W,t)';
VHn = num2cell(u);%Входная обучающая выборка
VUHn = num2cell(yp);%Целевая обучающая выборка
%p=50;

%net = network(1,1,0,1,1,1);% Создание нейронной сети с обратной
связью(однойслойная)
net = network(1,2, [0; 0], [1; 0], [1 0; 1 0], [0 1]); %
двухслойная с обратной связью первого слоя
%net = network(1,2, [0; 0], [1; 0], [1 1; 1 0], [0 1])%  двухслойная
с обратной связью первого слоя
net.layers{1}.size = 1;
net.layers{1}.initFcn = 'initnw';
net.layers{2}.initFcn = 'initnw';
net.layerWeights{1,1}.delays=[1 p];
net.inputWeights{1,1}.delays =[0 p];
net.inputWeights{1,1}.learnFcn = 'learngdm';
  %net.inputWeights{0,1}.delays = [0 2];

net.layerWeights{2,1}.delays=[1 p];
net.inputWeights{2,1}.learnFcn = 'learngdm';
%net.biasConnect=1;10;%логический вектор numLayers-by-1, нули.
 %net.layers{1}.transferFcn = 'tansig';
    % net.layers{2}.transferFcn = 'logsig';
net.trainFcn = 'trainlm';
net.trainParam.epochs=200;
[net,tr]=train(net,VHn,VUHn);


%  Us=ones(size(t));
```

```
Us = 1*sin(2*3.14*0.2*t); %подаваемый входной сигнал

% Us=(Us.^2);
% for j=1
% Us(j)=1*sin(2*3.14*0.2*(j-1));
%
% end
% for j=250:1001
% Us(j)=0;
% end
%Yss = lsim(W,Us,t)'+0.05*sin(2*3.14*0.2*40*t);%Выход датчика
Yss = lsim(W,Us,t)'+0.05*randn(1,length (t));
Pss = num2cell(Yss);
Ur = sim(net,Pss);
Ur = cell2mat(Ur);

upp=lsim(W,Ur,t)';
Uvost = num2cell(upp);
Uff = sim(net,Uvost);
Uff = cell2mat(Uff);




Uppp= [zeros(1,p) Ur(1:end-p)];


Uss= [zeros(1,p) Us(1:end-p)];
A(2,i) = std(Ur-Uss); %СКО идеальная оценка
A(3,i)= std(Uff-Uppp); %СКО приблизительная оценка
i=i+1;
end


figure;
plot(t,Us,'m','linewidth',3)
hold on;
%plot(t,Ys,'r')
%hold on;
plot(t,Yss,'b')
hold on;
plot(t,Ur,'g','linewidth',3)
hold on;
plot(t,Uss,'r','linewidth',3)
hold on;

legend('входной сигнал','зашумленный сигнал','восстановленный
сигнал','задержка восстановленного сигнала','задержка входного
сигнала' );
set(gca,'FontName','Arial Cyr','FontSize',20)
```

```
grid;

set(gca,'FontName','Arial Cyr','FontSize',20)
grid; figure; figure(2);
plot(A(1,:),A(2,:),A(1,:),A(3,:),'linewidth',3);
hold on;
set(gca,'FontName','Arial Cyr','FontSize',20)
title('Нахождение оптимального параметра по СКО');
xlabel('Параметр')
ylabel('СКО, отн. ед.')
legend('теоретическая оценка СКО','расчетная оценка СКО');

grid;
```

ПРИЛОЖЕНИЕ Б

Листинг программы

```
%Алгоритм для восстановления сигнала:

%1. Формирование начальных данных : время ,входной сигнал, задание
шума:
clear; clc;
t=0:0.01:20; %задание интервала времени
rng default; %инициализация генератора случайных чисел
W=tf(1,[2 1]); % описание передаточной функции
U=1*sin(2*3.14*0.2*t);% здание входного сигнала в виде синусоиды
Av = 0.03;
V=3*Av*randn(1,length (t));% наложения шума в виде белого гауссова
шума
%y = step(W,t)';%переходная характеристика датчика
%y = ones(size(t));%переходная характеристика датчика
%y = sin(2*3.14*0.2*t);%переходная характеристика датчика

%2. Подготовка обучающей последовательности:
r=50; %- параметр задержки
%yp=[zeros(1,r) y(1:end-r)]; % переходная характеристика датчика с
задержкой
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%
%Новая обучающая выборка
yp = ones(size(t)); yp(1:2*r+1) = 0.5-0.5*cos(pi*(0:1/(2*r):1));
u = step(W,t)';
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%

