Министерство образования и науки Украины

Харьковский национальный университет радиоэлектроники

На правах рукописи

БЕССОНОВ АЛЕКСАНДР АЛЕКСАНДРОВИЧ

УДК 004.852 : 004.896

ЭВОЛЮЦИОНИРУЮЩИЕ ИСКУССТВЕННЫЕ НЕЙРОННЫЕ СЕТИ ПРЯМОГО РАСПРОСТРАНЕНИЯ: АРХИТЕКТУРЫ, ОБУЧЕНИЕ, ПРИМЕНЕНИЯ

05.13.23 – системы и средства искусственного интеллекта

Диссертация на соискание ученой
степени доктора технических наук

Научный консультант
Руденко Олег Григорьевич
доктор технических наук,
профессор

Цей примірник дисертації ідентичний
за змістом з іншими, що подані до
спеціалізованої вченої ради Д.64.05.01

Учений секретар О.А. Винокурова

Харьков-2017

СОДЕРЖАНИЕ

ВВЕДЕНИЕ 7
РАЗДЕЛ 1. АНАЛИЗ СОСТОЯНИЯ ПРОБЛЕМЫ И ПОСТАНОВКА ЗАДАЧ ИССЛЕДОВАНИЯ 18
1.1 Эвристическая оптимизация 18
1.2 Эволюционные алгоритмы: генетические алгоритмы, эволюционные стратегии, генетические программирование и роевой интеллект 19
1.2.1 Генетические алгоритмы (ГА) 21
1.2.2 Эволюционные стратегии (ЭС) 28
1.2.3 Генетическое программирование (ГП). 29
1.2.4 Программирование с экспрессией генов (ПЭГ) 30
1.3 Искусственные нейронные сети прямого распространения (ИНС) .. 32
1.3.1 Многослойный персептрон (МСП) 33
1.3.2 Радиально-базисная сеть (РБС) 36
1.3.3 Обобщенно-регрессионная сеть (ОРС) 40
1.3.4 Нейронная сеть CMAC 41
1.3.5 Нейронная сеть Wavenet 42
1.4 Синтез ИНС прямого распространения 48
1.4.1 Выбор структуры ИНС 48
1.4.2 Обучение ИНС прямого распространения 64
Выводы по разделу 1 и постановка задачи исследования 72
РАЗДЕЛ 2. ПРИНЦИПЫ ПОСТРОЕНИЯ И ФУНКЦИОНИРОВАНИЯ ЭВОЛЮЦИОНИРУЮЩИХ НЕЙРОННЫХ СЕТЕЙ ПРЯМОГО РАСПРОСТРАНЕНИЯ 78
2.1 Основные теории эволюции и их применение при проектировании ЭИНС. 78
2.1.1 Теория эволюции Дарвина 78
2.1.2 Эволюция Ламарка 81
2.1.3 Эффект Болдуина 83
2.2 Эволюционная оптимизация ИНС 85

2.3 Эволюционирующие ИНС 91
2.3.1 Инициализация популяции 92
2.3.2 Оценка приспособленности 103
2.3.3 Селекция 105
2.3.4 Скрещивание 110
2.3.5 Мутация 120
2.3.6 Операторы мутации и стратегии их применения 122
2.3.7 Критерии останова 122
Выводы по разделу 2 123
РАЗДЕЛ 3. МЕТОДЫ ОБУЧЕНИЯ ЭВОЛЮЦИОНИРУЮЩИХ НЕЙРОННЫХ СЕТЕЙ ПРЯМОГО РАСПРОСТРАНЕНИЯ 125
3.1 Одношаговые процедуры обучения 126
3.1.1 Одношаговые процедуры обучения при наличии информации о статистических свойствах помех 126
3.1.2 Одношаговые процедуры обучения при наличии ограниченных помех 139
3.2 Адаптация параметров процедур обучения 145
3.2.1 Адаптация параметра регуляризации 145
3.2.2 Адаптация зоны нечувствительности 151
3.3 Рекуррентные процедуры обучения, основанные на методе наименьших квадратов 153
3.3.1 Рекуррентная форма МНК 154
3.3.2 Рекуррентная форма МНК со скользящим окном 156
3.3.3 Рекуррентные формы проекционных процедур с $S < N$ 158
3.3.4 Модификации РМНК с зоной нечувствительности 161
3.4 Традиционное обучение, основанное на методах Гаусса-Ньютона и Левенберга-Марквардта 165
3.4.1 Обучение в режиме off-line 165
3.4.2 Рекуррентные формы процедур обучения: обучение в режиме on-line 167
3.4.3 Модификация процедур обучения Гаусса-Ньютона и Левенберга-Марквардта с зоной нечувствительности 171

3.5 Исследование скорости сходимости и надежности процедур обучения . 172
Выводы по разделу 3 174
РАЗДЕЛ 4. РОБАСТНОЕ ОБУЧЕНИЕ ИНС ПРЯМОГО РАСПРОСТРАНЕНИЯ 180
4.1. Некоторые подходы, используемые при робастном обучении 180
4.1.1 Робастный медианный метод наименьших квадратов 180
4.1.2 Укороченный метод наименьших квадратов 181
4.1.3 М-обучение 182
4.2 Выбор критерия робастного обучения 184
4.2.1 Оптимальные процедуры обучения 184
4.2.2 Традиционные критерии М-обучения 185
4.2.3 Выбор критерия обучения при несимметричном распределении 191
4.3 Процедуры робастного обучения 194
4.3.1 Оптимальные процедуры обучения 194
4.3.2 Одношаговые робастные процедуры обучения 195
4.3.3 Многошаговые робастные процедуры обучения 198
4.4 Модификации многошаговых процедур, содержащие зону нечувствительности 201
4.4.1 Модификация многошаговых процедур 201
4.4.2 Сходимость модифицированных многошаговых процедур 202
4.5 Робастные процедуры Гаусса-Ньютона и Левенберга-Марквардта 205
4.5.1 Рекуррентные процедуры Гаусса-Ньютона и Левенберга-Марквардта 205
4.5.2 Робастные процедуры Гаусса-Ньютона и Левенберга-Марквардта с зоной нечувствительности (с ограниченной точностью) 207
4.6 Оценивание параметров функционалов 210
4.6.1 Оценивание параметра масштаба в режиме off-line 210
4.6.2 Оценивание параметров помех в режиме on-line 212
4.6.3 Аппроксимация асимметрично распределенной помехи 213

4.7 Имитационное моделирование 215
Выводы по разделу 4 221
РАЗДЕЛ 5. МНОГОКРИТЕРИАЛЬНАЯ ОПТИМИЗАЦИЯ ЭВОЛЮЦИОНИРУЮЩИХ НЕЙРОСЕТЕВЫХ МОДЕЛЕЙ 226
5.1 Задача многокритериальной оптимизации по Парето 229
5.2 Парето-эволюционирующие ИНС прямого распространения 235
5.3 Коэволюционные ГА 239
5.3.1 Конкуренция 240
5.3.2 Аменсализм 241
5.3.3 Мутуализм 241
5.3.4 Комменсализм 242
5.3.5 Хищничество 242
5.4 Коэволюционирующие ИНС 244
5.4.1 Многокритериальный метод обучения ИНС на основе кооперативной коэволюции 247
5.4.2 Многокритериальный метод обучения ИНС на основе конкурентной коэволюции 253
5.4.3 Многокритериальный метод обучения ИНС на основе комбинированного коэволюционного подхода 256
5.4.4 Робастность коэволюционирующих систем 257
5.5 Особенности решения задачи многокритериального обучения при использовании РБС 258
5.6 Многокритериальное обучение ИНС с использованием алгоритмов кластеризации 260
5.7 Имитационное моделирование 264
5.7.1 Тестирование алгоритмов многокритериальной оптимизации 264
5.7.2 Моделирование работы коэволюционирующих ИНС 273
Выводы по разделу 5 282
РАЗДЕЛ 6. ЭКСПЕРЕМЕНТАЛЬНЫЕ ИССЛЕДОВАНИЯ И РЕШЕНИЕ ПРАКТИЧЕСКИХ ЗАДАЧ 285
6.1 Решение задачи идентификации 288

6.2 Нейросетевое управление многомерными нелинейными объектами 295
6.3 Задача нейросетевого прогнозирующего управления 302
6.4 Нейросетевое сжатие изображений на основе ЭИНС 313
6.5 Управление технологическими процессами в сахарной промышленности 321
6.6 Нейросетевое управление длиной петли в травильных ваннах 329
6.7 Использование нейросетевого подхода для контроля распределения электрической мощности в процессе плавки 337
6.8 Использование эволюционирующих нейронных сетей в самообучающихся интеллектуальных системах прогнозирования 344
Выводы по разделу 6 345
ОБЩИЕ ВЫВОДЫ 348
СПИСОК ИСПОЛЬЗОВАННОЙ ЛИТЕРАТУРЫ 352
ПРИЛОЖЕНИЯ 387
СОДЕРЖАНИЕ ПРИЛОЖЕНИЙ 388

ВВЕДЕНИЕ

Актуальность темы. Решение широкого круга задач различных отраслей науки, техники и экономики, таких как идентификация, фильтрация, восстановление, прогнозирование и т. д., связано с аппроксимацией некоторых нелинейных функций. Отсутствие информации о виде нелинейности зачастую приводит к неэффективности традиционных методов аппроксимации, а в ряде случаев – к их неприменимости. Альтернативой традиционным методам является применение нейросетевых технологий.

Являясь универсальными аппроксиматорами, некоторые типы искусственных нейронных сетей (ИНС) позволяют восстановить с заданной точностью любую сколь угодно сложную непрерывную нелинейную функцию. Наибольшее распространение при решении такой задачи получили статические ИНС прямого распространения (многослойный персептрон (МП) [1], радиально-базисные сети (РБС) [2], обобщенно-регрессионная сеть (ОРС). вейвлет-сети) [3]). Все эти сети используют представление нелинейного оператора некоторой системой базисных функций, реализуемой нейронами.

При синтезе или использовании ИНС возникают задачи структурной и параметрической оптимизации, соответствующие выбору оптимальной топологии сети и ее обучению (настройке параметров). Если задача определения структуры является дискретной оптимизационной (комбинаторной), то поиск оптимальных параметров осуществляется в непрерывном пространстве с помощью классических методов оптимизации.

Для обучения (оценивания параметров) сети применяются, как правило, методы, требующие вычисления градиента используемого функционала (алгоритм обратного распространения ошибки (ОР), метод сопряженных градиентов, алгоритм Гаусса-Ньютона, Левенберга-Марквардта и т.д.) Несмотря

на популярность этих методов не только при обучении ИНС, но и решении других задач оптимизации, они имеют ряд существенных недостатков.

Попытки устранить недостатки традиционных методов синтеза и функционирования ИНС привели к появлению нового класса сетей – эволюционирующих ИНС (ЭИНС), в которых в дополнение к традиционному обучению используется другая фундаментальная форма адаптации – эволюция, реализуемая путем применения эволюционных вычислений [4-10].

Использование в ЭИНС этих двух форм адаптации – эволюции и обучения, позволяющих изменять структуру сети, ее параметры и алгоритмы обучения без внешнего вмешательства, делает данные сети наиболее приспособленными для работы в нестационарных условиях и наличии неопределенности относительно свойств исследуемого объекта и условий его функционирования.

Основным преимуществом использования эволюционных алгоритмов (ЭА) в качестве алгоритмов обучения является то, что многие параметры ИНС могут быть закодированы в геноме и определяться параллельно. Более того, в отличие от большинства алгоритмов оптимизации, предназначенных для потактового решения задачи, ЭА оперируют с множеством решений – популяцией, что позволяет достичь глобального экстремума, не застревая в локальных. При этом информация о каждой особи популяции кодируется в хромосоме (генотипе), а получение решения (фенотипа) осуществляется после эволюции (отбора, скрещивания, мутации) путем декодирования.

Связь работы с научными программами, планами, темами. Диссертационная работа выполнена в рамках госбюджетных тем «Эволюционные гибридные системы вычислительного интеллекта со сменной структурой для интеллектуального анализа данных» (№ДР0110U000458), раздел «Эволюционные гибридные методы и модели интеллектуальной обработки информации со сменной структурой в условиях неопределенности», «Нейро-

фаззи системы для текущей кластеризации и классификации последовательностей данных в условиях их искаженности отсутствующими и аномальными наблюдениями» (№ДР0113U000361), раздел «Адаптивные методы и модели классификации данных и прогнозирования временных рядов в условиях их искаженности отсутствующими и аномальными наблюдениями на основе искусственных иммунных систем», «Разработка теоретических основ и математического обеспечения нейро-фаззи-систем ранней диагностики, прогнозирования и моделирования в условиях априорной и текущей неопределенности» (№ДР0101U001762), а также госдоговорной темы «Разработка и изготовление системы сбора и обработки информации о состоянии вращающихся трубчатых печей производства ферросплавов» (№ДР0104U009291), утвержденных Министерством образования и науки Украины, которые выполнялись в ХНУРЭ и в которых автор принимал участие как исполнитель.

Цель и задачи исследования. Целью диссертационной работы является развитие теоретических основ и разработки новых эволюционирующих ИНС для решения проблемы повышения качества интеллектуального анализа и обработки информации при наличии априорной и текущей неопределенности. Для достижения этой цели в работе решаются такие основные задачи:

1. Анализ существующих методов нейросетевой обработки информации в условиях априорной и текущей неопределенностей.

2. Разработка новых и усовершенствование существующих архитектур ИНС с использованием эволюционного и коэволюционного подходов, ориентированных на решение задачи интеллектуального анализа данных в условиях априорной и текущей неопределенностей.

3. Разработка методов автоматического определения и коррекции структуры и параметров ИНС в зависимости от изменения свойств исследуемого объекта.

4. Разработка методов упрощения структур и методов обучения ИНС прямого распространения с целью ускорения процессов обработки информации при допустимой неточности.

5. Разработка Парето-эволюционирующих ИНС прямого распространения.

6. Разработка методов обучения эволюционирующих ИНС, которые обладали бы повышенной скоростью обучения и робастностью, при наличии негауссовских помех, в частности, с ассиметричными распределениями.

7. Экспериментальные исследования свойств и характеристик различных методов, разработка рекомендаций по их применению, решение тестовых и практических задач с помощью разработанных ЭИНС.

Объект исследования – процессы построения интеллектуальных систем обработки информации.

Предмет исследования – эволюционирующие ИНС прямого распространения, предназначенные для интеллектуальной обработки данных в условиях нестационарности и априорной и текущей неопределенностей.

Методы исследования: основываются на теории вычислительного интеллекта, а именно на методах теории искусственных нейронных сетей, которая позволила синтезировать нейросетевые модели и нейрорегуляторы и получить процедуры их обучения; методы теории оптимальности, с помощью которой были синтезированы быстродействующие процедуры обучения; методы теории идентификации, на основе которых были синтезированы настраиваемые модели рассматриваемых объектов управления; методы имитационного моделирования, позволившие подтвердить эффективность полученных результатов и разработать рекомендации по их практическому использованию. Экспериментальные исследования проводились в лабораторных условиях и на реальных объектах.

Научная новизна результатов диссертационной работы заключается в следующем:

1. Впервые предложен метод робастной многокритериальной оптимизации (Парето-оптимизации) на основе робастных фитнес-функций и информационных критериев, дающих возможность определять оптимальную структуру нейросетевой модели исследуемого объекта при наличии негауссовских помех.

2. Впервые предложен обобщенный ЭА Парето-оптимизации на основе коэволюционного подхода, который позволяет изменять архитектуру сети, адаптируясь к изменяющейся внешней среде.

3. Впервые предложены простые в вычислительном отношении одно- и многошаговые рекуррентные процедуры обучения, обеспечивающие требуемую точность при наличии ограниченных помех, и разработаны процедуры адаптивной коррекции параметров.

4. Впервые предложен робастный метод обучения ИНС, дающий возможность обрабатывать информацию при наличии помех с негауссовскими, в частности асимметричными, распределениями и устранять смещение оценок параметров сетей, характерное для традиционных методов обучения.

5. Впервые предложены рекуррентные методы оценивания параметров используемых функционалов и помех для модели Тьюки-Хьюбера, что позволяет при отсутствии информации о статистических свойствах помех корректировать получаемые в процессе обучения параметры ИНС.

6. Впервые предложены законы адаптивного прогнозирующего нейроуправления нелинейными нестационарными объектами, функционирующими в условиях неопределенности, на основе эволюционного подхода с коррекцией эталонной траектории, что позволяет существенно ускорить процесс синтеза модели и вычисления управляющего сигнала.

7. Впервые предложены методы аппроксимации гауссовских базисных функций в РБС нулевого и первого порядков, позволяющие существенно

упростить вычисления, сопутствующие процессам построения модели исследуемого объекта.

8.Усовершенствована структура эволюционирующей ИНС, которая отличается от аналогов тем, что учитывает эволюцию модели помехи и процедуры обучения, что позволяет решать задачи обработки информации на новом качественном уровне по сравнению с существующими системами.

9. Получил дальнейшее развитие метод гибридного обучения ЭИНС путем использования для окончательной тонкой настройки параметров сети рекуррентной робастной процедуры Левенберга-Марквардта, что позволяет повысить качество получаемой модели и устойчивость процесса обучения.

10. Получили дальнейшее развитие эволюционные методы устранения влияния помех при определении структуры и параметров нейросетевых моделей путем использования оценок параметров модели помехи Тьюки-Хьюбера и процедуры М-обучения, что позволяет упростить структуру хромосомы, так как не требует хранения дополнительных параметров.

11. Получил дальнейшее развитие эволюционный метод многокритериальной оптимизации структуры и параметров ИНС путем выделения общих для эволюционного и иммунного подходов операторов и использование их для построения сети, что позволяет устранить большинство трудностей и недостатков классических методов решения задачи многокритериальной оптимизации.

Практическое значение результатов диссертационной работы состоит в том, что:

- разработаны программные средства, реализующие предложенные методы построения эволюционирующих ИНС прямого распространения, позволяющие автоматизировать процесс построения нейросетевых моделей исследуемых объектов, осуществить структурно-параметрический синтез в условиях априорной и текущей неопределенности;

- проведены экспериментальные исследования свойств и характеристики разработанных методов, которые подтвердили основные положения, выносимые на защиту, и показали, что предложенные методы за счет использования дополнительной информации о свойствах объекта и действующих помехах позволяют существенно сократить время построения моделей, а также обеспечить их устойчивость и робастность;

- синтезированные в диссертации структуры ИНС, их модели и процедуры обучения могут быть использованы при разработке систем интеллектуального анализа данных, систем интеллектуального управления объектами с непрерывными технологическими процессами;

- разработанное программное обеспечение, реализующее предложенные и исследованные в диссертации методы, внедрено в ООО «Побужзкий ферроникелевый комбинат» при разработке инфракрасной телевизионной системы измерения температурных полей вращающихся трубчатых печей (акт внедрения от 14.03.2015); в системах микроконтроллерного управления процессом травления полосовой стали на металлургических предприятиях Украины (АТ «Співдружність-Т»), микропроцессорного устройства управления расходом теплоносителя для различных технологических процессов (ООО «АО Содружество-Т») (акт внедрения от 21.10.2015); в АСУ ТП диффузии, дефекосатурации, выпаривания и кристаллизации в ООО «Кириковский сахарный завод» (акты внедрения от 24.12.2009 г.);

- научные положения, выводы и рекомендации диссертационной работы использованы в учебном процессе при подготовке курсов «Нейронные вычислительные структуры», «Интеллектуальные компьютерные системы» «Методы и средства вычислительного интеллекта», «Параллельные и распределенные вычисления», и «Иммунные вычислительные системы» на кафедре ЭВМ Харьковского национального университета радиоэлектроники (акт внедрения от 24.06.2016).

Личный вклад соискателя. Основные результаты получены лично автором. В работах, написанных с соавторами, соискателю принадлежит: в [33] – разработка алгоритма ПЭГ и схем кодирования синтаксических деревьев для их последующей линеаризации и хранения; в [34] – ускорение процесса получения модели и повышение ее качества с помощью алгоритма ПЭГ; в [36] – модификация эволюционного процесса, используемого в ПЭГ, улучшающая свойства традиционного алгоритма; в [56] – разработка робастного подхода к обучению вэйвлет-нейронных сетей нулевого и первого порядка на основе М-оценивания; в [60] – сравнительный анализ эффективности радиально-базисных сетей и вэйвлет-нейросетей; в [61] – разработка алгоритмов управления сложными технологическими процессами с помощью робастных вэйвлет-нейронных сетей; [76] – разработка эффективных алгоритмов обучения искусственных нейронных сетей; [92] – разработка робастных процедур обучения радиально-базисных сетей с использованием асимметричных функционалов; [97] – разработка устойчивых алгоритмов обучения радиально-базисных сетей; [141] – разработка модификации нейронных сетей, позволяющей использовать дополнительный «нормализующий» вход; в [142] – разработка нейроконтроллера на базе РБС для управления нелинейными динамическими объектами с использованием кусочно-линейной аппроксимации гауссовских базисных функций; в [143] – аппроксимации базисных функций; в [145] – разработка метода выбора структуры эволюционирующей РБС, ее адаптация и обучение на основе генетического алгоритма; в [155] – разработка методов устойчивого обучения радиально-базисных сетей при наличии помех измерений, имеющих распределения, отличные от нормального; в [156] – получение робастного метода обучения радиально-базисных сетей при наличии помех измерений, имеющих несимметричные распределения; в [157] – разработка метода оценивания параметров помех с помощью алгоритма стохастической аппроксимации; в [215] – получение алгоритма робастного обучения РБС,

обеспечивающего устойчивость процесса обучения при наличии помех; в [224] – определение структуры и обучение РБС с помощью генетического алгоритма; в [234] – разработка устойчивых процедур обучения ИНС при робастной идентификации нелинейных объектов; в [242] – разработка и исследование алгоритмов оценивания параметра масштаба при робастном обучении искусственных нейронных сетей в режимах on- и off-line; в [250] – получение и исследование устойчивых алгоритмов обучения РБС на основе взвешенного МНК, содержащих зону нечувствительности; в [252] – использование многокритериального подхода к обучению эволюционирующих нейронных сетей прямого распространения; в [267] – получение процедуры выбора оптимальной модели из фронта Парето с помощью робастных информационных критериев; в [268] – разработка метода борьбы с негауссовскими помехами с помощью робастных фитнес-функции; в [295] – разработка метода прогнозирующего управления нелинейными объектами с помощью эволюционирующих искусственных нейронных сетей прямого распространения; в [324] – анализ особенностей построения ЭИНС прямого распространения при решении задач прогнозирования экономических процессов, моделирование их работы.

Апробация результатов диссертации. Основные результаты диссертационной работы докладывались и обсуждались на Международной научно-технической конференции „Автоматизация: проблемы, идеи, решения” (Севастополь, 2009 г., 2010 г., 2011 г., 2012 г., 2013 г.); на 16-й, 17-ой международных конференциях по автоматическому управлению «Автоматика-2009», «Автоматика-2010» (Черновцы, 2009 г., Севастополь, 2010 г.); на Международной научно-технической конференции „Сучасні методи, інформаційне, програмне та технічне забезпечення систем управління організаційно-технологічними комплексами” (Киев, 2009 г.); на Международной научной конференции «Інтелектуальні системи прийняття рішень та проблеми обчислювального інтелекту» (Евпатория, 2009 г.); на 1-ой, 2-ой

Международных научно-технических конференциях «Сучасні напрями розвитку інформаційно-комунікаційних технологій та засобів управління» (Харьков-Киев, 2010 г., 2011 г.); на 1-ой, 2-ой Международных научно-технических конференциях „Інформаційні технології в навігації і управлінні: стан та перспективи розвитку", (Киев, 2010 г., 2011 г.); на 1-ой Международной научно-технической конференции «Обчислювальний інтелект (результати, проблеми, перспективи» (Черкассы, 2011 г.); на 2-ой, 3-ей, 5-ой Международных научно-практических конференциях (Смоленск, 2012 г., 2013 г., 2015 г.); на 3-ей, 4-ой, 5-ой, 6-ой Международных научно-технических конференциях «Современные направления развития информационно-коммуникационных технологий и средств управления» (Полтава-Белгород-Харьков-Киев-Кировоград, 2013 г., Полтава-Баку-Белгород- Кировоград-Харьков, 2014 г., Полтава-Баку-Кировоград-Харьков, 2015 г., 2016 г.); на IX Международной конференции «Стратегия качества в промышленности и образовании» (Варна (Болгария), 2013 г.); на 3-ей Международной научно-практической конференции «Стратегічні рішення інформаційного розвитку економіки, суспільства та бізнесу», (м. Рівне, 2014 г.); на 3-ей Международной научно-технической конференции «Проблеми інформатизації», (Черкаси-Баку-Бельсько-Бяла-Полтава, 2015 г.); на IV Международной научно-практической конференции «Общество и экономическая мысль в XXI в.: пути развития и инновации» (Воронеж, 2016 г.); на VIII Международной научно-практической конференции «Проблеми та перспективи розвитку IT-індустрії» (Харьков, 2016 г.).

Публикации. Основные положения и результаты диссертационной работы достаточно полно отображены в 56 опубликованных работах (2 коллективные монографии, 25 статей в изданиях, входящих в перечень ВАК Украины (6 единоличных), 28 – тезисы докладов на международных конференциях и форумах).

Структура и объем диссертации. Диссертация состоит из введения, шести разделов, заключения, списка использованных источников (324 наименования на 35 страницах) и 11 приложений на 50 страницах. Работа

содержит 126 рисуноков, из них 83 на 44 отдельных страницах, и 16 таблиц, из них 10 на 6 отдельных страницах. Общий объем работы составляет 436 страниц, в том числе 295 страниц основного текста.

РАЗДЕЛ 1
АНАЛИЗ СОСТОЯНИЯ ПРОБЛЕМЫ И ПОСТАНОВКА ЗАДАЧ ИССЛЕДОВАНИЯ

1.1 Эвристическая оптимизация

Многие практические и теоретические проблемы оптимизации характеризуются многомерностью пространства поиска. Эти проблемы включают в себя NP-полные задачи комбинаторной оптимизации, идентификации сложных структур или многомерной оптимизации функций.

В области планирования производства и логистики такие проблемы возникают особенно часто (например, при распределении задач, маршрутизации и т.п.). Применение традиционных методов исследования, таких как градиентные, динамическое программирование, симплекс-метод для этих видов задач часто не дает желаемого результата, так как вычислительные затраты растут экспоненциально вместе с размерностью задачи. В связи с этим довольно часто для решения практических задач применяются эвристические методы с гораздо более низкими вычислительными затратами, несмотря на то, что они могут и не обеспечивать достижения глобального оптимального решения.

Около трех десятилетий назад в литературе начали обсуждать заимствованные у природы эвристические методы, которые являются более гибкими и эффективными. К таким методам оптимизации относят метод имитации отжига, который проводит аналогию между отжигом материала до его самого низкого энергетического состояния и решением задачи оптимизации, эволюционные алгоритмы (ЭА), в основном заимствованных из биологической эволюции. Такие современные подходы, как табу поиск, муравьиные алгоритмы, метод роя частиц также упоминаются в контексте заимствованных у природы методов оптимизации. В последнее время получает все большее

значение в области эвристической оптимизации теория агентов [7].

Систематизация оптимизационных методов представлена на рис. 1.1. В данной работе основное внимание уделяется таким направлениям, как искусственные нейронные сети (ИНС) и эволюционные алгоритмы (ЭА), а также их комбинации в эволюционирующих искусственных нейронных сетях (ЭИНС).

Рисунок 1.1 – Систематизация оптимизационных методов

1.2 Эволюционные алгоритмы: генетические алгоритмы, эволюционные стратегии, генетические программирование и роевой интеллект

В настоящее время быстро развивается новое направление в теории и практике искусственного интеллекта – эволюционные вычисления (ЭВ). Этот термин обычно используется для общего описания алгоритмов поиска, оптимизации или обучения основанных на некоторых формализованных принципах естественного эволюционного отбора. Особенности идей эволюции и самоорганизации заключаются в том, что они находят подтверждение не только для биологических систем развивающихся много миллиардов лет. Эти идеи в настоящее время с успехом используются при разработке многих технических и, в особенности, программных систем [8].

История эволюционных вычислений началась в 60-е годы XX века, когда различные группы ученых в области кибернетики независимо друг от друга исследовали возможности применения принципов эволюции биологических систем при решении различных технических проблем, как правило, требующих решения задач оптимизации. Таким образом, было основано новое научное направление, которое в настоящее время принято называть "эволюционными вычислениями".

ЭВ используют следующие механизмы естественной эволюции:

1) Первый принцип основан на концепции выживания сильнейших и естественного отбора по Дарвину, который был сформулирован им в 1859 году в книге «Происхождение видов путем естественного отбора». Согласно Дарвину, особи, которые лучше способны решать задачи в своей среде, выживают и больше размножаются (репродуцируют). В генетических алгоритмах каждая особь представляет собой решение некоторой проблемы. По аналогии с этим принципом особи с лучшими значениями целевой (фитнесс) функции имеют большие шансы выжить и репродуцировать. Формализация этого принципа, как мы увидим далее, дает оператор репродукции.

2) Второй принцип обусловлен тем фактом, что хромосома потомка состоит из частей полученных из хромосом родителей. Этот принцип был открыт в 1865 году Менделем. Его формализация дает основу для оператора скрещивания (кроссинговера).

3) Третий принцип основан на концепции мутации, открытой в 1900 году де Вре. Первоначально этот термин использовался для описания существенных (резких) изменений свойств потомков и приобретение ими свойств, отсутствующих у родителей. По аналогии с этим принципом генетические алгоритмы используют подобный механизм для резкого изменения свойств потомков и тем самым, повышают разнообразие (изменчивость) особей в популяции (множестве решений).

Эти три принципа составляют ядро ЭВ. Используя их, популяция (множество решений данной проблемы) эволюционирует от поколения к поколению.

В области искусственного интеллекта, эволюционные алгоритмы (ЭА) являются подмножеством ЭВ – общего популяционного алгоритма оптимизации. В ЭА используются такие вдохновленные биологической эволюцией механизмы, как воспроизведение, мутация, рекомбинация и селекция. Возможные решения задачи оптимизации играют роль особей в популяции, и фитнес-функция (функция потерь) определяет качество каждого из решений. Эволюция популяции происходит вследствие многократного применения выше указанных эволюционных операторов.

ЭА являются универсальными аппроксиматорами решений всех типов задач, так как они не делают никаких предположений о ландшафте фитнес-функции решаемой задачи. Эта способность к обобщению обусловливает успех применения ЭА в таких разнообразных областях, как инжиниринг, искусство, биология, экономика, маркетинг, генетика, робототехника, социальные науки, физика, химия и политика. Основные преимущества и недостатки ЭА приведены в табл. 1.1.

1.2.1 Генетические алгоритмы (ГА).

Среди ЭА, являющихся стохастическими и включающих эволюционное программирование, эволюционные стратегии, генетические алгоритмы, генетическое программирование, в частности, программирование с экспрессией генов, роевой интеллект, одними из наиболее распространенных являются генетические алгоритмы (ГА) [11-12], которые были впервые предложены в Мичиганском университете американским исследователем Дж. Холландом для решения задач оптимизации в качестве достаточно эффективного механизма комбинаторного перебора вариантов решения. В отличие от многих других

работ, целью Холланда было не только решение конкретных задач, но исследование явления адаптации в биологических системах и применение его в вычислительных системах. При этом потенциальное решение – особь представляется хромосомой – двоичным кодом. Популяция содержит множество особей. В процессе эволюции используются три основных генетических оператора: репродукция, кроссинговер и мутация. Голдберг (ученик Холланда) успешно развил ГА и расширил области их применения. Его монография [13], в которой систематически изложены основные результаты и области практического применения ГА, является в настоящее время наиболее известной и цитируемой.

Основываясь на идеях Холланда была принята концепция стандартного генетического алгоритма (SGA), на который сильно повлиял его биологический архетип [14]. В связи со значительным увеличением вычислительной мощности с 1975 года, потенциал ГА увеличивается все больше и больше. В связи с чем популярность GA-концепций неуклонно растет, и многие ученые по всему миру стали решать различные проблемы науки и техники с помощью ГА. Однако вскоре стало очевидным, что для большинства практических задач двоичное кодирование, первоначально предложенное Холландом, вовсе не является достаточным. В связи с этим были разработаны различные варианты кодировки информации, а также новые операторы скрещивания и мутации, применение которых весьма разнообразило поведение проектируемых на основе ГА приложений. Обзор различных кодировок и операторов, разработанных для различных приложений может быть найден в [15]. С тех пор ГА были успешно использованы для решения широкого круга задач, включая многие комбинаторные задачи оптимизации, оптимизации многомерных функций, машинного обучения и эволюции сложных структур, таких как нейронные сети. Обзор ГА и приложений на их основе в различных сферах дается Голдбергом [13] и Михаливичем [16].

Таблица 1.1 – Преимущества и недостатки ЭА

Преимущества	**Недостатки**
Робастность	Методы ЭА не гарантируют нахождение оптимального решения за конечное время
Относительно просты в реализации	Экспертные знания предметной области должны быть явно добавлены с помощью внешних процессов
Хорошо подходят для многоцелевой оптимизации	Время оптимизации непостоянно, дисперсия между лучшим и худшим решениями может быть значительной
Хорошая масштабируемость за счет распараллеливания	В некоторых случаях высокая вычислительная сложность может являться проблемой
Очень гибкие (для широкого использования)	Должна быть точно определена фитнес-функция, в противном случае ЭА не работают
Хороший вариант решения задач, не имеющих лучшего традиционного способа решения	Во многих случаях ЭА медленнее, чем «жадные» алгоритмы
Доступно большое количество программных библиотек	Не лучший выбор, если традиционный метод уже решает проблему эффективным образом
Для использования ЭА необходимо небольшое количество специфических математических знаний	
Подходит для эффективного решения NP-полных задач	

Генетические алгоритмы (ГА) являются методом машинного обучения основанными на механизмах отбора в природе [17], которые ведут случайный и параллельный поиск решений, которые оптимизируют заранее определенную фитнес функцию [18].

В природе генетическая информация определяется четверичным кодом, основанном на четырех нуклеотидах: аденине, цитозине, гуанине и тимине, соединенных вместе в последовательность ДНК, которая лежит в основе генетического кода [19]. При переносе этой структуры в информатику, кажется естественным основывать все кодирование на двоичном коде, общепринятом в компьютерных науках. То есть используются хромосомы, которые являются двоичными строками, кодирующими решения поставленных задач, и именно это является отличительной чертой генетических алгоритмов [20]. ГА выполняет глобальный поиск в пространстве решений, которое состоит из векторов данных. Первым шагом алгоритма является инициализация пространства решений случайно сгенерированными значениями. На каждой итерации, доступные решения мутируют или скрещиваются с другими решениями для того, чтобы создавать новые решения. В конце каждой итерации, каждый индивидуум (решение-кандидат) оценивается с использованием заданной фитнес функции. Таким образом, фитнес-функция - ключевая часть каждого эволюционного алгоритма, и предназначена для поиска того решения, которое является наилучшим для данной задачи. После того, как каждая особь была оценена, наименее пригодные кандидаты отклоняются, при этом остаются только лучшие из доступных решений в популяции. Это и есть принцип Дарвина о выживании наиболее приспособленных особей при решении вычислительных задач. Цикл итераций повторяется до тех пор, пока предопределенный критерий останова не будет достигнут. В качестве критерия останова может использоваться либо число итераций, либо условие прохождения определенного порога значения фитнес-функции, которое должно

быть достигнуто в пространстве решений. Более подробная информации по ГА может быть найдена в [13], [21], [22].

Рассмотрим, прежде всего, классический генетический алгоритм, являющийся базовым для эволюционных методов оптимизации. ГА используют принципы и терминологию, заимствованные у биологической науки – генетики. В ГА каждая особь представляет потенциальное решение некоторой проблемы. В классическом ГА особь кодируется строкой двоичных символов – хромосомой, каждый бит которой называется геном. Множество особей – потенциальных решений составляет популяцию. Поиск (суб)оптимального решения проблемы выполняется в процессе эволюции популяции – последовательного преобразования одного конечного множества решений в другое с помощью генетических операторов репродукции, кроссинговера и мутации.

Эволюцию искусственной популяции – поиска множества решений некоторой проблемы формально можно описать алгоритмом, который представлен на рис.1.2.

Генетические алгоритмы – не просто случайный поиск, они эффективно используют информацию накопленную в процессе эволюции.

ГА берет множество параметров оптимизационной проблемы и кодирует их последовательностями конечной длины в некотором конечном алфавите (в простейшем случае двоичный алфавит «0» и «1»). ГА работает до тех пор, пока не будет выполнено заданное количество поколений (итераций) процесса эволюции или на некоторой генерации будет получено заданное качество или вследствие преждевременной сходимости при попадании в некоторый локальный оптимум.