%Сформирование обучающих последовательностей в виде массивов ячеек:
VHn = num2cell(u);%Входная обучающая выборка
VUHn = num2cell(yp);%Целевая обучающая выборка

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%
p=r; %параметр равный задержке
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%

%3. Создание нейронной сети и её настройка:
%net = network(1,1,0,1,1,1);% Создание нейронной сети с обратной
связью(однойслойная)
```

```
net = network(1,2, [0; 0], [1; 0], [1 0; 1 0], [0 1]); %
двухслойная с обратной связью первого слоя
%net = network(1,2, [0; 0], [1; 0], [1 1; 1 0], [0 1])%  двухслойная
с обратной связью первого слоя
net.layers{1}.size = 1;
net.layers{1}.initFcn = 'initnw';


net.layers{2}.initFcn = 'initnw';
net.layerWeights{1,1}.delays=[1 p];

net.inputWeights{1,1}.delays =[0 p];


net.inputWeights{1,1}.learnFcn = 'learngdm';
% net.layers{1}.transferFcn = 'tansig';
 % net.layers{1}.transferFcn = 'logsig';
net.layerWeights{2,1}.delays=[1 p];
net.inputWeights{2,1}.learnFcn = 'learngdm';
net.trainFcn = 'trainlm'; % функция тренировки

[net,tr]=train(net,VHn,VUHn);

%4. Подача входных сигналов(ступенчатый, синусоидальный и
импульсный) и наложение шума:
% Us=ones(size(t)); % Ступеначатый входной сигнал
Us = 1*sin(2*3.14*0.2*t); %синусоидальный входной сигнал
%
%
% Us=(Us.^2);
% for i=1
% Us(i)=1*sin(2*3.14*0.2*(i-1));
% end
% for i=250:1001
% Us(i)=0;
% end

%Yss = lsim(W,Us,t)'+0.05*sin(2*3.14*0.2*40*t);% Выходной сигнал с
гармоническим шумом
 Yss = lsim(W,Us,t)'+0.05*randn(1,length (t)); % Выходной сигнал со
случайны  шумом

%5. Восстановление входного сигнала:
Pss = num2cell(Yss);
Ur = sim(net,Pss);
Ur = cell2mat(Ur);
Uss= [zeros(1,r) Us(1:end-r)]; % Задержка входного сигнала
```

```
%6. Построение графиков:
figure;
plot(t,Us,'m','linewidth',3)
hold on;
plot(t,Yss,'b');
hold on;
plot(t,Ur,'g','linewidth',5)
hold on;
plot(t,Uss,'r','linewidth',2)
hold on;
legend('входной сигнал','зашумленный сигнал','восстановленный сигнал
','вход сигнал с задержкой');
set(gca,'FontName','Arial Cyr','FontSize',20)
title('Восстановление входного сигала');
xlabel('Время,с');
ylabel('Амплитуда, отн. ед.');
grid;
```

ПРИЛОЖЕНИЕ В

Листинг программы

```
%Алгоритм для восстановления сигнала:

%1. Формирование начальных данных : время ,входной сигнал, задание
шума:
clear; clc;
t=0:1:500; %задание интервала времени
rng default; %инициализация генератора случайных чисел
W=tf(1,[2 1]); % описание передаточной функции
U=1*sin(2*3.14*0.2*t);% здание входного сигнала в виде синусоиды
Av = 0.03;
V=3*Av*randn(1,length (t));% наложения шума в виде белого гауссова
шума
%y = step(W,t)';%переходная характеристика датчика
%y = ones(size(t));%переходная характеристика датчика
%y = sin(2*3.14*0.2*t);%переходная характеристика датчика

%2. Подготовка обучающей последовательности:
r=38; %- параметр задержки
%yp=[zeros(1,r) y(1:end-r)]; % переходная характеристика датчика с
задержкой
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%
%Новая обучающая выборка
yp = ones(size(t)); yp(1:2*r+1) = 0.5-0.5*cos(pi*(0:1/(2*r):1));
u = step(W,t)';
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%

%Сформирование обучающих последовательностей в виде массивов ячеек:
VHn = num2cell(u);%Входная обучающая выборка
VUHn = num2cell(yp);%Целевая обучающая выборка

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%
p=r; %параметр равный задержке
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%

%3. Создание нейронной сети и её настройка:
%3. Создание нейронной сети и её настройка:
%net = network(1,1,0,1,1,1);% Создание нейронной сети с обратной
связью(однойслойная)
net = network(1,2, [0; 0], [1; 0], [1 0; 1 0], [0 1]); %
двухслойная с обратной связью первого слоя
```

```
%net = network(1,2, [0; 0], [1; 0], [1 1; 1 0], [0 1])%  двухслойная
с обратной связью первого слоя
net.layers{1}.size = 1;
net.layers{1}.initFcn = 'initnw';


net.layers{2}.initFcn = 'initnw';
net.layerWeights{1,1}.delays=[1 p];
net.inputWeights{1,1}.delays =[0 p];


net.inputWeights{1,1}.learnFcn = 'learngdm';
net.layerWeights{2,1}.delays=[1 p];
net.inputWeights{2,1}.learnFcn = 'learngdm';
net.trainFcn = 'trainlm'; % функция тренировки

[net,tr]=train(net,VHn,VUHn);

%interp1(VarName1,VarName2,1);


%Us=ones(size(t));
a0=13.859715;
a1=-5.278544;
a2=0.787682;
a3=-0.015852;
a4=0.00151349;
a5=-0.000000751187;
a6=0.0000000013563949;
a7=0.00000000000539664;
a8=-0.00000000000004129;
a9=0.00000000000000012;
a10=-0.0000191535*(10.^(-14));
a11=0.0000019006729126*(10.^(-14));
a12=-0.00000000005872*(10.^(-14));
%Us=a0*t.^0+a1*t+a2+(t.^2)+a3*(t.^3)+a4*(t.^4)+a5*(t.^5)+a6*(t.^6)+a
7*(t.^7)+a8*(t.^8)+a9*(t.^9)+a10*(t.^10)+a11*(t.^11)+a12*(t.^12);
%Us=794*(1-2.71182818.^(-t/50));
%Us=794*(1.-exp(-t/50));
Op = [1.1837;
2.6931;
4.1538;
5.6555;
7.4325;
9.4653;
11.6909;
16.4274;
22.2868;
26.2811;
31.1558;
```

35.9758;
42.0104;
52.8173;
59.9222;
67.5600;
74.5519;
84.8692;
92.6900;
100.5392;
116.1165;
128.4392;
138.7679;
148.5865;
159.6583;
169.8424;
185.6703;
201.3838;
213.3491;
227.3330;
238.7309;
250.4307;
264.8360;
275.8073;
287.2898;
307.4801;
320.3325;
331.5280;
341.8941;
352.8949;
362.3767;
379.8807;
391.7272;
402.4963;
412.7400;
422.5388;
434.1243;
444.6927;
452.7578;
463.7321;
478.8701;
487.9671;
496.1029;
503.6230;
511.8141;
518.7896;
531.7172;
538.6288;
547.4312;
554.7434;
560.6427;

```
572.3123;
587.7276;
598.4787;
608.5552;
618.2225;
631.1346;
639.5649;
648.0225;
654.6804;
665.3496;
671.7423;
677.7677;
685.7047;
691.7625;
696.9260;
702.2473;
706.2829;
712.3060;
716.5978;
719.9887;
750.6958;
766.7929;
776.4281;
782.1241;
785.6783;
787.9309;
789.4838;
790.5915;
791.2872;
791.8782;
792.2941;
792.6870;
792.9784;
793.2754;
793.4909;
793.7480;
793.8527;
794.0449;
794.1171;
794.2437]';
x = [0:59 60:2:98 100:20:500];
xx = 0:500;
Us = csapi(x,Op,xx);


%Us = 1*sin(2*3.14*0.2*t); %синусоида как шум


%Us=(Us.^2);
```

```
%for i=1
%Us(i)=1*sin(2*3.14*0.2*(i-1));
%end
%for i=250:1001
%Us(i)=0;
%end
%Yss = lsim(W,Us,t)'+10*sin(2*3.14*500*t);%Выход датчика
Yss = lsim(W,Us,t)'+40*randn(1,length (t));
Pss = num2cell(Yss);%Преобразование выходного сигнала в ячейки
Ur = sim(net,Pss);% Полученный выход сети
Ur = cell2mat(Ur);%Преобразование выхода в вектор