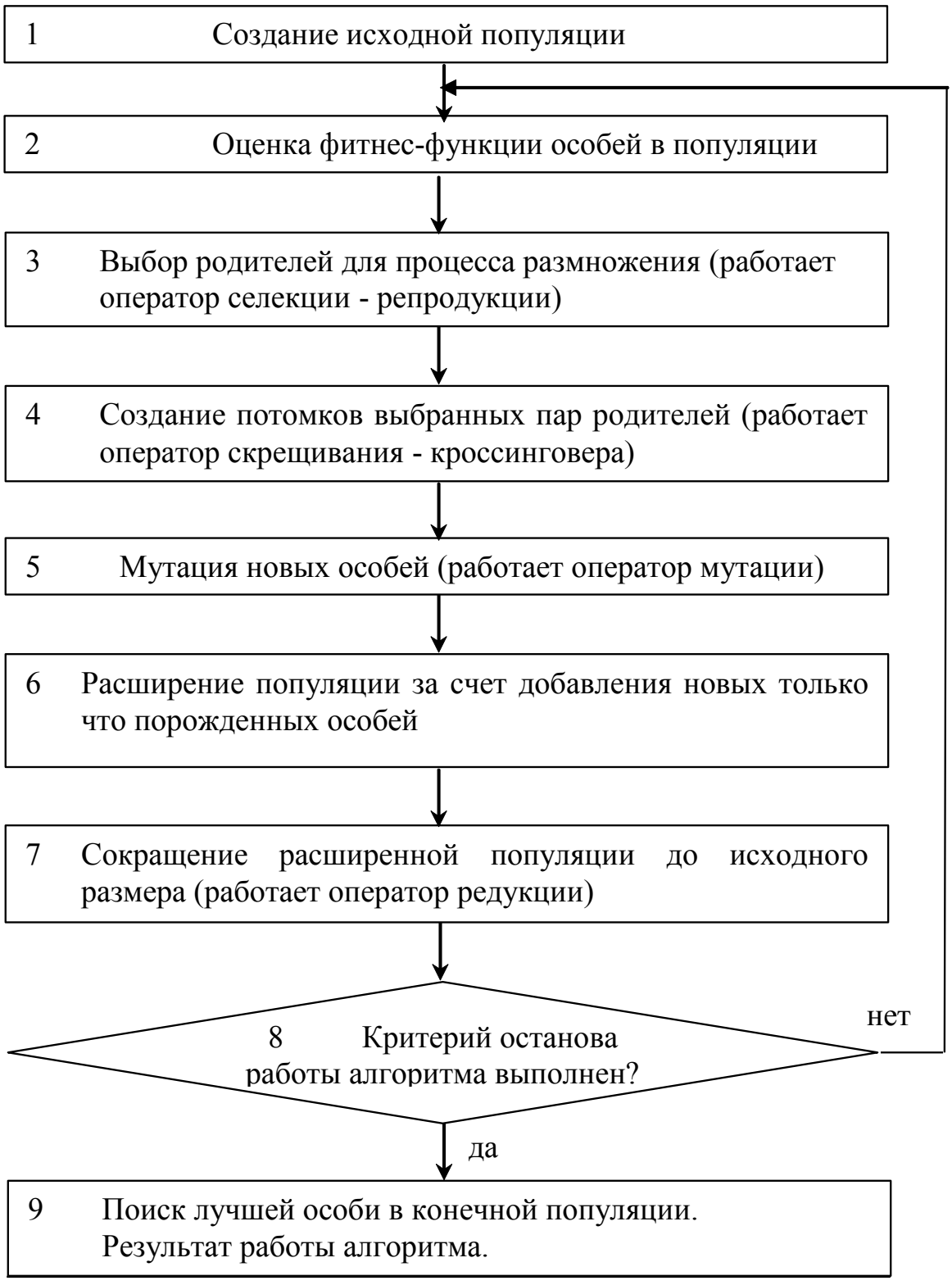


Рисунок 1.2 – Простой генетический алгоритм

В каждом поколении множество искусственных особей создается с использованием старых и добавлением новых с хорошими свойствами. Генетические алгоритмы – не просто случайный поиск, они эффективно используют информацию накопленную в процессе эволюции.

В процессе поиска решения необходимо соблюдать баланс между "эксплуатацией" полученных на текущий момент лучших решений и расширением пространства поиска. Различные методы поиска решают эту проблему по разному.

Например, градиентные методы практически основаны только на использовании лучших текущих решений, что повышает скорость сходимости с одной стороны, но порождает проблему локальных экстремумов с другой стороны. В полярном подходе случайные методы поиска используют все пространство поиска, но имеют низкую скорость сходимости. В ГА предпринята попытка объединить преимущества этих двух противоположных подходов. При этом операторы репродукции и кроссинговера делают поиск направленным. Широту поиска обеспечивает то, что процесс ведется на множестве решений – популяции и используется оператор мутации.

В отличие от других методов оптимизации ГА оптимизируют различные области пространства решений одновременно и более приспособлены к нахождению новых областей с лучшими значениями целевой функции за счет объединения квазиоптимальных решений из разных популяций.

Предварительно простой ГА случайным образом генерирует начальную популяцию стрингов (хромосом). Затем алгоритм генерирует следующее поколение (популяцию), с помощью трех основных генетических операторов:

1) Оператор репродукции (ОР);
2) Оператор скрещивания (кроссинговера, ОК);
3) Оператор мутации (ОМ).

Генетические операторы являются математической формализацией приведенных выше трех основополагающих принципов Дарвина, Менделя и де Вре естественной эволюции и выполняются следующим образом.

Каждый из функциональных блоков ГА рис.3.1 может быть реализован различными способами.

1.2.2 Эволюционные стратегии (ЭС).

Второй по величине представитель эволюционных алгоритмов - эволюционные стратегии (ЭС), был предложен в 70-х годах в Германии Рехенбергом [23] для решения задачи оптимизации вещественных параметров в расчете линий электропередачи. Это направление развивалось долгие годы независимо и здесь были получены важные фундаментальные результаты. В ЭС потенциальное решение – особь является вектором вещественных чисел, популяция состоит из двух особей и основным генетическим оператором является мутация. ЭС, как правило, достаточно эффективны при поиске локальных оптимумов. Хотя, в случае многомерных пространств решений, эволюционные стратегии, как правило, не способны обнаружить глобальный оптимум, если ни одно из начальных значений не находится в поглощающей области такого глобального оптимума. Тем не менее, ЭС считается одним из самых мощных и эффективных концепций для оптимизации векторов вещественных параметров.

ЭС являются стохастическим подходом к численной оптимизации, показывающим в целом хорошую производительность и подражающий принципам органической эволюции в области оптимизации параметров [24]. Чтобы улучшить параметр "самоадаптации" стратегии, Окура [25] предложил расширенный алгоритм ЭС, называемый Robust Evolution Strategy (RES), который имеет избыточные параметры нейтральной стратегии и который выборочно использует нейтральные мутации для улучшения адаптивности

параметров стратегии, подобный подход был предложен в [26] и более подробно исследован в [27-29].

1.2.3 Генетическое программирование (ГП).

Коза заложил основы генетического программирования (ГП) [30] как самостоятельного направления в области эволюционных вычислений в Массачусетском технологическом институте США. В качестве особи в ГП выступала программа на LISP, которая представлялась древовидной структурой. На этих структурах были разработаны генетические операторы кроссинговера и мутации.

Принимая во внимание основные соображения Коза [30] для интерпретации подзадач более общим и динамическим образом, чем в обычном ГА, основные механизмы отбора, рекомбинации и мутации в ГП были адаптированы и применены аналогичным ГА образом. Более общее представление проблем в ГП позволяет определить особь в популяции как структуру, формулу, или даже в более общем случае как программу. Это позволяет рассматривать новые области применения для ЭА. Следует, однако, отметить, что в настоящее время формирование более сложных программ с помощью ГП, выглядит весьма амбициозной задачей.

ГП отличается от ГА в основном в виде необходимых алгоритму входных данных и генерируемых им выходных значений [31]. В случае ГП, значениями являются не простые точки данных, но части функций или программ. Цель алгоритма состоит в том, чтобы найти процедуру, которая решает данную проблему наиболее эффективным образом. При этом вначале работы алгоритма случайным образом генерируется пространство вариантов решений (случайные программы в данном случае), которые развиваются с помощью процессов мутаций и скрещивания, генерирующих новые деревья программ. Конечным

результатом является функция или программа, которая может быть использована для решения конкретного типа задач.

1.2.4 Программирование с экспрессией генов (ПЭГ).

Как в целом и ГА и ГП, является генетическим алгоритмом с такими характерными чертами, как популяция особей, их отбор на основе приспособленности (фитнеса), изменчивость на основе генетических операторов [32-34]. Фундаментальное отличие этих трёх алгоритмов заключается в природе особей: в ГА это строки фиксированной длины (хромосомы); в ГП – нелинейные сущности переменной длины и формы (синтаксические деревья). В ПЭГ особи кодируются в виде строк фиксированной длины (называемых геномом, либо хромосомами [35-36]), которые затем декодируются для получения синтаксических деревьев.

Различие между ГА и ГП выражено не сильно: обе системы используют один вид объектов, который служит как генотипом, так и фенотипом. Такой подход имеет одно из двух ограничений: простота применения генетических операторов к объектам означает их недостаточную выразительность и сложность этих объектов (в случае ГА), а их сложность ведёт к трудностям при воспроизводстве и модификации (в случае ГП).

ПЭГ лишено указанных ограничений и обладает следующими преимуществами:

- Простота хромосом: линейность, компактность, относительно небольшой размер, простота применения операторов, таких как репликация, мутация, рекомбинация, перенос и пр.
- Экспрессия (декодирование компактной структуры) генома порождает фенотип – синтаксическое дерево, которое может представлять математическую формулу либо программу. Значения вычисленной формулы, выполненной программы служат для численной оценки приспособленности

особи с помощью функции фитнеса.

Таким образом, участие хромосомы в процессе воспроизведения зависит от фитнеса дерева, которое она кодирует. Для этого требуется универсальная система перевода формата генома в формат дерева и обратно, при которой любое изменение хромосомы с помощью операторов всегда приводит к синтаксически корректному дереву, обеспечивая приспосабливаемость и эволюционирование.

На рисунке 1.3 показана блок-схема алгоритма ПЭГ [32]. Процесс начинается с конструирования популяции случайным образом созданных хромосом. После экспрессии (декодирования) хромосом каждая особь выполняется на заданном наборе входных данных (в виде координат точек для математических задач, таблиц истинности при поиске булевых выражений, наборов признаков при решении задач классификации и т.д.). На основании результатов выполнения особей вычисляется фитнес каждой из них, позволяющей отобрать и подвергнуть изменениям особи, нацеленным на получение потомства – популяции с новыми характеристиками. С этой популяцией проводятся те же операции: декодирование генома, отбор, воспроизведение с модификациями. Процесс повторяется заданное количество итераций либо до получения решения.

Воспроизведение включает в себя не только репликацию, но также выполнение генетических операторов, вносящих разнообразие. Во время репликации геном особи в точности, без изменений копируется и переносится в новое поколение. Очевидно, этот оператор не способен вносить разнообразие, потому требуется вмешательство оставшихся операторов, случайным образом выбирающих особи, к которым они будут применены. Таким образом, в ПЭГ особь может быть модифицирована сразу несколькими операторами, а может оказаться и не измененной вовсе [33].

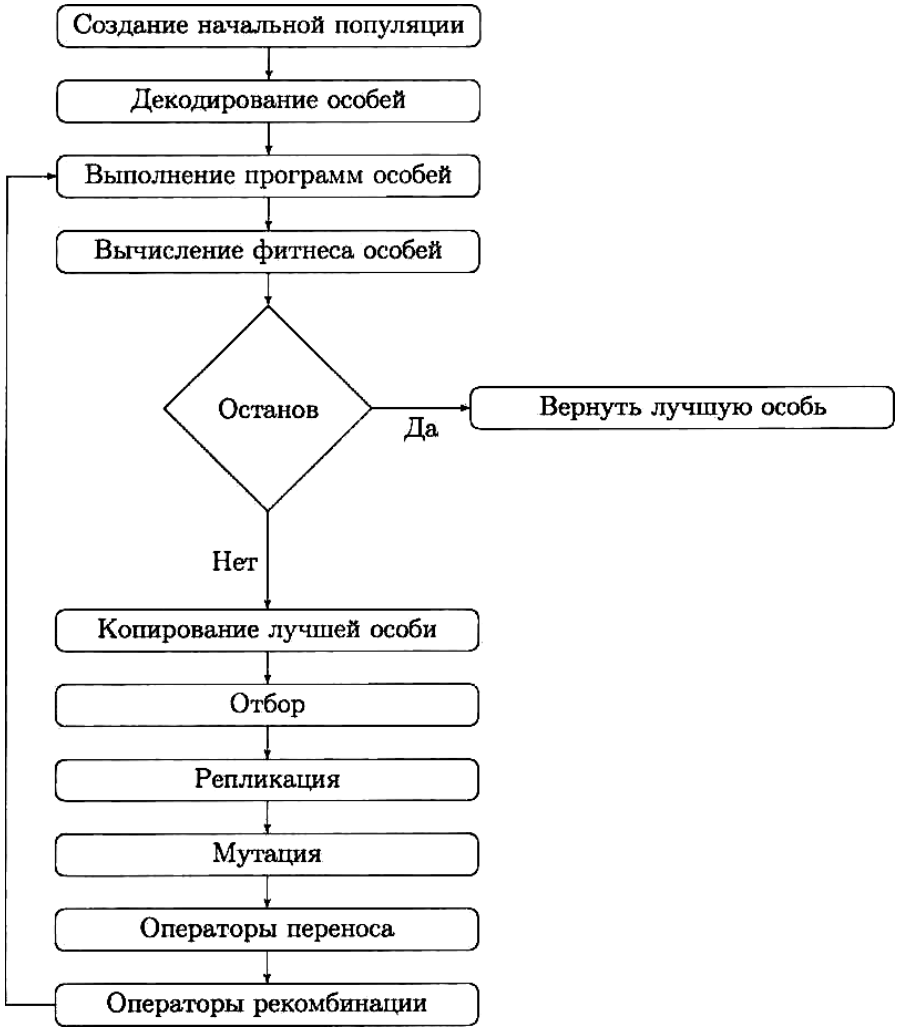


Рисунок 1.3 – Блок-схема исходного алгоритма ПЭГ

1.3 Искусственные нейронные сети прямого распространения (ИНС)

Многие задачи обработки информации либо сводятся к задаче аппроксимации некоторой, в общем случае нелинейной функции (обработка сложных сигналов, идентификация, прогнозирование временных последовательностей)

$$y(x) = f(\boldsymbol{x}) + \xi, \tag{1.1}$$

где $\boldsymbol{x}$ - вектор $M \times 1$; $f(x)$ - неизвестная нелинейная функция; ξ - помеха, либо используют получаемые при этом результаты для решения более сложной задачи (управление нелинейными объектами, классификация, распознавание образов, обработка изображений и т.д.)

Возможность аппроксимации со сколь угодно малой ошибкой любой непрерывной функции $f(x)$ искусственной нейронной сетью ИНС [37] обусловила достаточно широкое распространение нейросетевого подхода для решения данной задачи. При этом аппроксимируемая функция представляется некоторой сетью, образованной нейронами, параметры которых определяются путем обучения сети на основании предъявления обучающих пар $\{\boldsymbol{x}(k), y(k)\}, k = 1,2,...$

Среди существующего в настоящее время большого количества сетевых структур для этих целей обычно используют статические ИНС прямого распространения МСП, РБФ, сеть *CMAC,* нейронная сеть WaveNet. Существуют также и другие нейросетевые подходы, например, LOLIMOT – LOgical-LIinear-MOdel-Tree, GMDHNN – Group Method of Data Handling Neural Network, основанные на предложенном А.Г. Ивахненко методе группового учета аргументов (МГУА), и т.д.

Так как наибольшее распространение для аппроксимации сложных нелинейных функций и идентификации нелинейных объектов получили МСП, РБС, CMAC и WaveNet, рассмотрим вкратце каждую из них.

1.3.1 Многослойный персептрон (МСП).

МСП представляет собой нейронную сеть с несколькими слоями, каждый из которых состоит из вычислительных узлов (нейронов). Топология многослойного персептрона показана на рис. 1.4. Входы сети подсоединены к каждому нейрону в первом слое. Выходы нейронов первого слоя затем

становятся входами для нейронов второго слоя и так далее. Последний слой является выходным, все другие слои между входным и выходным слоями называются скрытыми слоями. Архитектура многослойного персептрона может быть удобно записана как $n_0 - n_1 - \ldots - n_l$, где n_0 является размерностью входного вектора сети, а n_i, $1 \le i \le l$ обозначает номера узлов в соответствующих слоях.

Таким образом, МСП использует следующую аппроксимацию нелинейного оператора $f(\bullet)$:

$$\hat{y}(k) = \hat{f}(k) = f^q\left[\left(W^q\right)^T f^{q-1}\left[\left(W^{q-1}\right)^T f^{q-2}\left[\ldots f^1\left[\left(W^1 x(k) + b_1\right)^T\right]\ldots\right]\right]\right] + b_q, \quad (1.2)$$

где W^i – вектор весовых параметров нейронов *i*-го слоя сети; $f^i[\bullet]$ – активационная функция (АФ) *i*-го слоя, b_i – смещение *i*-го нейрона.

Так как в практических приложениях приходится производить различные операции не только с самой активационной функцией, но и с ее первой производной, необходимым является использование в качестве активационной монотонной, дифференцируемой и ограниченной функции. Особо важную роль играют такие функции при моделировании нелинейных зависимостей между входными и выходными переменными. Это так называемые логистические или сигмоидальные (*S*-образные) функции.

К числу таких функций относятся

- логистическая (униполярная)

$$f_{\log}(z) = \frac{1}{1 + e^{-\alpha z}}; \qquad \frac{d}{dZ} f_{\log}(z) = \alpha f_{\log}(z)\left(1 - f_{\log}(z)\right), \quad (1.3)$$

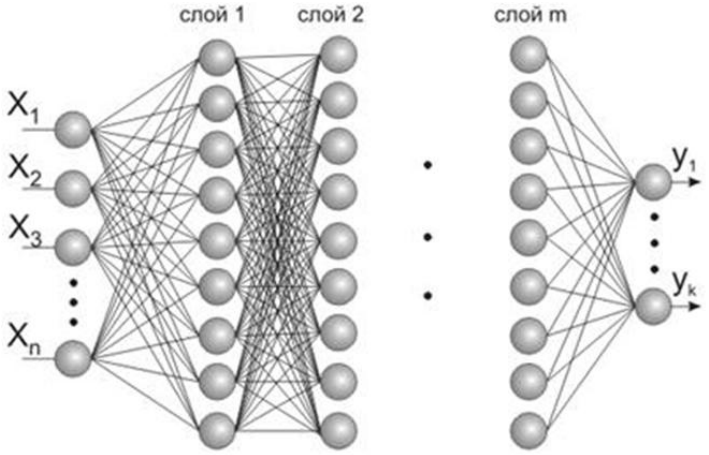


Рисунок 1.4 – Топология многослойного персептрона

- гиперболического тангенса (биполярная)

$$f_{th}(z) = \tanh(\alpha z) = \frac{e^{\alpha z} - e^{-\alpha z}}{e^{\alpha z} + e^{-\alpha z}};$$
$$\frac{d}{dZ} f_{th}(z) = 1 - \tanh^2(\alpha z). \qquad (1.4)$$

Функции (1.3) и (1.4) могут быть выражены друг через друга.

Следует также отметить, что преимущество функции $f_{th}(z)$ перед $f_{\log}(z)$ состоит в ее симметричности относительно начала координат (в ряде случаев это существенно облегчает вычисления).

В общем виде представление входа-выхода можно выразить как $\hat{f} : R^{n_l} \to R^m$.

Аппроксимирующие возможности МСП были исследованы и описаны многими авторами [38-42]. Основная идея состоит в том, что любая непрерывная функция $f : D_f \subset R^{n_l} \to R^m$ может быть равномерно аппроксимирована с произвольной точностью функцией $\hat{f}$ по D_f, где D_f это компактное подмножество R^{n_l}, при условии, что существует достаточное

количество скрытых слоев в сети. Это утверждение справедливо даже для сетей только с одним скрытым слоем. Типичным допущением для функций активации в скрытом слое является следующее: $f(\cdot)$ является непрерывной, ограниченной и неконстантой. Как видно, эти требования очень мягкие, а сигмоидальная функция (1.3) является только одним из множества возможных вариантов активационных функций. Данный теоретический результат предоставляет логическое обоснование для моделирование нелинейных систем с использованием МСП.

Эти математические результаты доказывают, что МСП является в общем виде аппроксиматором функций и гарантирует, что сети с одним скрытым слоем всегда будет достаточно для представления любой произвольной непрерывной функции. Но в данном утверждении ничего не говорится о количестве нейронов в скрытом слое, которые обеспечивают заданную точность аппроксимации. Важным также является то, что доказательство возможностей аппроксимации с помощью МСП предполагает, что веса заданы корректно. Однако остается открытым вопрос выбора алгоритма обучения.

1.3.2 Радиально-базисная сеть (РБС).

РБС является альтернативной МСП [43-45]. РБС имеет двухслойную вычислительную структуру. Скрытый слой состоит из матрицы узлов, каждый из которых содержит вектор параметров, называемый центром. Узел вычисляет некоторое расстояние между центром и входным вектором сети и передает результат через нелинейную функцию. Выходной слой является по существу набором линейных сумматоров. Архитектура РБС представлена на рис.1.5.

РБС использует следующую аппроксимацию $f(\bullet)$:

$$\hat{y}(k) = \hat{f}(k) = w_0 + \sum_{i=1}^{N} w_i \Phi_i(x) = w_0 + W^T \Phi(k), \qquad (1.5)$$

где $\Phi(k)$ – вектор выбранных базисных функций (БФ).

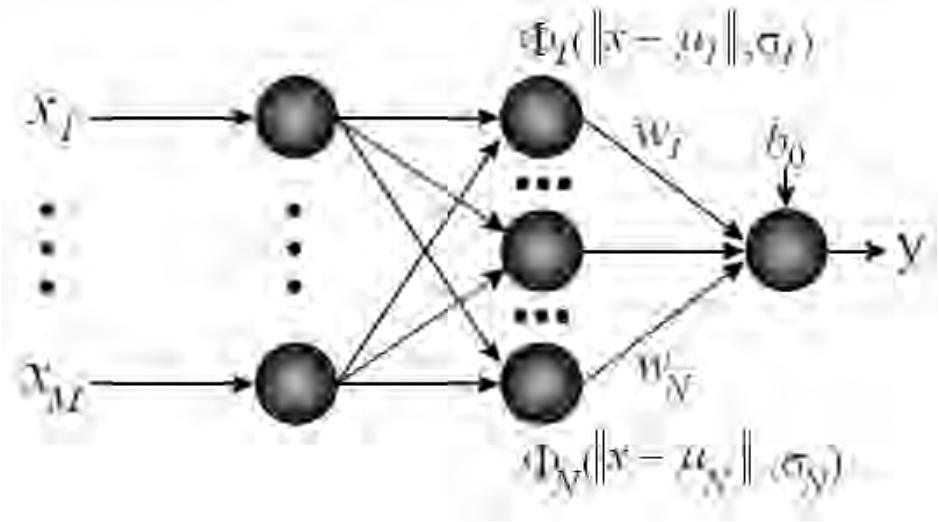


Рисунок 1.5 – Схема радиально-базисной сети

В данных сетях в качестве базисных выбираются некоторые функции расстояния между векторами

$$\phi_i(u) = f(\|u - c_i\|).$$

Векторы c_i называют центрами базисных функций. Функции $\phi_i(u)$ выбираются неотрицательными и возрастающими при увеличении $\|u - c_i\|$. В качестве меры близости векторов u и c_i берется обычно либо евклидова метрика

$\|u - c_i\| = \left(\sum_{i=1}^{N} (u - c_i)^2 \right)^{\frac{1}{2}}$, либо манхэттенская

$$|u - c_i| = \sum_{j=1}^{N} |u_j - c_{ij}|, \text{ где } |u_j - c_{ij}| = (u_j - c_{ij})\,\mathrm{sign}(u_j - c_{ij}),$$

$$\mathrm{sign}(u_j - c_{ij}) = \begin{cases} 1, \text{ если } (u_j - c_{ij}) > 0; \\ 0, \text{ если } (u_j - c_{ij}) = 0; \\ -1, \text{ если } (u_j - c_{ij}) < 0. \end{cases}$$

В качестве функций $\phi(\cdot)$ наиболее часто выбирают гауссову функцию

$$\phi(z, \sigma) = \exp(-z^2/\sigma^2), \tag{1.6}$$

здесь σ^2 - параметр отклонения (дисперсия).

Кроме того, используется также плоская сплайн-функция, мультиквадратичная и обратная мультиквадратичная функция.

Топология РБС, очевидно, является похожей на архитектуру многослойного персептрона, а различие заключается только в характеристиках (параметрах) скрытых нейронов.

В отличие от персептрона РБС характеризуют три типа параметров:

- линейные весовые параметры выходного слоя w_{ij} (входят в описание сети линейно);

- центры c_j - нелинейные (входят в описание нелинейно) параметры скрытого слоя;

- отклонения (радиусы базисных функций) σ_{ij} - нелинейные параметры скрытого слоя.

Существующее сходство между РБС и двухслойным персептроном свидетельствует о том, что данная сеть имеет аналогичные с персептроном возможности аппроксимации. Это, фактически, и было доказано. Таким образом, при очень мягких допущениях на нелинейность $\phi(\cdot)$, любая

непрерывная функция $f : D_f \subset R^{n_l} \to R^m$ может быть равномерно аппроксимирована с любой точностью при помощи РБС сети $\hat{f}$ по D_f при условии, что существует достаточное количество скрытых нейронов.

В доказательствах универсальной возможности аппроксимации РБС часто предполагается, что $\phi(\cdot)$ является непрерывной и ограниченной функцией, однако при этом выбор нелинейности $\phi(\cdot)$ не является решающим при использовании данной сети.

Несмотря на то, что параметр σ в каждом нейроне может иметь различное значение, одинаковое значение σ для каждого нейрона является достаточным условием для универсальной аппроксимации. Это означает, что все σ_j могут иметь фиксированные значения σ для применения более простой обучающей методики. На практике значение σ может некоторым образом отражаться на вычислительных свойствах алгоритмов обучения, однако это не сказывается на аппроксимирующих свойствах РБС. Наконец, как и в случае МСП, несмотря на то, что теория гарантирует возможность точного представления произвольной непрерывной функции РБС с корректными весами w_{ji} и центрами c_j, остается открытым вопрос о том, могут или нет реально эти параметры быть обучены с использованием какого-либо алгоритма обучения.

Следует также отметить, что при наличии большого числа неизвестных параметров (весов w_{ij}, центров c_j, дисперсий σ) использование большой обучающей выборки приводит к тому, что при идентификации сложных нелинейных объектов резко возрастает сложность вычислений, и применение данной сети для решения задачи идентификации в реальном времени становится проблематичным.

1.3.3 Обобщенно-регрессионная сеть (ОРС).

Данная сеть, предложенная Шпехтом Д. [46] для построения обобщенных (линейных и нелинейных) регрессий, является дальнейшим развитием РБС. Как и РБС, данная сеть использует некоторые базисные функции, например (1.6), которые однако являются нормированными. Так базисная функция i-го нейрона, основывающаяся на гауссовской (1.6), для данной сети принимает вид

$$\varphi_i(n) = \frac{\exp\left[-\frac{1}{2}\left(\frac{(u_1 - c_{1,i})^2}{\sigma_1^2} + \ldots + \frac{(u_N - c_{N,i})^2}{\sigma_N^2}\right)\right]}{\sum_{j=1}^{n_h} \exp\left[-\frac{1}{2}\left(\frac{(u_1 - c_{1,j})^2}{\sigma_1^2} + \ldots + \frac{(u_N - c_{N,j})^2}{\sigma_N^2}\right)\right]}. \quad (1.7)$$

При этом

$$\sum_{i=1}^{n_h} \varphi_i(n) = 1. \quad (1.8)$$

Использование нормированных базисных функций (1.7) привело к тому, что данную сеть называют еще нормализованной РБС [47, 48]. Структура данной сети приведена на рис.1.6.

Использование в данной сети функций, обладающих свойством (1.8), приводит к ряду положительных эффектов, позволяющих в отличие от радиально-базисной сети достичь большей точности аппроксимации.

Как отмечается в [49], данная сеть является очень эффективной при аппроксимации нелинейных функций одной и двух переменных. С увеличением же числа переменных вычислительные затраты, как и РБС, существенно возрастают и эффективность сети ОРС падает. Это приводит к тому, что

использовать данную сеть для идентификации сложных нелинейных объектов становится невозможным. Если же известно, что исследуемый объект является стационарным или квазистационарным, то целесообразно применение сети ОРС (как и РБС) для решения задачи идентификации.

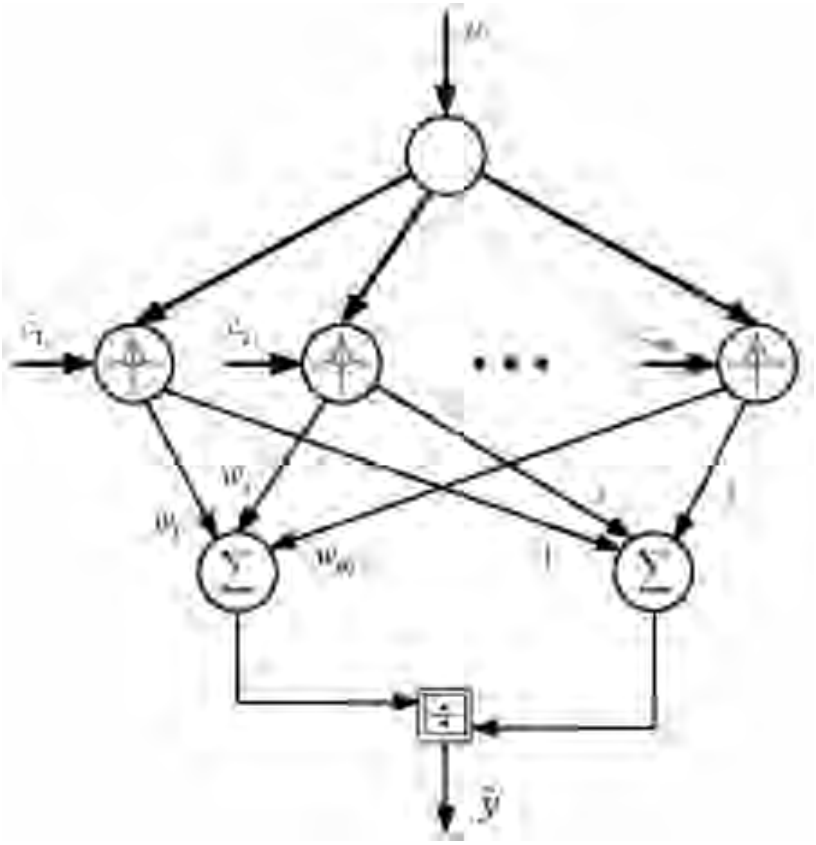


Рисунок 1.6 – Структура сети ОРС

1.3.4 Нейронная сеть CMAC.

В 1975г. *Дж. Альбусом* была предложена модель, описывающая процессы управления движением, происходящие в мозжечке, которая впоследствии была реализована в нейросетевом регуляторе для управления роботом-манипулятором, названом им *CMAC – Cerebellar Model Articulation Controller* (церебральная модель артикуляционного контроллера) [50, 51] . Впоследствии модель явилась основой для разработки нейросетевой ассоциативной памяти,

называемой также CMAC, однако в этом случае данная аббревиатура часто расшифровывается как Cerebellar Model Arithmetic Computer (церебральная модель арифметического компьютера). Простота реализации и хорошие аппроксимирующие свойства сети обеспечили ее достаточно широкое применение не только при управлении в реальном времени роботом-манипулятором, но и при решении задач распознавания образов, кодирования изображений, цифровой обработки и фильтрации сигналов, аппроксимации, восстановления и интерполяции одно- и многомерных функций.

1.3.5 Нейронная сеть Wavenet.

Новые возможности повышения эффективности решения задачи аппроксимации возникли с появлением вейвлет-нейронных сетей (ВНС), которые также осуществляют аппроксимацию вида (1.2), используя при этом в качестве БФ вейвлеты. Вейвлеты представляют собой локализованные функции, конструируемые из материнского вейвлета $\psi(x)$ путем операции сдвига по времени t_i и изменения временного масштаба d_i

$$\psi_i(x) = \psi\left(\frac{x - t_i}{d_i}\right).$$

Благодаря уникальным свойствам вевлетов (ортогональности, нормируемости, локальности и полноты вейвлет-базиса) вейвлет-преобразования стали мощным инструментом анализа сложных нелинейных зависимостей, резко изменяющихся функций и т.д. А возможность легкого обобщения вейвлет-преобразования на множества любых размерностей делает их весьма эффективными при анализе многомерных объектов [52].

Использование ВНС как специализированных ИНС прямого распространения для решения различных задач, начатое, по-видимому, с работы

[53] и развитое в [54-58], в настоящее время получило весьма широкое распространение.

Предложенная и наиболее исследованная в [55] вейвлет-нейросеть (WaveNet) представляет собой частный случай МП и обучается, как и МП, с помощью алгоритма обратного распространения ошибки, минимизирующего квадратичный функционал от ошибки аппроксимации $e(k) = y(k) - \hat{f}(k)$ на основе предъявления обучающих пар $(x(k), y(k)), k = 1,2,\ldots$

Сеть WaveNet образуется нейронами, в которых БФ являются вейвлетами. Так как выбор материнского вейвлета определяет свойства вейвлет-преобразования, в качестве материнских вейвлетов часто используются производные гауссовской функции, которая имеет наилучшую локализацию в частотной и временной областях. При повышении порядка производной область определения вейвлета несколько увеличивается, при этом вследствие подавления низкочастотных компонент появляется возможность анализировать высокочастотные структуры сигналов. Однако одним из наиболее широко используемых является вейвлет «мексиканская шляпа» (MHAT), вычисляемый по второй производной гауссовской функции и имеющий вид

$$\psi(x) = \left(1 - x^2\right) e^{-\frac{x^2}{2}}. \qquad (1.9)$$

Обобщенная функция MHAT, являющаяся M-мерной БФ *i*-го нейрона имеет вид

$$\psi(\boldsymbol{x}) = \Phi_i(\boldsymbol{x}, \boldsymbol{t}, R) = \left(1 - (\boldsymbol{x} - \boldsymbol{t})^T R^{-1} (\boldsymbol{x} - \boldsymbol{t})\right) e^{-(\boldsymbol{x}-\boldsymbol{t})^T R^{-1} (\boldsymbol{x}-\boldsymbol{t})}, \qquad (1.10)$$

где $R^{-1} = \left[r_{ij}^{k} \right], i, j = \overline{1, M}, k = \overline{1, N}$ (M – размерность входного сигнала, N – количество нейронов) – масштабирующая матрица (использование матрицы R^{-1} позволяет изменить не только ориентацию вейвлета, но и в ряде случаев, как видно из рис. 1.7, его форму).

Аппроксимация нелинейности $f(\bullet)$ ВНС $\Phi_i(\boldsymbol{x}, \boldsymbol{t}, R)$ приводит к вейвлет-нейросетевым моделям нулевого

$$\hat{f}(k) = \sum_{i=1}^{N} c_i \Phi_i(\boldsymbol{x}, \boldsymbol{t}, R) = \boldsymbol{c}^T \Phi(\boldsymbol{x}) \tag{1.11}$$

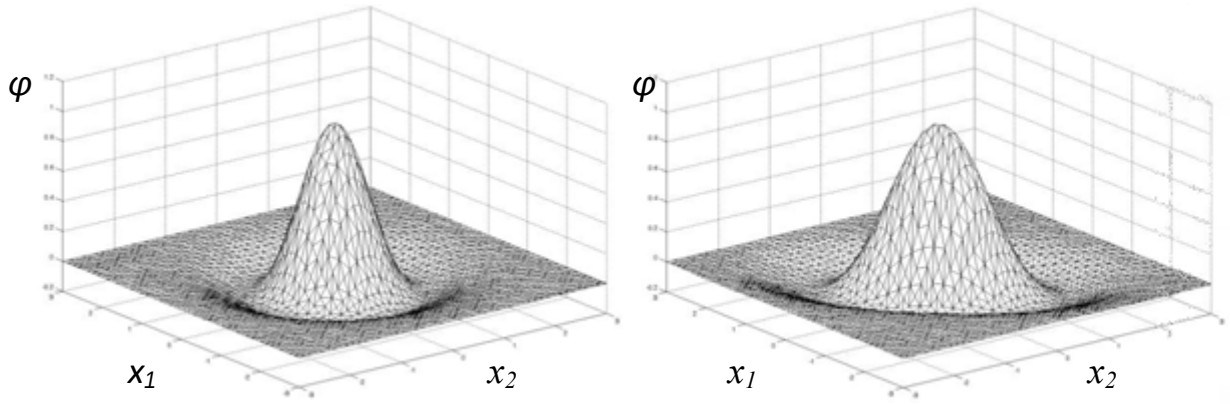
и первого

$$\hat{f}(k) = \left(\boldsymbol{c} + V\boldsymbol{x} \right)^T \Phi(\boldsymbol{x}) \tag{1.12}$$

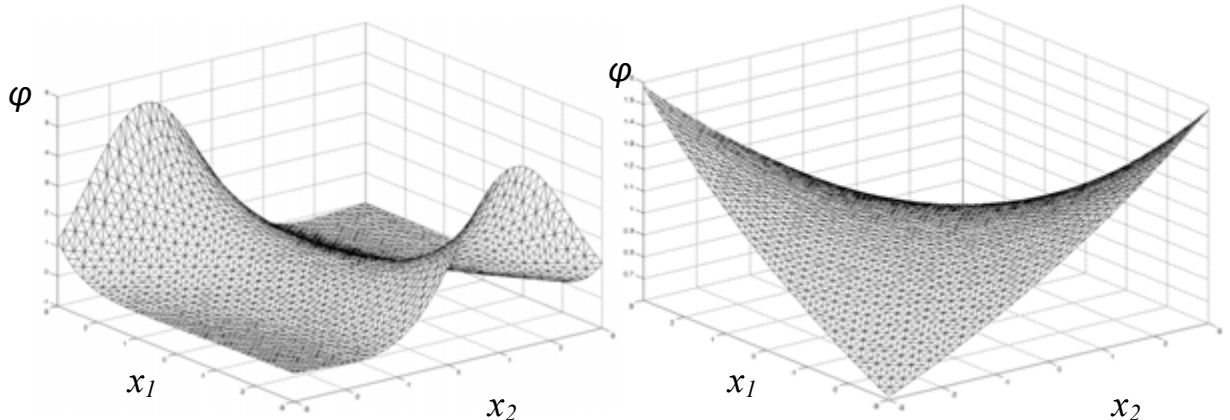
порядков.