figure;
%plot(t,U,'k','linewidth',3)
%hold on;
plot(t,Us,'m','linewidth',3)
hold on;
%plot(t,Ys,'r')
%hold on;
plot(t,Yss,'b')
hold on;
plot(t,Ur,'g','linewidth',3)
hold on;
legend('входной сигнал','зашумленный сигнал','восстановленный сигнал
','выход сети' );
set(gca,'FontName','Arial Cyr','FontSize',20)
title('Восстановление входного сигала');
xlabel('Время,с')
ylabel('Температура, град.С')



grid;
```

ПРИЛОЖЕНИЕ Г

Листинг программы

```
clear; clc;
t=0:1:500;
rng default; %инициализация генератора случайных чисел
W=tf(1,[2 1]);
U=1*sin(2*3.14*0.2*t);% синусоида как ВХОД
Av = 0.03;
V=3*Av*randn(1,length (t));% белый гауссовкий шум


A=NaN(3,80);
i=1;
for p=20:2:120
A(1,i)=p;
r=p;
yp = ones(size(t)); yp(1:2*r+1) = 0.5-0.5*cos(pi*(0:1/(2*r):1));
u = step(W,t)';
VHn = num2cell(u);%Входная обучающая выборка
VUHn = num2cell(yp);%Целевая обучающая выборка
%p=50;

%net = network(1,1,0,1,1,1);% Создание нейронной сети с обратной
связью(однойслойная)
net = network(1,2, [0; 0], [1; 0], [1 0; 1 0], [0 1]); %
двухслойная с обратной связью первого слоя
%net = network(1,2, [0; 0], [1; 0], [1 1; 1 0], [0 1])%  двухслойная
с обратной связью первого слоя
net.layers{1}.size = 1;
net.layers{1}.initFcn = 'initnw';
net.layers{2}.initFcn = 'initnw';
net.layerWeights{1,1}.delays=[1 p];
net.inputWeights{1,1}.delays =[0 p];
net.inputWeights{1,1}.learnFcn = 'learngdm';
  %net.inputWeights{0,1}.delays = [0 2];

net.layerWeights{2,1}.delays=[1 p];
net.inputWeights{2,1}.learnFcn = 'learngdm';
net.trainFcn = 'trainlm';
net.trainParam.epochs=200;
[net,tr]=train(net,VHn,VUHn);

Op = [1.1837;
2.6931;
```

```
4.1538;
5.6555;
7.4325;
9.4653;
11.6909;
16.4274;
22.2868;
26.2811;
31.1558;
35.9758;
42.0104;
52.8173;
59.9222;
67.5600;
74.5519;
84.8692;
92.6900;
100.5392;
116.1165;
128.4392;
138.7679;
148.5865;
159.6583;
169.8424;
185.6703;
201.3838;
213.3491;
227.3330;
238.7309;
250.4307;
264.8360;
275.8073;
287.2898;
307.4801;
320.3325;
331.5280;
341.8941;
352.8949;
362.3767;
379.8807;
391.7272;
402.4963;
412.7400;
422.5388;
434.1243;
444.6927;
452.7578;
463.7321;
478.8701;
487.9671;
```

```
496.1029;
503.6230;
511.8141;
518.7896;
531.7172;
538.6288;
547.4312;
554.7434;
560.6427;
572.3123;
587.7276;
598.4787;
608.5552;
618.2225;
631.1346;
639.5649;
648.0225;
654.6804;
665.3496;
671.7423;
677.7677;
685.7047;
691.7625;
696.9260;
702.2473;
706.2829;
712.3060;
716.5978;
719.9887;
750.6958;
766.7929;
776.4281;
782.1241;
785.6783;
787.9309;
789.4838;
790.5915;
791.2872;
791.8782;
792.2941;
792.6870;
792.9784;
793.2754;
793.4909;
793.7480;
793.8527;
794.0449;
794.1171;
794.2437]';
x = [0:59 60:2:98 100:20:500];
```

```
xx = 0:500;
Us = csapi(x,Op,xx);

%Us=794*(1.-exp(-t/50));
%Us=ones(size(t));
% Us = 1*sin(2*3.14*0.2*t); %подаваемый входной сигнал
%
% Us=(Us.^2);
% for j=1
% Us(j)=1*sin(2*3.14*0.2*(j-1));
%
% end
% for j=250:1001
% Us(j)=0;
% end
%Yss = lsim(W,Us,t)'+0.05*sin(2*3.14*0.2*40*t);%Выход датчика
Yss = lsim(W,Us,t)'+40*randn(1,length (t));
Pss = num2cell(Yss);
Ur = sim(net,Pss);
Ur = cell2mat(Ur);
upp=lsim(W,Ur,t)';
Uvost = num2cell(upp);
Uff = sim(net,Uvost);
Uff = cell2mat(Uff);
Uppp= [zeros(1,p) Ur(1:end-p)];