Здесь $\boldsymbol{c}$ и V - подлежащие определению $N \times 1$ вектор и $N \times M$ матрица весовых коэффициентов соответственно.

Искусственная нейронная сеть Wavenet [55, 59-61], топология которой представлена на рисунке 1.8, состоит из скрытого слоя L1 параллельно соединенных нейронов – структурных элементов сети [62]. Каждый нейрон условно разбит на блоки, в которых происходит вычисление активационной функции от входных данных. В частности, блок D_i – параметр дилатации (масштабирования) активационной функции нейрона, а смещение T_i – параметр трансляции (сдвига) этой функции. После вычисления активационной функции от входных данных полученный результат умножается на весовой коэффициент w_i. Значения выходов всех нейронов суммируются в узле S для получения



а) $R^{-1} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$ б) $R^{-1} = \begin{bmatrix} 1 & 0.5 \\ 0.5 & 1 \end{bmatrix}$



в) $R^{-1} = \begin{bmatrix} 0.3 & 0.13 \\ 0.13 & -0.05 \end{bmatrix}$ г) $R^{-1} = \begin{bmatrix} -0.0003 & 0.013 \\ 0.013 & -0.0005 \end{bmatrix}$

Рисунок 1.7 – Изменение формы вэйвлета в зависимости от R^{-1}

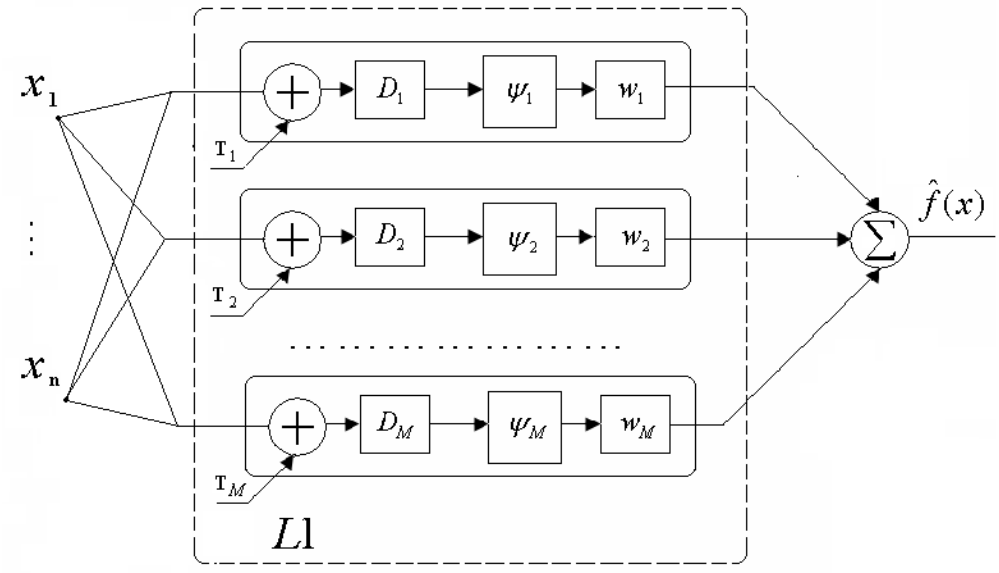


Рисунок 1.8 – Топология нейронной сети WaveNet

выхода сети y.

Нейронная сеть Wavenet является однослойной сетью прямого распространения без обратных и латеральных связей между нейронами [63]. При настройке сети изменению поддаются три параметра, а именно:

D_i – растяжение активационной функции;

T_i – сдвиг активационной функции;

w_i – весовые коэффициенты.

При выборе активационной функции для нейронной сети следует учитывать такие факты, как наличие аналитического выражения и дифференцируемость активационной функции, а также возможность построения на ее основе вейвлет-функции многих переменных.

Рассмотрим активационные функции, которые можно использовать при построении нейронной сети Wavenet:

– биортогональные вейвлеты с компактным носителем любой размерности могут быть сгенерированы с помощью лифтинг-схемы вейвлет-преобразования. Такие вейвлеты непрактичны при слишком большой размерности, поскольку их построение приводит к увеличению памяти, требуемой для хранения активационной функции. Кроме того, биортогональные вейвлеты, построенные по лифтинг-схеме не обладают аналитическим выражением, что не дает возможности применения к сетям, построенным на их основе, методов градиентной оптимизации. Достоинством биортогональных вейвлетов является то, что активационные функции состоят только из положительных целочисленных значений. Это позволяет интерпретировать данные функции в качестве лингвистических переменных при добавлении в реализацию Wavenet аппарата нечеткой логики;

– радиально-базисные функции, на основе которых Мишели [64] были разработаны полуортогональные вейвлеты и доказана возможность расширения

этих вейвлетов до любой размерности, а также предложены скейлинг-функции $\varphi(x)$, которые иначе называются полигармоническими $b-$ сплайнами:

$$\varphi(x)=\begin{cases}\|x\|^{2r-d}*\log\|x\|, & d-\textit{нечетное}\ ,\\ \|x\|^{2r-d}, & d-\textit{четное}\ .\end{cases}$$

Отсчеты функции $\psi(x)$, которая, как и биортогональные вейвлеты, не имеет аналитического выражения, вычисляются путем Фурье-преобразования функции $\varphi(x)$. На практике применение данной функции в нейронной сети ведет к неоправданным временным затратам и увеличению объема требуемой памяти, что ограничивает перспективу их использования;

– вейвлет Занга, предложенный в работах [55, 59, 65]. Если сеть имеет *n* входов, то данный вейвлет имеет следующее аналитическое выражение:

$$\psi(x)=(x^T x-d))*e^{-\frac{x^T x}{2}}\ . \qquad (1.13)$$

По своей структуре и количеству настраиваемых параметров нейронная сеть Wavenet схожа с нейронной сетью на основе радиально базисных функций [65] (в этой сети при обучении изменяются параметры z,σ,w). Закон функционирования нейронной сети Wavenet описывается формулой:

$$\hat{f}(x)=\sum_{i=1}^{N} w_i\psi[D_i(x-t_i)]\ , \qquad (1.14)$$

где $\hat{f}(x)$ – выход нейронной сети Wavenet .

Отметим, что кроме классической структуры ИНС Wavenet, описанной в [66], существует несколько популярных вариантов данной нейронной сети:

– диадрическая нейронная сеть Wavenet (сеть Бакши) была предложена Бакши в 1994 г. и является упрощенной моделью Wavenet, а именно радиально-базисной сетью, использующей вейвлеты в качестве активационных функций, в которой при обучении настраиваются только весовые коэффициенты:

$$f(x) = \sum_{n,m} d_{m,n} * \psi_{m,n}(x) + \bar{f} \,, \tag{1.15}$$

где $\bar{f}$ – среднее значение $f(x)$;

$d_{m,n}$ – коэффициенты нейронной сети;

– нейронная сеть Fuzzy-Wavenet, объединяющая теорию вейвлет-преобразования, нечеткую логику и аппарат нейронных сетей. Данная сеть отличается особенно эффективными процедурами обучения online благодаря применению нечеткой логики в своей структуре. Сеть адаптируется к каждому новому примеру обучающей выборки путем адаптации весовых коэффициентов только одного нейрона. За выбор этого нейрона отвечает набор логических правил, входящих в структуру сети, которые строятся по алгоритму Такаги-Сугено. Достоинством Fuzzy-Wavenet является полностью автоматический процесс обучения, делающий эту сеть полезной для практического применения.

1.4 Синтез ИНС прямого распространения

1.4.1 Выбор структуры ИНС

При нейросетевом подходе построение математической модели основывается на концепции «черного ящика», когда исследователь фиксирует реакцию сети и вызывающие её входные факторы, а функция, связывающая

факторы и реакцию сети, неизвестна. В связи с этим выбор структуры модели, под которым понимается определение необходимого количества регрессоров, является одним из важнейших этапов построения модели. Однако этот этап разработан и формализован недостаточно, субъективные суждения играют при его проведении существенную роль.

Задачу выбора структуры можно сформулировать как оптимизационную [67]:

$$s^* = \arg\min_{s \in \Omega} CR(s), \tag{1.16}$$

где s^* - оптимальная структура модели; s - структура модели; Ω – множество всех возможных структур, содержащих некоторые регрессоры из «полного» набора регрессоров; $CR(s)$ - критерий качества структуры.

Модель со структурой, доставляющей критерию качества минимальное значение, называют моделью оптимальной сложности [67], т.е. моделью, содержащей минимальное количество регрессоров.

Заметим, что не все подходы к выбору структуры могут быть сформулированы в виде задачи (1.16). Например, если при добавлении регрессоров в модель значение критерия качества монотонно уменьшается, то задача выбора структуры может быть модифицирована следующим образом: выбрать наиболее простую структуру s, критерий качества для которой удовлетворяет условию $CR(s) \le k$, где k – пороговое значение [67]. Сочетание проверки гипотез и оптимизации - особенность подхода, реализованного в методе пошаговой регрессии, где используются критерии качества, основанные на *F*-статистике.

В этом случае качество получаемой структуры модели оценивается с помощью параметрического статистического F -теста проверяющего гипотезу о

том, что дисперсии двух случайных величин X и Y, представленных выборками X_s и Y_s, совпадают. Для корректной работы требуется выполнение двух условий: обе случайные величины имеют нормальное распределение; выборки независимы. Если эти условия выполняются, а исследуемые модели являются линейными, то применение F-теста даёт хорошие результаты. В частности, данным тестом можно воспользоваться, если в РБС настраиваются только весовые коэффициенты, а центры и дисперсии задаются. В этом случае нейросетевая модель будет линейной относительно настраиваемых весов.

В целом же критерии, разработанные для выбора структуры, удобно разделить на две группы: критерии, использующие экзаменационную выборку, и критерии, не использующие экзаменационную выборку.

Для критериев первой группы используется принцип, согласно которому параметры модели необходимо оценить по одной части выборки (обучающей), а качество структуры - по другой части выборки (экзаменационной или проверочной). К этим критериям относятся: критерии регулярности, стабильности, непротиворечивости и вариативности [66, 67].

Для линейных регрессионных моделей данные критерии получаются путем разбиения исходной выборки $W:\{Y,X\}$ на части $A:\{Y_A,X_A\}$ и $B:\{Y_B,X_B\}$. Введем следующие обозначения: $\hat{\theta}_W$, $\hat{\theta}_A$ и $\hat{\theta}_B$ - МНК-оценки вектора параметров θ, вычисленные с использованием выборок *W, A* и *B* соответственно. Тогда критерии можно представить в следующем виде:

- симметричный и несимметричный критерии регулярности

$$\mathrm{Re}\, g = (Y_A^* - X_A\hat{\theta}_B)^T(Y_A^* - X_A\hat{\theta}_B) + (Y_B^* - X_B\hat{\theta}_A)^T(Y_B^* - X_B\hat{\theta}_A), \quad (1.17)$$

$$\mathrm{Re}\, g(B) = (Y_B - X_B\hat{\theta}_A)^T(Y_B - X_B\hat{\theta}_A); \quad (1.18)$$

- симметричный и несимметричный критерии стабильности

$$Stab = (Y - X\hat{\theta}_A)^T (Y - X_B\hat{\theta}_A) + (Y - X\hat{\theta}_B)^T (Y - X\hat{\theta}_B), \quad (1.19)$$

$$Stab(A) = (Y - X\hat{\theta}_A)^T (Y - X\hat{\theta}_A); \quad (1.20)$$

- симметричный и несимметричный критерии непротиворечивости (минимума смещения решений)

$$NC = (\hat{\theta}_A - \hat{\theta}_B)^T X^T X(\hat{\theta}_A - \hat{\theta}_B), \quad (1.21)$$

$$NC(B) = (\hat{\theta}_A - \hat{\theta}_B)^T X^T X_B(\hat{\theta}_A - \hat{\theta}_B); \quad (1.22)$$

- симметричный и несимметричный критерии вариативности (абсолютно помехоустойчивые критерии)

$$V = (\hat{\theta}_W - \hat{\theta}_A)^T X^T X(\hat{\theta}_B - \hat{\theta}_W), \quad (1.23)$$

$$V(B) = (\hat{\theta}_W - \hat{\theta}_A)^T X_B^T X_B(\hat{\theta}_B - \hat{\theta}_W). \quad (1.24)$$

Заметим, что любой симметричный критерий равен сумме двух несимметричных: $CR = CR(A)+CR(B)$.

К критериям первой группы относятся и критерии типа скользящего контроля (cross-validation) [68-75].

При вычислении критериев, относящихся ко второй группе используется остаточная сумма квадратов ошибок модели

$$RSS = \frac{1}{K}\sum_{i=1}^{K}\left[y(i) - \hat{y}(i)\right]^2 , \qquad (1.25)$$

где $y(i)$, $\hat{y}(i)$ - измерение и оценки соответственно.

Сама эта величина не может служить критерием для выбора структуры, так как с увеличением сложности модели S происходит все более точное приближение, что возможно и допустимо только при отсутствии помех. Если известно, что шум имеет нормальное распределение. То применяют скорректированный *RSS*

$$\frac{RSS}{K - S} \qquad (1.26)$$

или же статистика Фишера

$$F(S) = \frac{K}{K - S}\frac{RSS(S)}{\|y - \hat{y}\|^2} . \qquad (1.27)$$

Если при этом возможно получение оценки дисперсии помехи, то применяют критерий Мэллоуса:

$$C_S = \frac{1}{\hat{\sigma}^2} RSS + 2S - K . \qquad (1.28)$$

Величина *RSS* используются также в предложенном Акаике [75] критерии финальной ошибки прогнозирования

$$FPE(S) = \frac{K+S}{K-S} RSS(S) \quad . \tag{1.29}$$

В последнее время всё более широкое распространение получают информационные критерии качества оценивания структуры модели. Среди таких критериев следует отметить [68-76]:

Критерий Акаике (AIC – Akaike's information criterior)

$$AIC = K \ln\left[\frac{1}{K}\sum_{i=1}^{K} e^2(k)\right] + K \ln\frac{K+S}{K-S} \quad ; \tag{1.30}$$

Критерий Шварца-Риссанена (BIC – Bayesian information criterior)

$$BIC = K \ln\left[\frac{1}{K}\sum_{i=1}^{K} e^2(k)\right] + S \ln K \quad ; \tag{1.31}$$

Критерий Кульбака (KIC – Bayesian information criterior)

$$KIC = K \ln\left[\frac{1}{K}\sum_{i=1}^{K} e^2(k)\right] + 3S \ln K \quad ; \tag{1.32}$$

Критерий Хэннана-Куинна (HQ)

$$HQ = K \ln\left[\frac{1}{K}\sum_{i=1}^{K} e^2(k)\right] + 2S\ln(\ln K) \quad . \tag{1.33}$$

Критерий Хубера, получаемый минимизацией критерия

$$Q = \sum_{i=1}^{n} \rho\left(y[i] - \sum_{j=1}^{\rho} x_j[i]\beta_i\right) = \sum_{i=1}^{n} \rho(\varepsilon[i]), \tag{1.34}$$

где $\varepsilon[i]$ – ошибка i–го наблюдения, а ρ имеет вид

$$\rho(\varepsilon) = \begin{cases} \frac{1}{2}\varepsilon^2, & |\varepsilon| \le c \\ c|\varepsilon| - \frac{1}{2}c^2, & |\varepsilon| > c \end{cases} . \tag{1.35}$$

При этом полагаем, что c = kS, где k обычно равно 1.5 или 2.1, а S – оценка параметра масштаба.

Критерий Эндрюса. При этом наблюдениям, ошибки которых велики по абсолютной величине, присваиваются нулевые веса:

$$\psi(\varepsilon) = \begin{cases} 2.1S(1 - \cos\left(\frac{\varepsilon}{2}1S\right)), & |\varepsilon| \le 2.1S\pi \\ 4.2S, & |\varepsilon| > 2.1S\pi \end{cases} . \tag{1.36}$$

Критерий Мэллоуза является модификацией хуберовский оценок, в которых наряду со взвешиванием остатков ε производится взвешивание факторов, что позволяет уменьшить влияние точек, резко выделяющихся в

пространстве независимых переменных. Для нахождения оценок Мэллоуза необходимо решить $\rho+1$ уравнений

$$\psi(\varepsilon)=\begin{cases}\sum_{i=1}^{n}\psi(\varepsilon[i])\\ \sum_{i=1}^{n}x_m[i]\psi^x(x[i])\psi(\varepsilon[i]),\ m=0,1,..,\rho\end{cases}, \tag{1.37}$$

где ψ определяется по Хуберу, а

$$\psi^x(x[i])=\prod_{j=1}^{\rho}\psi^{(j)}(x[i]). \tag{1.38}$$

В оценках Форсайта минимизирующий критерий имеет вид

$$Q=\sum_{i=1}^{n}\rho(\varepsilon[i])=\sum_{i=1}^{n}|\varepsilon[i]|^{\alpha}, \tag{1.39}$$

где $1\le\alpha\le 2$. Эти оценки близки к хуберовским, однако, не имеют столь убедительного теоретического обоснования. Эмпирически показано, что приемлемым является α = 1.5. При α = 2 оценки Форсайта совпадают с оценками МНК, а при α = 1 получаем метод наименьших абсолютных отклонений (модулей), при котором критерий идентификации имеет вид

$$Q=\sum_{i=1}^{n}|\varepsilon[i]|. \tag{1.40}$$

Для получения оценок Меррила-Швеппа используется

$$\rho(\varepsilon)=\begin{cases} \gamma_1\varepsilon^2, |\varepsilon| \le k \\ \gamma_2\sqrt{\varepsilon}, |\varepsilon| > k \end{cases}, \quad (1.41)$$

где $\gamma_1>0$, $\gamma_2>0$, $0 \le k$.

Следует отметить, что в частном случае нормального распределения критерий AIC принимает вид

$$AIC(S) = RSS(S) + 2S, \quad (1.42)$$

называемом критерием Акаике-Мэллоуса.

Использование какого-либо критерия выбора структуры модели не гарантирует определение оптимальной структуры. Поиск наилучшей по какому-либо критерию качества структуры модели является задачей целочисленного нелинейного программирования.

Гарантированное решение задачи можно получить в результате перебора всех структур. Подобный алгоритм называется алгоритмом всех возможных регрессий [68,70], комбинаторным алгоритмом [71]. Количество возможных структур составляет 2^m, где m - количество регрессоров в «полном» наборе регрессоров, из которых выбирается оптимальный набор. По этой причине реально данным алгоритмом можно воспользоваться при относительно небольших значениях m.

Другой путь состоит в просмотре только наиболее перспективных вариантов. Этот принцип реализуют алгоритмы направленного перебора [60, 69]. Они осуществляют поиск минимума критерия с существенно меньшими затратами, однако в общем случае не обеспечивают нахождение оптимальной структуры.

Алгоритм *исключения* [72] начинается с оценки качества модели, содержащей все регрессоры, и состоит из шагов удаления из нее регрессоров до тех пор, пока величина критерия не перестанет уменьшаться. Алгоритм *включения* [72] действует в обратном направлении. Начиная с модели, содержащей аддитивную постоянную (регрессор «1»), он последовательно добавляет в нее регрессоры.

При использовании критериев качества, описанных выше, регрессор, включаемый (исключаемый) на каждом шаге, выбирается из условия наибольшего уменьшения значения критерия. Поиск оптимальной структуры заканчивается, если значение критерия невозможно более уменьшить путем включения или исключения регрессора.

Весьма важными являются вопросы устойчивости в задаче выбора структуры. В данном случае можно выделить три аспекта устойчивости.

Первый аспект связан с устойчивостью результатов моделирования к варьированию выборки [69, 73]. Если строятся две модели с одной структурой по двум различным частям выборки и результаты моделирования получаются различными, то выбранная структура не обеспечивает устойчивость результатов и, в целом, приводит к противоречивой модели. Одним из способов обеспечения устойчивости в данном смысле является применение критериев, использующих экзаменационную выборку.

Второй аспект связан с помехоустойчивостью критериев качества, т.е. с их способностью выбирать структуры, обеспечивающие удовлетворительные в каком-либо смысле результаты моделирования, в условиях увеличивающейся дисперсии ошибок.

Третий аспект устойчивости связан с устойчивостью процедур выбора модели к отклонению от классических предположений о свойствах ошибок наблюдений. Традиционно процедуры выбора структуры опираются на МНК-оценки и сами критерии строятся на основе остаточных сумм квадратов. При

таком подходе не учитываются, например, негауссовость, гетероскедастичность, зависимость ошибок наблюдений.

При синтезе или использовании ИНС возникают задачи структурной и параметрической оптимизации, соответствующие выбору оптимальной топологии сети и ее обучению (настройке параметров). Если задача определения структуры является дискретной оптимизационной (комбинаторной), то поиск оптимальных параметров осуществляется в непрерывном пространстве с помощью классических методов оптимизации.

Традиционные методы определения структуры сети заключаются либо в последовательном ее усложнении путем ввода новых нейронов и новых связей между ними, либо в последовательном ее упрощении, начиная с некоторой достаточно сложной топологии. Наиболее известный метод выбора топологии сети с одновременным определением весов – каскадная корреляция, использующий обучение с учителем и основанный на максимизации значения корреляции между выходами вновь вводимого узла (нейрона) и ошибкой ИНС, был предложен С.Фалманом [77, 78]. Структура каскадно-корреляционной сети, начальная топология которой включает только один нейрон, путем автоматического обучения становится многослойной. При этом число скрытых нейронов, определяющих сложность сети, пошагово увеличивается, уменьшая ошибку обучения. Основной недостаток данного метода – сложность распараллеливания вычислений вследствие наличия связей между всеми скрытыми узлами устраняется путем различных модификаций данного метода (использование упрощенной каскадной корреляции, случайного выбора слоя и т.д.) [79].

Существуют некоторые рекомендации по определению необходимого количества нейронов для МСП и РБС. Количество нейронов скрытого слоя МП предварительно можно оценить следующим образом [80]:

$$\frac{N}{10} - n - m \le n_k \le \frac{N}{2} - n - m, \tag{1.43}$$

где n_k – количество нейронов в скрытом слое;

n – количество входных сигналов;

m – количество выходных сигналов;

N – количество элементов статической выборки, необходимой для обучения.

С другой стороны, значение n_k можно оценить следующим образом:

$$\frac{N \cdot \varepsilon_o}{n} > n_k, \tag{1.44}$$

где ε_o – относительная ошибка программирования нейросети.

С учетом (1.44) можно записать

$$\begin{cases} \dfrac{N}{10} - n - m \le \dfrac{N \cdot \varepsilon_o}{\text{n}}; \\ \dfrac{N \cdot \varepsilon_o}{\text{n}} \le \dfrac{N}{2} - n - m. \end{cases} \tag{1.45}$$

Решая (1.45) относительно N, получаем

$$\frac{2n(n+m)}{(n-10\varepsilon_o)} \le N \le \frac{10n(n+m)}{(n-10\varepsilon_o)}. \tag{1.46}$$

При $\varepsilon_o \to 0$ неравенство (1.46) принимает вид

$$2n(n+m) \le N \le 10n(n+m). \qquad (1.47)$$

Количество синаптических весов для персептрона с единым скрытым слоем вычисляются так

$$K_w = n \cdot n_k \cdot m. \qquad (1.48)$$

Наиболее простым способом изменения структуры сети РБС является ее постепенное усложнение путем добавления новых нейронов, проводимое каждый раз, когда при предъявлении очередного *i*-го входного сигнала сети возникает ошибка аппроксимации $e(i) = f(i) - \hat{f}(i)$, превышающая допустимую. На первом шаге структура сети включает один нейрон с базисной (активационной) функцией (1.6). Затем подается обучающая последовательность и в зависимости от реакции сети, характеризуемую ошибкой $\boldsymbol{e}$, происходит либо ее обучение, либо добавление нового нейрона. В этом случае, если в l – й момент времени сеть содержала n нейронов, а предъявление вектора $\boldsymbol{x}(l)$ привело к появлению ошибки $e(l) > e_{доп} = \alpha$, то в сеть вводится новый, $(n+1)$-й, нейрон, величина сдвига базисной функции μ_{n+1} которого принимается равным $\mu_{n+1} = \boldsymbol{x}(l)$, вес – $c_{n+1} = e(l)$, а начальные элементы масштабирующей таблицы $R_{n+1}^{-1} = \|\boldsymbol{x}(l) - \mu_m(l)\|^{-1} I$, где $\mu_m(l)$ – величина сдвига базисной функции ближайшего к $\boldsymbol{x}(l)$ m-го нейрона для l – го входного сигнала ВНС, I - единичная матрица $M \times M$. Таким образом, условием введения нового нейрона является выполнение неравенств [81-83]

$$e(l) > \alpha, \qquad (1.49)$$

$$\|\boldsymbol{x}(l) - \mu_m(l)\| > \beta, \qquad (1.50)$$

где α и β – априорно устанавливаемые предельно допустимые значения ошибки реакции сети и отклонения обобщенного сигнала $\boldsymbol{x}(l)$ от ближайшего к данному входу центра.

Отметим, что задача определения количества нейронов в сети N, т.е. структуры сети, является важной и весьма сложной. Для ее решения могут использоваться, например, кросс-валидация, статистическая проверка гипотез, а также получающие в последнее время широкое распространение информационные критерии (IC), укладывающиеся в схему

$$IC = -2\ln\left(L\left(\hat{\theta}_N \middle| \boldsymbol{x}\right)\right) + F(N)K, \tag{1.51}$$

где $L(\bullet)$ – функция правдоподобия; $\hat{\theta}_N$ – оценка максимального правдоподобия вектора параметров θ; K С длина обучающей выборки, и отличающиеся видом функции $F(N)$. Так $F(N) = 2$ соответствует критерию Акаике (AIC – Akaike information criterion), $F(N) = 3$ – критерию Кульбака (KIC – Kullback information criterion), $F(N) = \ln N$ – критерию Шварца-Риссанена (BIC – Bayesian information criterion), $F(N) = 2\ln(\ln N)$ – критерию Хеннана-Куинна (HQ или LILC – “Law of iterated information criterion ”).

В последнее время в литературе по статистическому выбору модели значительное внимание уделяется принципу минимальной длины описания (МДО, Minimum Description Length, MDL), являющемуся весьма эффективным подходом при выборе модели и решении других проблем статистической обработки информации, а также обобщающему методы максимального правдоподобия и информационные критерии. Ключевым компонентом в принципе МДО является стохастическая сложность, введенная и использованная в работах [84-87], оценивающая насколько хорошо вероятностная модель соответствует статистическим закономерностям данных.

Из этих работ следует, что стохастическая сложность наблюдаемых данных по отношению к данному классу параметрических моделей может быть выражена следующим образом:

$$SC(\boldsymbol{\theta}) = -\log \rho(\boldsymbol{\theta}) + \frac{1}{2}\log|I(\boldsymbol{\theta})| + \sum_{i=1}^{N}\log\left(\left|\hat{\theta}_i\right| + K^{-1/4}\right), \qquad (1.52)$$

где $I(\boldsymbol{\theta}) = -\dfrac{\partial^2 \log \rho}{\partial\boldsymbol{\theta}\partial\boldsymbol{\theta}^T}$ – информационная матрица NxN; $\hat{\boldsymbol{\theta}}$ – М-оценка вектора $\boldsymbol{\theta}$; K – количество математических моделей.

Таким образом, штрафуя излишнюю сложность модели, МДО обосновывает идею максимизации регуляризованного правдоподобия, т.е. позволяет обосновать корректность регуляризации правдоподобия.

Область применения МДО шире, чем у статистических методов обучения, т.е. МДО можно применять и там, где вводить вероятности некорректно или бессмысленно.

Однако зачастую прямая максимизация логарифма функции максимального правдоподобия и вычисление информации Фишера не представляются возможными.

График соотношения между сложностью модели (количеством параметров) и ее ошибкой (прогноз ошибки) для общего информационного критерия показан на рис. 1.9.

Как видно из рисунка, информационные критерии используются для нахождения модели, которая наилучшим образом балансирует ошибку модели и ее сложности, чтобы предотвратить чрезмерное недо- или переобучение. Следует отметить, что абсолютные значения критериев смысла не имеют – они указывают только на относительный порядок сравниваемых моделей.

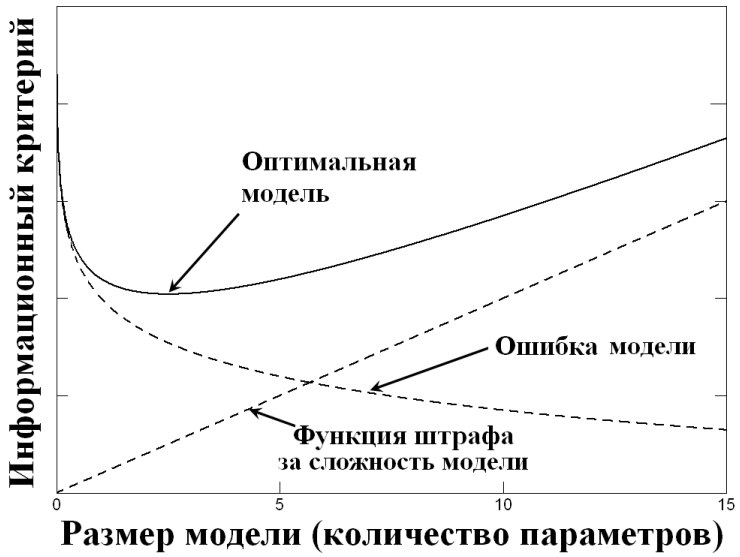


Рисунок 1.9 – Соотношения между сложностью модели и ее ошибкой

Считается, что минимум информационного критерия соответствует лучшей (оптимальной) сложности модели и чем меньше значение, тем лучше модель.

Другой важной причиной использования информационных критериев, является ограничение ненужного увеличения размера модели (ее вырождение).

Критерий МДО является более общим, чем популярные информационные критерии Акаике (AIC) [75] и Байесовский или критерий Шварца-Риссанена (BIC) [88].

Следует отметить, что в большинстве случаев BIC сводится к максимизации функции правдоподобия, поскольку, как правило, число параметров моделей совпадает с числом рассматриваемых моделей. Кроме того, критерий AIC никогда не выбирает модель с меньшим числом параметров, чем BIC.

Робастные варианты AIC и BIC, предложенные в работах [89-91], укладываются в схему

$$ICR = 2\sum_{i=1}^{K}\rho\left(e\left(i,\theta\right)\right)+\alpha_N, \tag{1.53}$$

где для $AICR$ $\alpha_N = 2tr\left(J^{-1}Q\right)$, $J = -M\left\{\frac{\partial^2\rho\left(e\left(i,\theta\right)\right)}{\partial\theta\partial\theta^T}\right\}$, $Q = M\left\{\frac{\partial\rho}{\partial\theta}\frac{\partial\rho^T}{\partial\theta}\right\}$, $M\{\cdot\}$ - символ математического ожидания; для $BICR$ $\alpha_N = \frac{1}{2}N\log K$.

1.4.2 Обучение ИНС прямого распространения.

В настоящее время существует большое число методов настройки параметров сети, отличающихся объемом используемой информации, влияющим как на динамические свойства алгоритмов, так и на их вычислительную сложность.

Обучение или адаптация ИНС происходит обычно путем настройки весов и других параметров сети так, чтобы отобразить возможные входные значения в значения на выходе сети.

Существует две основных парадигмы обучения: бат или оффлайн обучение, когда сети предоставляются все обучающие образы одновременно, и она может использовать их так часто, как это необходимо; и онлайн обучение, достоинство которого - естественная способность к адаптации к нестационарной среде, является важным в задачах обработки сигналов и автоматизированного управления.

Базовыми стратегиями обучения являются: обучение с учителем и обучением без учителя (рис. 1.10). В случае обучения с учителем, особенность состоит в том, что для каждого обучающего образа, предъявляемого сети, также предъявляется некоторая информация о корректности отклика сети.

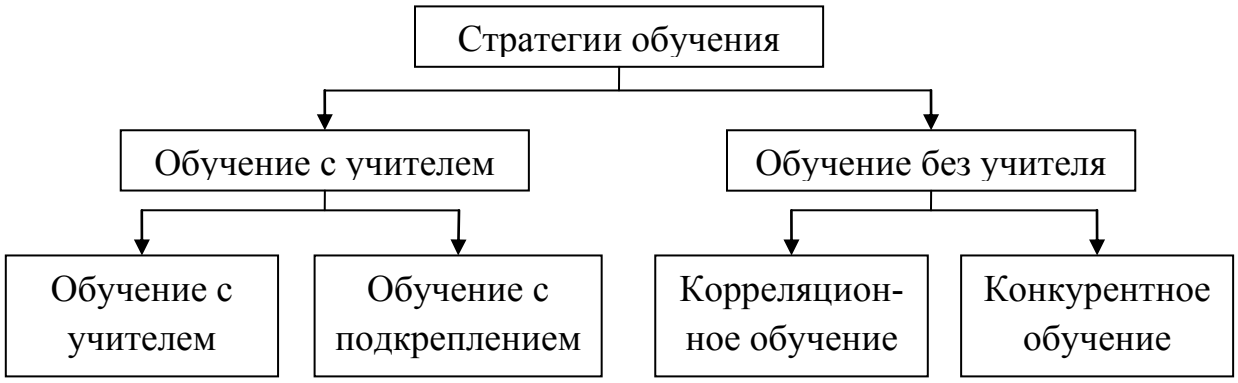


Рисунок 1.10 – Стратегии обучения ИНС

При обучении без учителя информация о желаемом значении на выходе сети, соответствующая каждому входному образу, отсутствует. Система должна обучаться, анализируя информацию, содержащуюся только в обучающих примерах, определяя их собственные свойства и группы.

Так как в задачах ИНС прямого распространения используется обучение с учителем, остановимся на нем подробнее.

Обучение ИНС – итеративный процесс [38, 92]. На каждой итерации вычисляются сетевые выходы для одного (или больше) образца в наборе обучения, и корректируются сетевые веса с целью уменьшения ошибки между фактическим сетевым выходом $(y_{i,p}^{L})(i=1,..N^{L})$ и целевым выходом $(y^{*}{}_{i,p})$ для данного образца. При использовании подходящей сетевой архитектуры и алгоритма обучения сетевые веса стремятся к значениям, при которых сетевой выход становится приемлемо близким к целевому выходу для каждого образца в наборе обучения.

При нейросетевом подходе задача обучения сети заключается в определении вектора ее параметров θ (весов, параметров активационных и базисных функций и т.д.), обеспечивающего минимум функционала

$$F(e) = \frac{1}{k}\sum_{i=1}^{k}\rho\big(e(i,\theta)\big), \tag{1.54}$$

т.е. являющегося решением системы уравнений

$$\frac{\partial F(e)}{\partial \theta_j} = \sum_{i=1}^{k}\rho'\big(e(i,\theta)\big)\frac{\partial e(i,\theta)}{\partial \theta_j} = 0, \tag{1.55}$$

где $\rho\big(e(i,\theta)\big)$ - некоторая функция потерь, зависящая от вида закона распределения помехи ξ; $e(i) = y(i) - \hat{y}(i)$; $\hat{y}(i)$ - выходной сигнал модели;

$\rho'\big(e(i,\theta)\big) = \dfrac{\partial\rho\big(e(i,\theta)\big)}{\partial e(i,\theta)}$ - функция влияния.

Введение весовой функции $\omega(i,\theta) = \dfrac{\rho'\big(e(i,\theta)\big)}{e(i,\theta)}$ позволяет записать систему уравнений (1.55) следующим образом:

$$\sum_{i=1}^{k}\omega\big(e(i,\theta)\big)e(i,\theta)\frac{\partial e(i,\theta)}{\partial \theta_j} = 0. \tag{1.56}$$

При этом минимизация функционала (1.54) будет эквивалентна минимизации взвешенного квадратичного функционала

$$F(e) = \frac{1}{k}\sum_{i=1}^{k}\omega(i,\theta)e^2(i,\theta). \tag{1.57}$$

Выбор функции ошибки имеет столь же существенное влияние на эффективность ИНС, как и выбор алгоритма обучения. Наиболее широко используемыми функциями ошибки являются квадратичная функция ошибки

(1.57) и нормализованная ее версия, называемая среднеквадратичной функцией ошибки.