Uss= [zeros(1,p) Us(1:end-p)];
A(2,i) = std(Ur-Uss); %СКО идеальная оценка
A(3,i)= std(Uff-Uppp); %СКО приблизительная оценка
i=i+1;
end
figure;
plot(t,Us,'m','linewidth',3)
hold on;
%plot(t,Ys,'r')
%hold on;
plot(t,Yss,'b')
hold on;
plot(t,Ur,'g','linewidth',3)
hold on;
plot(t,Uss,'r','linewidth',3)
hold on;

legend('входной сигнал','зашумленный сигнал','восстановленный
сигнал','задержка восстановленного сигнала','задержка входного
сигнала' );
set(gca,'FontName','Arial Cyr','FontSize',20)
grid;
```

```
set(gca,'FontName','Arial Cyr','FontSize',20)
grid; figure; figure(2);
plot(A(1,:),A(2,:),A(1,:),A(3,:),'linewidth',3);
hold on;
set(gca,'FontName','Arial Cyr','FontSize',20)
title('Нахождение оптимального параметра по СКО');
xlabel('Параметр')
ylabel('СКО, отн. ед.')
legend('теоретическая оценка СКО','расчетная оценка СКО');

grid;
```

ПРИЛОЖЕНИЕ Д

Листинг программы

```
clear; clc;
t=0:0.01:20;
rng default; %инициализация генератора случайных чисел
W=tf(1,[2 1]);
U=1*sin(2*3.14*0.2*t);% синусоида как ВХОД
Av = 0.03;
V=3*Av*randn(1,length (t));% белый гауссовкий шум


A=NaN(3,80);
i=1;
for p=20:2:120
A(1,i)=p;
r=p;
yp = ones(size(t)); yp(1:2*r+1) = 0.5-0.5*cos(pi*(0:1/(2*r):1));
u = step(W,t)';
VHn = num2cell(u);%Входная обучающая выборка
VUHn = num2cell(yp);%Целевая обучающая выборка
%p=50;

net = network(1,1,0,1,1,1);% Создание нейронной сети с обратной
связью(однойслойная)
%net = network(1,2, [0; 0], [1; 0], [1 0; 1 0], [0 1]); %
двухслойная с обратной связью первого слоя
%net = network(1,2, [0; 0], [1; 0], [1 1; 1 0], [0 1])%  двухслойная
с обратной связью первого слоя

net.layerWeights{1,1}.delays=[1 p];
net.inputWeights{1,1}.delays =[0 p];
net.inputWeights{1,1}.learnFcn = 'learngdm';
  %net.inputWeights{0,1}.delays = [0 2];


%net.biasConnect=1;10;%логический вектор numLayers-by-1, нули.
 %net.layers{1}.transferFcn = 'tansig';
    % net.layers{2}.transferFcn = 'logsig';
net.trainFcn = 'trainlm';
net.trainParam.epochs=200;
[net,tr]=train(net,VHn,VUHn);


%  Us=ones(size(t));
Us = 1*sin(2*3.14*0.2*t); %подаваемый входной сигнал

% Us=(Us.^2);
```

```
% for j=1
% Us(j)=1*sin(2*3.14*0.2*(j-1));
%
% end
% for j=250:1001
% Us(j)=0;
% end
%Yss = lsim(W,Us,t)'+0.05*sin(2*3.14*0.2*40*t);%Выход датчика
Yss = lsim(W,Us,t)'+0.05*randn(1,length (t));
Ps = num2cell(Yss);
Ur = sim(net,Ps);
Ur = cell2mat(Ur);

upp=lsim(W,Ur,t)';
Uvost = num2cell(upp);
Uff = sim(net,Uvost);
Uff = cell2mat(Uff);
Uppp= [zeros(1,p) Ur(1:end-p)];