Преимущество последней состоит в том, что она нечувствительна как к числу образцов в наборе обучения, так и к числу нейронов в выходном слое сети, что позволяет использовать ее для сравнения в различных задачах обучения.

Обучение ИНС является успешным при достижении сетевой ошибкой допустимого значения, определяемого из условий конкретной задачи. Следует отметить, что достижение минимального значения ошибки часто является нежелательным, т.к. приводит к явлению переобучения ИНС и ухудшению ее фильтрующих и обобщающих свойств. Исключение составляют задачи аппроксимации функций, для которых требуется высокая точность получаемых решений.

Первый теоретически доказанный алгоритм для настройки весов при обучении многослойных ИНС известен как алгоритм обратного распространения ошибки (ОР) [38, 93]. В настоящее время ОР может рассматриваться как эталонный тест, с которым сравниваются все другие методы обучения.

Термин “обратное распространение“ относится к процессу, с помощью которого могут быть вычислены производные функционала ошибки по параметрам сети. Этот процесс может использоваться в сочетании с различными стратегиями оптимизации [94].

В стандартной реализации алгоритма ОР, называемым также пакетным или автономным ОР, для минимизации целевой функции (функции ошибки) используется градиентный метод поиска (основанный на локальной линейной аппроксимации целевой функции), согласно которому на каждой итерации k вычисляется градиент минимизируемого функционала $\nabla f(\theta(k))$ и корректируются сетевые веса по правилу

$$\theta(k) = \theta(k-1) - \gamma \nabla f(k); \qquad (1.58)$$

где γ - итерационный параметр, влияющий на скорость обучения (обычно $\gamma \in (0,1)$), $\nabla f(k) = \left(\frac{\partial F(e)}{\partial \theta_1(k)} \quad \frac{\partial F(e)}{\partial \theta_2(k)} \quad \cdots \quad \frac{\partial F(e)}{\partial \theta_N(k)} \right)^{\mathrm{T}}$.

В литературе об ИНС уравнение (1.58) известно, как обобщенное дельта правило, а γ - как обучающий коэффициент. Правило (1.58) гарантирует уменьшение общей сетевой ошибки (функционала (1.54)) в каждом периоде. Однако на практике возможны увеличения (1.58), так как значение γ может оказаться достаточно большим для сети и алгоритм проскочит через минимум в данном направлении поиска.

Градиент $\nabla f(k)$ в (1.58) вычисляется в двух различных периодах – при прямом и обратном проходе.

Хотя пакетный алгоритм ОР зачастую дает удовлетворительные результаты, он обладает рядом существенных недостатков, которые проявляются, когда поверхности ошибки имеют локальные экстремумы, плоские области и овраги. В связи с этим были предложены различные эвристические модификации стандартного алгоритма ОР, направленные на преодоление этих недостатков.

Очевидно, что свойства (1.58) зависят от выбора коэффициента γ. Существуют различные рекомендации по выбору этого коэффициента. Так в теории стохастической аппроксимации этот коэффициент выбирается переменным и удовлетворяющим условиям

$$\lim_{k\to\infty}\gamma(k)=0,\ \sum_{k=1}^{\infty}\gamma(k)=\infty,\ \sum_{k=1}^{\infty}\gamma^2(k)=\infty. \qquad (1.59)$$

Таким условиям удовлетворяет, например, гармонический ряд $\gamma(k)=\gamma_0 k^{-a}$, где γ_0 - некоторая константа, $0.5<\alpha\le 1$. В теории стохастической аппроксимации нет рекомендаций по выбору константы γ_0, кроме ее положительности.

Выбор коэффициента γ в виде

$$\gamma(k)=\left\|\nabla_w(k)\right\|^{-2}$$

приводит к алгоритму Качмажа [95], известному в теории ИНС как алгоритм Уидроу-Хоффа [96]. В теории же оценивания данный алгоритм называют нормализованным алгоритмом МНК.

Отметим, что практически все подобные процедуры обучения реализуют методы оптимизации первого порядка, в которых используется вычисление градиента функционала. Среди этих методов наибольшей скоростью сходимости обладает метод сопряженных градиентов.

Существенно большую скорость сходимости имеют методы второго порядка, требующие вычисления вторых производных минимизируемого функционала [97-99]. Среди таких методов в первую очередь следует отметить метод Ньютона

$$\theta=\theta(k-1)-H_k^{-1}g_k, \qquad (1.60)$$

где H_k^{-1} - матрица, обратная матрице Гессе

$$H_k = \nabla^2 F(\theta) = \begin{bmatrix} \frac{\partial^2 F(\theta)}{\partial \theta_1^2} & \frac{\partial^2 F(\theta)}{\partial \theta_1 \theta_2} & \cdots & \frac{\partial^2 F(\theta)}{\partial \theta_1 \theta_N} \\ \frac{\partial^2 F(\theta)}{\partial \theta_2 \theta_1} & \frac{\partial^2 F(\theta)}{\partial \theta_2^2} & \cdots & \frac{\partial^2 F(\theta)}{\partial \theta_2 \theta_N} \\ \cdots & \cdots & \cdots & \cdots \\ \frac{\partial^2 F(\theta)}{\partial \theta_N \theta_1} & \frac{\partial^2 F(\theta)}{\partial \theta_N \theta_2} & \cdots & \frac{\partial^2 F(\theta)}{\partial \theta_N^2} \end{bmatrix}; \quad (1.61)$$

$$g_k = \begin{pmatrix} \frac{\partial F(\theta)}{\partial \theta_1} & \frac{\partial F(\theta)}{\partial \theta_2} & \cdots & \frac{\partial F(\theta)}{\partial \theta_N} \end{pmatrix}^T \quad (1.62)$$

(данные производные вычисляются в точке $\theta = \theta(k)$) или

$$\theta(k) = \theta(k-1) - \left[J_k^T J_k \right]^{-1} J_k^T e(k), \quad (1.63)$$

где J - якобиан, имеющий вид

$$J = \begin{bmatrix} \frac{\partial e_1}{\partial \theta_1} & \frac{\partial e_1}{\partial \theta_2} & \cdots & \frac{\partial e_1}{\partial \theta_N} \\ \frac{\partial e_2}{\partial \theta_1} & \frac{\partial e_2}{\partial \theta_2} & \cdots & \frac{\partial e_2}{\partial \theta_N} \\ \cdots & \cdots & \cdots & \cdots \\ \frac{\partial e_M}{\partial \theta_1} & \frac{\partial e_M}{\partial \theta_2} & \cdots & \frac{\partial e_M}{\partial \theta_N} \end{bmatrix}; \quad (1.64)$$

$e(k) = (e_1, e_2, \ldots, e_M)^T$.

Для обучения нейронных сетей применение метода Ньютона, основанного на выражении (1.60), нецелесообразно ввиду того, что, во-первых, этот метод можно использовать только, если матрица Гессе положительно определена; во-вторых, он требует существенных вычислительных затрат и значительного

объема памяти для хранения данных; в-третьих, при его использовании необходимы аналитические выражения для первых и вторых производных в каждой точке $w(k)$ [99].

Использование различных стратегий для регулирования положительной определенности матрицы Гессе и для выбора приемлемой длины шага на каждой итерации позволяет модифицировать метод Ньютона таким образом, чтобы придать ему свойство глобальной сходимости и снизить его вычислительные затраты.

Весьма эффективной является другая модификация метода Ньютона (называемая методом Левенберга-Марквардта) [99, 100] , в которой направление поиска отличается от задаваемого методом Ньютона. В данном методе к аппроксимирующей функции добавляется квадратичный штраф за отклонение от точки $\theta(k)$, а новая точка $\theta(k+1)$ определяется из условия минимума функции

$$f_k(\theta) + \alpha(k)\|\theta - \theta(k)\|^2 / 2,$$

где $f_k(\theta) = f(\theta(k)) + (\theta - \theta(k))^{\mathrm{T}} \nabla f(\theta(k)) + 0.5(\theta - \theta(k))^{\mathrm{T}} G(k)(\theta - \theta(k))$, что приводит к методу

$$\theta(k+1) = \theta(k) - [G(k) + \alpha(k)I]^{-1} \nabla f(k). \qquad (1.65)$$

При $\alpha(k) = 0$ метод переходит в метод Ньютона, при $\alpha(k) \to \infty$ направление поиска стремится к антиградиенту. Таким образом, метод Левенберга-Марквардта (1.65) представляет собой компромисс между этими двумя методами. За счет выбора $\alpha(k)$ можно добиться глобальной сходимости метода. Кроме того, данный метод пригоден не только для выпуклых функций, тогда как в методе (1.60) требуется положительная определенность матрицы Гессе.

Однако в рассмотренных модификациях метода Ньютона каждая итерация (как

и в основном методе Ньютона) требует больших вычислительных затрат (вычисление вторых производных, решение систем линейных уравнений), а скорость сходимости вдали от минимума не высока.

Весьма эффективными являются квазиньютоновские методы, представителями которых являются метод Давидона-Флетчера-Пауэлла, метод Бройдена и метод Бройдена-Флетчера-Шенно. Главный недостаток квазиньютоновских методов по сравнению, например, с методом сопряженных градиентов, заключается в необходимости хранить и пересчитывать матрицу $H(k)$ размерности $n \times n$, что для больших n требует большого объема памяти и выполнения большого числа арифметических операций.

Выводы по разделу 1 и постановка задач исследования

Многие практические и теоретические проблемы оптимизации характеризуются многомерностью пространства поиска. Эти проблемы включают в себя NP-полные задачи комбинаторной оптимизации, идентификации сложных структур или многомерной оптимизации функций.

Применение традиционных методов исследования, таких как градиентные, динамическое программирование, симплекс-метод для этих видов задач часто не дает желаемого результата, так как вычислительные затраты растут экспоненциально вместе с размерностью задачи. В связи с этим довольно часто для решения практических задач применяются эвристические методы, являющиеся более гибкими и эффективными и требующие гораздо меньших вычислительных затрат, несмотря на то, что они могут и не обеспечивать достижения глобального оптимального решения. К таким методам оптимизации относят метод имитации отжига, искусственные нейронные сети (ИНС), эволюционные алгоритмы (ЭА), муравьиные алгоритмы и т.д. Анализ существующих интеллектуальных систем обработки информации показывает,

что весьма перспективным представляется использование ИНС и ЭА, а также их комбинации в эволюционирующих искусственных нейронных сетях (ЭИНС).

Являясь универсальными аппроксиматорами, некоторые типы ИНС позволяют восстановить с заданной точностью любую сколь угодно сложную непрерывную нелинейную функцию. Наибольшее распространение при решении такой задачи получили статические ИНС прямого распространения (МП, РБС, ОРС, вейвлет -сети).

При синтезе или использовании ИНС возникают задачи структурной и параметрической оптимизации, соответствующие выбору оптимальной топологии сети и ее обучению (настройке параметров).

Традиционно процедуры выбора структуры опираются на МНК-оценки и сами критерии строятся на основе остаточных сумм квадратов. При таком подходе не учитываются, например, негауссовость, гетероскедастичность, зависимость ошибок наблюдений. Поэтому целесообразной представляется разработка методов выбора структуры нейросетевых моделей при нарушении классических предположений о свойствах ошибок наблюдений.

Отметим, что задача определения структуры сети, является важной и весьма сложной. Для ее решения могут использоваться, например, кросс-валидация, статистическая проверка гипотез, а также получающие в последнее время широкое распространение информационные критерии, укладывающиеся в схему (1.51). Информационные критерии используются для нахождения модели, которая наилучшим образом балансирует ошибку модели и ее сложности, чтобы предотвратить чрезмерное недо- или переобучение. Следует отметить, что абсолютные значения критериев смысла не имеют – они указывают только на относительный порядок сравниваемых моделей. Считается, что минимум информационного критерия соответствует лучшей (оптимальной) сложности модели и чем меньше значение, тем лучше модель.

Другой важной причиной использования информационных критериев, является ограничение ненужного увеличения размера модели (ее вырождение).

Если задача определения структуры является дискретной оптимизационной (комбинаторной), то поиск оптимальных параметров осуществляется в непрерывном пространстве с помощью классических методов оптимизации, основанных на МНК.

Для обучения (оценивания параметров) сети применяются, как правило, методы, требующие вычисления градиента используемого функционала (алгоритм обратного распространения ошибки (ОР), метод сопряженных градиентов, алгоритм Гаусса-Ньютона, Левенберга-Марквардта и т.д.) Несмотря на популярность этих методов не только при обучении ИНС, но и решении других задач оптимизации, они имеют ряд существенных недостатков, например, получаемое решение зависит от формы минимизируемой функции, вектора начальных условий, они застревают в локальных экстремумах, а при наличии нескольких экстремумов не обеспечивают нахождение глобального, их применение требует дифференцируемости минимизируемых функционалов. Кроме того, сложным является определение оптимальных параметров алгоритмов, обеспечивающих их максимальную скорость сходимости, точность, робастность и т.д.

Метод наименьших квадратов не является устойчивым, поскольку при наличии негауссовских помех (выбросов) целевая функция может расти до бесконечности и выбросы могут стать доминирующими измерениями, которые фактически проверяют реальную модель. В качестве альтернативы, для обеспечения робастности целевую функцию модифицируют таким образом, чтобы ограничить влияние наибольших измерений.

Классические робастные методы, ориентированы на симметричность засорения, когда выбросы одинаково часто появляются как в области отрицательных, так и в области положительных значений.

В более общей ситуации произвольного вида засорения, например, когда гауссовское засоряющее распределение имеет ненулевое математическое ожидание или когда засоряющее распределение является несимметричным, оценки, даваемые этими методами, будут смещенными. Это вызывает потребность в разработке робастных методов обучения, обеспечивающих несмещенность оценок при наличии помех, имеющих асимметричные распределения. Кроме того, отсутствие информации о статистических свойствах помех вызывает потребность в разработке методов аппроксимации помехи и использовании получаемых оценок в процедуре обучения сети с целью устранения смещения.

Попытки устранить недостатки традиционных методов синтеза и функционирования ИНС привели к появлению нового класса сетей – эволюционирующих ИНС (ЭИНС), в которых в дополнение к традиционному обучению используется другая фундаментальная форма адаптации – эволюция, реализуемая путем применения эволюционных вычислений.

Использование в ЭИНС этих двух форм адаптации – эволюции и обучения, позволяющих изменять структуру сети, ее параметры и алгоритмы обучения без внешнего вмешательства, делает данные сети наиболее приспособленными для работы в нестационарных условиях и наличии неопределенности относительно свойств исследуемого объекта и условий его функционирования.

Основным преимуществом использования эволюционных алгоритмов (ЭА) в качестве алгоритмов обучения является то, что многие параметры ИНС могут быть закодированы в геноме и определяться параллельно. Более того, в отличие от большинства алгоритмов оптимизации, предназначенных для потактового решения задачи, ЭА оперируют с множеством решений – популяцией, что позволяет достичь глобального экстремума, не застревая в локальных. При этом информация о каждой особи популяции кодируется в

хромосоме (генотипе), а получение решения (фенотипа) осуществляется после эволюции (отбора, скрещивания, мутации) путем декодирования.

Для предотвращения эффекта переобучения модели необходимо контролировать ее сложность. Так как получаемая нейросетевая модель, с одной стороны, должна быть достаточно простой и удобной для использования ее в прикладных задачах, а с другой – наиболее полно отражать свойства исследуемого объекта, ее качество определяется некоторым набором критериев, т.е. задача построения нейромодели является многокритериальной. Таким образом, возникает задача развития эволюционного метода многокритериальной оптимизации и использование его для создания Парето-эволюционирующих ИНС прямого распространения.

Несмотря на широкое применение ГА для решения широкого спектра оптимизационных задач, его основным недостатком является постоянное стремление к популяции, содержащей один локальный оптимум, что приводит к постоянному уменьшению генетического разнообразия популяции, вследствие чего снижается способность ГА к поиску глобального оптимума и/или адаптации к изменяющимся параметрам целевых функций. Следовательно, актуальной является задача усовершенствования методов эволюционной оптимизации на основе коэволюционного подхода, при котором используется несколько популяций, что обеспечивает более разнообразные решения по сравнению с другими ЭА, основанными на одной популяции. Такой подход представляется более эффективным при решении задач в условиях неопределенности и нестационарности.

Для обеспечения возможности практического применения методов построения ЭИНС для создания интеллектуальных систем обработки информации необходимо разработать соответствующие программные средства, а для эффективного применения методов и программных средств построения ЭИНС необходимо провести экспериментальное исследование их свойств и

характеристик, а также разработать рекомендации относительно их применения при решении практических задач.

Для достижения конечной цели диссертационной работы, заключающейся в развитии теоретических основ и разработке новых эволюционирующих ИНС для решения проблемы повышения качества интеллектуального анализа и обработки информации при наличии априорной и текущей неопределенности, необходимо решить следующие задачи:

– разработать новые и усовершенствовать существующие архитектуры ИНС с применением эволюционного и коэволюционного подходов, ориентированных на решение задачи интеллектуального анализа данных в условиях априорной и текущей неопределенности;

– разработать методы автоматического определения и коррекции структуры и параметров ИНС в зависимости от изменения свойств исследуемого объекта;

–разработать методы упрощения структур и методов обучения ИНС прямого распространения с целью ускорения процессов обработки информации при допустимой неточности;

– разработать Парето-эволюционирующие ИНС прямого распространения;

– разработать методы обучения эволюционирующих ИНС, которые обладали бы повышенным быстродействием и робастностью, при наличии негауссовских помех, в частности, помех, имеющих асимметричные распределения;

– провести экспериментальные исследования свойств и характеристик различных методов, разработать рекомендации по их применению, решить тестовые и практические задачи с помощью разработанных ЭИНС и внедрить результаты работы на действующих предприятиях, в проектных и научных учреждениях.

Решению поставленных задач, возникающих в процессе построения ЭИНС, посвящены следующие разделы.

РАЗДЕЛ 2
ПРИНЦИПЫ ПОСТРОЕНИЯ И ФУНКЦИОНИРОВАНИЯ ЭВОЛЮЦИОНИРУЮЩИХ НЕЙРОННЫХ СЕТЕЙ ПРЯМОГО РАСПРОСТРАНЕНИЯ

2.1 Основные теории эволюции и их применение при построении ЭИНС

Основой построения и функционирования ЭИНС являются генетические алгоритмы (ГА), которые абстрагируют фундаментальные процессы дарвиновской эволюции: естественного отбора и генетических изменений вследствие рекомбинации и мутации. Однако помимо дарвиновских ГА могут реализовывать механизмы эволюции Ламарка, изменяющие (улучшающие) хромосомы, и Болдуина, улучшающие приспособляемость хромосомы без ее изменения [101, 102].

2.1.1 Теория эволюции Дарвина.

Теорию эволюции, которую представил в 1859 году Чарльз Дарвин (1809-1882) в его книге "О происхождении видов" [103] можно обобщить с помощью простого алгоритма: мутация - изменчивость - соревнование - отбор - наследование.