Uss= [zeros(1,p) Us(1:end-p)];
A(2,i) = std(Ur-Uss); %СКО идеальная оценка
A(3,i)= std(Uff-Uppp); %СКО приблизительная оценка
i=i+1;
end


figure;
plot(t,Us,'m','linewidth',3)
hold on;
%plot(t,Ys,'r')
%hold on;
plot(t,Yss,'b')
hold on;
plot(t,Ur,'g','linewidth',3)
hold on;
plot(t,Uss,'r','linewidth',3)
hold on;

legend('входной сигнал','зашумленный сигнал','восстановленный
сигнал','задержка восстановленного сигнала','задержка входного
сигнала' );
set(gca,'FontName','Arial Cyr','FontSize',20)
grid;

set(gca,'FontName','Arial Cyr','FontSize',20)
grid; figure; figure(2);
plot(A(1,:),A(2,:),A(1,:),A(3,:),'linewidth',3);
hold on;
set(gca,'FontName','Arial Cyr','FontSize',20)
```

```
title('Нахождение оптимального параметра по СКО');
xlabel('Параметр')
ylabel('СКО, отн. ед.')
legend('теоретическая оценка СКО','расчетная оценка СКО');

grid;
```

ПРИЛОЖЕНИЕ Е

Листинг программы

```
rng default; %инициализация генератора случайных чисел
W=tf(1,[2 1]);
U=1*sin(2*3.14*0.2*t);% синусоида как ВХОД
Av = 0.03;
V=3*Av*randn(1,length (t));% белый гауссовкий шум


A=NaN(3,80);
i=1;
for p=20:2:120
A(1,i)=p;
r=p;
yp = ones(size(t)); yp(1:2*r+1) = 0.5-0.5*cos(pi*(0:1/(2*r):1));
u = step(W,t)';
VHn = num2cell(u);%Входная обучающая выборка
VUHn = num2cell(yp);%Целевая обучающая выборка
%p=50;

net = network(1,1,0,1,1,1);% Создание нейронной сети с обратной связью(однойслойная)
%net = network(1,2, [0; 0], [1; 0], [1 0; 1 0], [0 1]); % двухслойная с обратной связью первого слоя
%net = network(1,2, [0; 0], [1; 0], [1 1; 1 0], [0 1])% двухслойная с обратной связью первого слоя


net.layerWeights{1,1}.delays=[1 p];
net.inputWeights{1,1}.delays =[0 p];
net.inputWeights{1,1}.learnFcn = 'learngdm';
  %net.inputWeights{0,1}.delays = [0 2];



%net.biasConnect=1;10;%логический вектор numLayers-by-1, нули.
 %net.layers{1}.transferFcn = 'tansig';

net.trainFcn = 'trainlm';
net.trainParam.epochs=200;
```

```
[net,tr]=train(net,VHn,VUHn);


%  Us=ones(size(t));
Us = 1*sin(2*3.14*0.2*t); %подаваемый входной сигнал

% Us=(Us.^2);
% for j=1
% Us(j)=1*sin(2*3.14*0.2*(j-1));
%
% end
% for j=250:1001
% Us(j)=0;
% end
%Yss = lsim(W,Us,t)'+0.05*sin(2*3.14*0.2*40*t);%Выход датчика
Yss = lsim(W,Us,t)'+0.05*randn(1,length (t));
Ps = num2cell(Yss);
Ur = sim(net,Ps);
Ur = cell2mat(Ur);

upp=lsim(W,Ur,t)';
Uvost = num2cell(upp);
Uff = sim(net,Uvost);
Uff = cell2mat(Uff);
Uppp= [zeros(1,p) Ur(1:end-p)];


Uss= [zeros(1,p) Us(1:end-p)];
A(2,i) = std(Ur-Uss); %СКО идеальная оценка
A(3,i)= std(Uff-Uppp); %СКО приблизительная оценка
i=i+1;
end


figure;
plot(t,Us,'m','linewidth',3)
hold on;
%plot(t,Ys,'r')
%hold on;
plot(t,Yss,'b')
hold on;
plot(t,Ur,'g','linewidth',3)
```

```
hold on;
plot(t,Uss,'r','linewidth',3)
hold on;

legend('входной сигнал','зашумленный сигнал','восстановленный сигнал','задержка восстановленного сигнала','задержка входного сигнала' );
set(gca,'FontName','Arial Cyr','FontSize',20)
grid;

set(gca,'FontName','Arial Cyr','FontSize',20)
grid; figure; figure(2);
plot(A(1,:),A(2,:),A(1,:),A(3,:),'linewidth',3);
hold on;
set(gca,'FontName','Arial Cyr','FontSize',20)
title('Нахождение оптимального параметра по СКО');
xlabel('Параметр')
ylabel('СКО, отн. ед.')
legend('теоретическая оценка СКО','расчетная оценка СКО');

grid;
```