Фитнес: ключевым понятием в теории эволюции Дарвина является идея пригодности или способности организмов выживать и размножаться. Геномные изменения в форме мутации или рекомбинации могут вызвать изменения приспособленности. Наиболее приспособленные организмы участвуют в скрещивании и их генетическая информация передается по наследству потомкам. Потомки наследуют геномные изменения и фенотипические черты, связанные с ними. Фенотипическое разнообразие определяется наследуемыми мутациями в ДНК-последовательности. Конкуренция также происходит и среди

аллелей, возможных генетических вариаций гена, за их присутствие в ДНК популяции. В зависимости от того, насколько успешными являются носители определенного аллеля, после нескольких поколений он будет либо зафиксирован в популяции либо исчезнет из генофонда. Однако, успех особи зависит лишь от аллеля в том случае, если изменение, произошедшее в фенотипе, оказывает влияние на внешний вид, функции тела или поведение рассматриваемого организма. Следовательно, в дарвинизме, эволюция является вторичным процессом. Организмы не адаптируются активно к их среде, а из множества различных характеристик и проявлений остаются лишь те, которые дают их обладателям преимущество в выживании или воспроизведении. Таким образом, центральную роль в дарвинизме играет отбор, в связи с чем, является необходимым рассмотрение различных типов отбора.

Естественный отбор: это отбор по биотическим или абиотическим факторам окружающей среды. Абиотическим фактором, например, является климат, а биотическим - давление со стороны хищники. Дарвин использовал этот термин для того, чтобы показать различие с искусственным отбором, и подчеркнул, что естественный отбор должен заканчиваться смертью либо неспособностью воспроизводства организма.

Половой отбор: В современной эволюционной биологии половой отбор считается частью естественного отбора. Сам Дарвин описал половой отбор как «менее строгий» чем естественный отбор, так как он не влияет на жизнь и смерть особей, но влияет только на число потомков, которое является косвенным условием выживания или успеха вида. Половой отбор является конкуренцией внутри вида за право воспроизводства. Результаты этих процессов не всегда совпадают с принципами естественного отбора, но часто даже противоречат им. Таким примером может являться хвост самца павлина и рога самцов оленей.

Искусственный отбор: искусственный отбор происходит в том случае, когда люди выбирают животных с желаемыми характеристиками и разводят их. Многие породы собак и лошадей являются результатом искусственного отбора.

Генный отбор: имеет большое значение в современной эволюционной теории, при таком отборе отдельные аллели конкурируют за максимальную частоту в популяции. В современной эволюционной биологии генный отбор заменил отбор особей по фенотипическим характеристикам из классической теории дарвинизма.

Стабилизирующий отбор: устраняет особей с экстремальным значением некоторой характеристики, например их размера. Возможным сценарием является пруд с рыбой различных размеров, в котором небольшая рыба вылавливается птицами, а большая - рыбаками. В связи с этим средние рыбы становятся большинством в пруду.

Распределенный отбор: Это полная противоположность стабилизирующему отбору, так как при таком отборе устраняются особи с посредственными значениями определенной характеристики. Так в примере с прудом, в этом случае рыбы среднего размера вылавливаются большими птицами. При этом мелкая и крупная рыба выживет.

Направленный отбор: Этот тип отбора является особенно интересным и направлен в одну сторону экстремальных и посредственный значений характеристики; например, в случае пруда, направленным отбором, будет случай, когда выдра охотится на малых и средних рыб. Таким образом, шансы на выживание выше у рыбы большего размера. Чем больше, тем безопаснее. При такого рода отборе этот вид рыбы будет постепенно увеличиваться в размерах.

Жесткий отбор: вид отбора отбор, при котором особь удаляется из популяции в том случае, когда некоторая ее характеристика (размер, цвет, и

т.д.) не достигает определенного значения. Например, все рыбы больше 30 см будут пойманы в сети рыбаков.

Мягкий отбор: при таком отборе используется не абсолютное значение характеристики, а относительное. В примере с прудом мягкий отбор будет означать, что наибольшая рыба будет поймана, независимо от того, насколько большой она является.

2.1.2 Эволюция Ламарка.

Однако дарвинизм не является единственной теорией эволюции. В дополнение к катастрофизму Жоржа Кювье (1769-1832), существует также ламаркизм, утверждающий, в отличие от дарвинизма, что не отбор является движущей силой эволюции, а наследование приобретенных признаков, полученных в процессе обучения. Жан-Батист де Ламарк (1744-1829) предположил, что соответствующие характеристики возникают из желания организмами их достигнуть (стремление к совершенствованию).

В отличие от дарвинизма, где эволюция является результатом конкуренции и отбора, в ламаркизме сами организмы контролируют эволюцию. Это достигается с помощью практики, обучения и частого использования специальных органов. Менее используемые органы со временем отмирают.

Самым популярным примером, иллюстрирующим идею ламаркизма, является эволюция шеи жирафа: жираф стремится достичь высоко расположенных на дереве листьев и вытягивает шею. Эта приобретенная особенность передается по наследству потомкам, которые снова вытягивают шею. Тем не менее, это очень простое объяснение преднамеренной адаптации приводит к некоторым вопросам с точки зрения современной биологии: Почему организмы должны иметь желание изменяться? Могут ли быть получены новые организмы с помощью обучения? Каким образом принимается решение о том, какая адаптация будет наследована, а какая - нет? Почему не наследуется

ампутированная нога? В биологии ламаркизм был бы возможен в том случае, если бы существовал механизм, переводящий изменения фенотипических признаков в последовательности соответствующих ген. Тем не менее, не следует полностью отбрасывать ламаркизм, так как он может дать некоторые ответы на проблемы современной генетики и медицины. В эпигенетике [104] [105], было установлено, что существуют специальные черты, которые могут быть унаследованы не будучи частью генетического кода. Это могло бы, например, объяснить возможную высокую подвижность большого пальца у предстоящих поколений людей [106] из-за частого его использования для набора текстовых сообщений на мобильных телефонах, которая, якобы, уже проявляется у некоторых людей, но эти особенности еще должны быть подтверждены. О возможность того, что приобретенные особенности поведения могут быть переданы от родителей к детям в настоящее время ведется серьезная дискуссия, результатом которой может быть глубокий сдвиг в нашем понимании наследования, т.е., привести к выводу о том, что гены также обладают "памятью" [107], [108].

Такой подход был использован в гибридном алгоритме обучения ИНС, предложенном Яо и Лю [5] в эволюционирующих сетях прямого распространения. На каждой итерации в процессе эволюции гибридный алгоритм выбирает сеть из популяции и обучает ее с помощью алгоритма обратного распространения ошибки определенное количество эпох. Если обучение повышает производительность сети, обученная сеть вместе со связанным с нею значением фитнес-функции будет возвращена обратно в популяцию для дальнейшей эволюции. Этот механизм сохраняет приобретенные характеристики, полученные на основе обучения, что соответствует эволюции по Ламарку. Перспективные результаты эволюции Ламарка получили Браун и Загорский [109] при построении сети обратного распространения для решения задач классификации. Браун и Загорский

утверждают, что тонкая настройка каждой сети с помощью быстрого алгоритма обратного распространения позволяет уменьшить пространство поиска до набора локальных оптимальных точек (седловых точек), в результате чего поиск глобального оптимума становится более эффективным. Стоит отметить, что предыдущие исследования в эволюции Ламарка (как упоминалось выше) обычно используют локальные методы поиска с высокой вычислительной сложностью. Это может представлять серьезную нагрузку для гибридных алгоритмов. Хотя эти исследования и продемонстрировали возможности эволюции Ламарка, большинство из них не показали реального улучшения времени обучения, в результате чего трудно обнаружить реальную выгоду от объединения локального и эволюционного поисков. Таким образом, эта глава оценивает возможности обучения Ламарка, путем сравнения фактического времени, затраченного в экспериментах.

2.1.3 Эффект Болдуина.

Так как ламаркизм не может быть обнаружен в биологических системах, был разработан биологически более правдоподобный механизм, основанный на эффекте Болдуина - по имени американского психолога Джеймса Болдуина (1861-1934), который выдвинул эту гипотезу в 1896 году (примерно в то же время Morgan (1896) и Osborn (1896) пришли к этой мысли независимо от Болдуина). Согласно эффекту Болдуина эпигенетические факторы влияют на формирование генома не менее эффективно, чем действие естественного отбора. Таким образом, изменившееся поведение может вести к изменению факторов отбора и, соответственно, к новому направлению эволюционного развития.

В частности, поведенческие решения и стереотипы, принимаемые людьми и передающиеся от поколения к поколению в виде культурных практик и

традиций, следует рассматривать важнейшим фактором, формирующим человеческий геном.

Хинтон и Ноулан [110] были первыми, кто применил обучение Болдуина для эволюционирующих нейронных сетей. В своих экспериментах они применили случайный поиск к каждой нейронной сети, полученной с помощью эволюционного поиска. Случайный поиск при этом не изменял сеть, а лишь обновлял значение ее фитнес-функции с целью показать расстояние до глобального оптимума. Их экспериментальные результаты показали, что гибридный алгоритм позволяет найти глобальный оптимум, который является недостижимым при использовании лишь эволюционного поиска.

В работе Экли и Литтман [111], обучение Болдуина было использовано в эволюции адаптивных агентов, борющихся за выживание в искусственной экосистеме. Поведение каждого агента определялось с помощью двух нейронных сетей, отвечавших за оценку ситуации и принятие решения. Авторы обнаружили, что эволюция без обучения приводит к неправильному поведению агентов, в следствие чего они становятся непригодными к выживанию в экосистеме и быстро вымирают. С другой стороны, на основе обучения Болдуина могут быть получены агенты, популяции из которых могут выживать в течение длительного времени. Экли и Литтман показали, что эффект Болдуина выгоден для эволюции, поскольку он позволяет агентам выживать в экосистеме. Несмотря на то, что был продемонстрирован положительный эффект от применения обучения Болдуина, также было показано, что при определенных обстоятельствах эффект Болдуина может приводить к неэффективным гибридным алгоритмам [112-114]. Это стимулирует исследование эффективности обучения Болдуина с целью определения ситуаций, при которых снижается производительность гибридных алгоритмов.

Существуют различные модели эффекта Болдуина [115, 116], заключающегося в том, что процесс эволюции направляется путем обучения.

В соответствии с эффектом Болдуина, особь будет жить дольше, если она лучше обучилась определенным навыкам, т.е. вероятность быть замененной в процессе эволюции у нее будет небольшой. Если она сможет выжить в течение достаточного числа поколений, то, возможно, она эволюционирует путем генетических операций в такой генотип, приспособленность которого будет эквивалентна особи с выученными навыками. Несмотря на то, что выученные навыки невозможно генетически идентифицировать в фенотипе, существуют примеры того, что эффект Болдуина способен направлять генотипические изменения [117].

Обучение Болдуина можно рассматривать как своего рода изменчивость, направляемую фенотипом. Следовательно, обучение может увеличить дисперсию процесса отбора. Преимуществом обучения Болдуина является то, что оно позволяет сделать более плоским рельеф фитнес-функции вокруг оптимальных областей [114]. Увеличение генетического полиморфизма и уплощение рельефа фитнес-функции позволяет улучшить процесс эволюции. Общий эффект от обучения Болдуина заключается в том, что оно позволяет найти глобальный оптимум [114].

2.2 Эволюционная оптимизация ИНС

Как отмечалось выше, эволюция в ЭИНС может касаться архитектуры сети, ее весов, вида и параметров АФ(БФ), алгоритма обучения (рис. 2.1) [118].

При использовании ГА для построения ЭИНС основными проблемами являются выбор метода кодирования возможного решения и генетических операторов; в различных задачах используются различные методы кодирования и генетические операторы.

Так эволюционная оптимизация топологии сети состоит из двух фаз: кодирование (прямое, когда хромосома содержит всю информацию о топологии

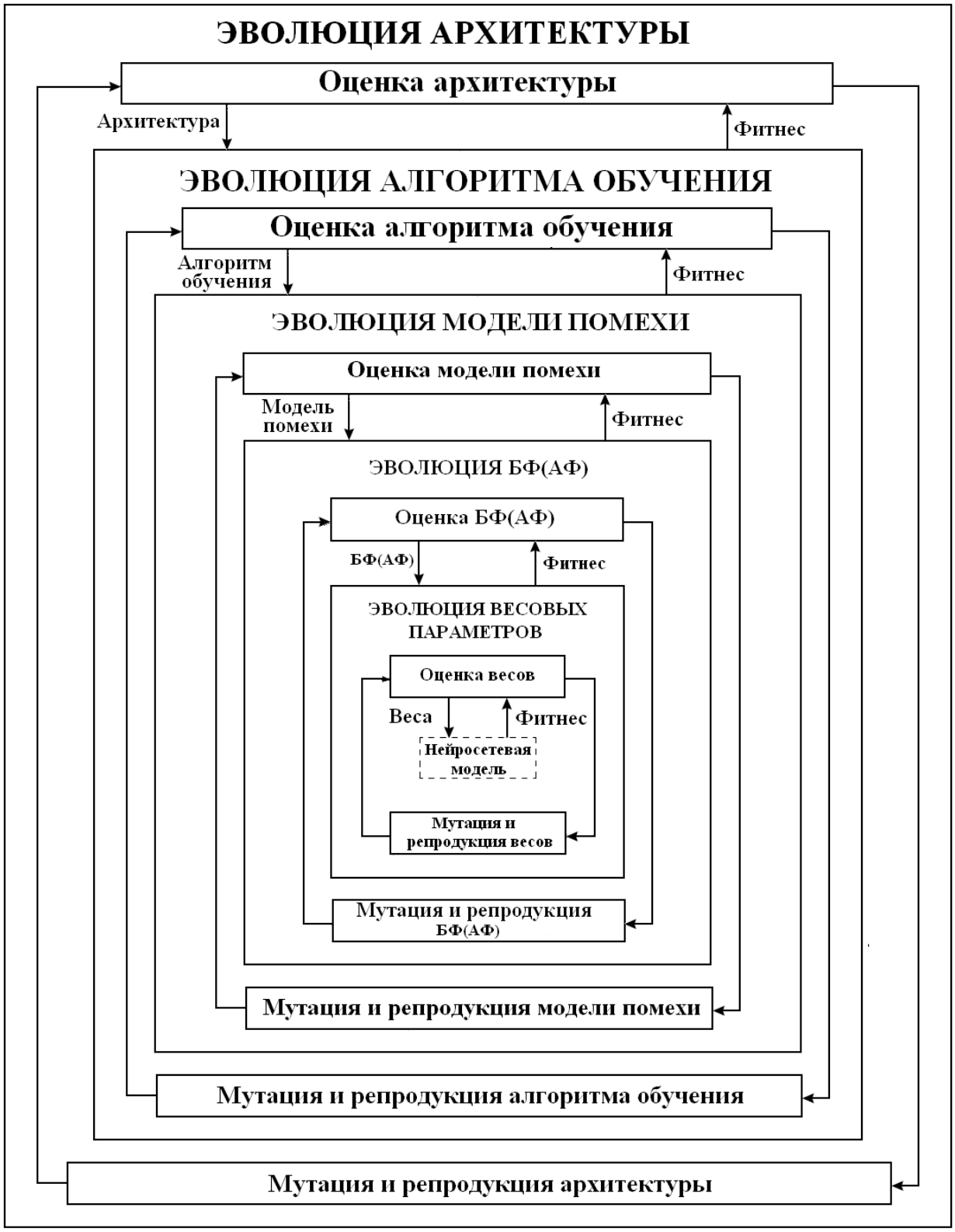


Рисунок 2.1 – Структура ЭИНС

сети (слои, нейроны, связи), либо непрямое, когда в хромосоме кодируется только число слоев и число нейронов в каждом слое, что существенно уменьшает длину хромосомы) и реализация ЭА.

Обычно настройка параметров осуществляется после того, как получена приемлемая топология сети, что характеризуется допустимым значением приспособленности [119, 120]. Эволюционная настройка весовых параметров сети также требует задания способа их кодирования (двоичное или вещественное), влияющего на выбор операторов скрещивания и мутации и определяющего исследуемое пространство поиска и отношение между ними генотипом и фенотипом («один-к-одному», «один-ко-многим» и «многие-к-одному»). При этом, однако, необходимо учитывать следующее.

Вследствие случайной инициализации весов и применения конкретного алгоритма обучения возникают ошибки в оценивании приспособленности (фитнесс-функции). В первом случае изменение начальных значений приводит к тому, что один и тот же генотип после обучения может иметь разную приспособленность. Попытка получения усредненных результатов путем обучения с разными начальными условиями приводит к резкому возрастанию вычислительных затрат. Второй источник ошибок вызван тем, что при одних и тех же начальных условиях различные алгоритмы обучения могут приводить к различным результатам, особенно если используемый критерий обучения имеет несколько экстремумов, что приводит к «один-ко-многим» отображению генотипов в фенотипы [4, 5]. Поэтому, как показано в [4, 5, 121, 122], получить более точные результаты можно путем одновременной эволюции архитектуры и весов.

Ввиду того, что ИНС образуют нейроны, осуществляющие нелинейные преобразования сигналов в соответствии с видом их активационных функций, выбор этих функций также играет важную роль в нейросетевом представлении, так как существенно влияет на сложность вычислений и для упрощения обычно

предполагается, что активационные функции нейронов одинаковы и выбраны экспертом. Информация об активационной функции также может быть закодирована в хромосоме.

Эффективность того или иного алгоритма обучения сети существенно зависит от ее архитектуры, поэтому для полученной каким-либо образом и неизменяющейся топологии сети может быть определен оптимальный алгоритм настройки ее параметров. Однако построение такого алгоритма в общем случае невозможно, поэтому возникает задача автоматической коррекции (адаптации) правила обучения в зависимости от изменения условий функционирования, архитектуры сети. При использовании эволюционного подхода для этих целей адаптация правила обучения может рассматриваться как динамический процесс «обучения обучению» сети [4, 5, 123-125]. Следует отметить, что в этом случае может осуществляться как адаптация отдельных параметров алгоритма, например, коэффициента усиления, параметра взвешивания, так и изменение его структуры, например, переход от алгоритма Гаусса-Ньютона к алгоритму Левенберга-Марквардта и пр.

Так как вследствие своей стохастической природы ЭА всегда пытаются найти глобальный экстремум, в некоторых задачах эволюционное обучение происходит медленнее, чем традиционное, использующее, например, ускоренное обратное распространение, а в некоторых – быстрее [126-128]. Объединить преимущества обоих подходов удается при гибридном обучении, когда ЭА применяются для приближенного определения области глобального экстремума в пространстве параметров, а традиционные алгоритмы – для тонкой настройки параметров, т.е. результаты эволюционной (глобальной) оптимизации используются в качестве начальных условий для процедуры локальной оптимизации [129-131].

Блок-схема такого обучения приведена на рис. 2.2.

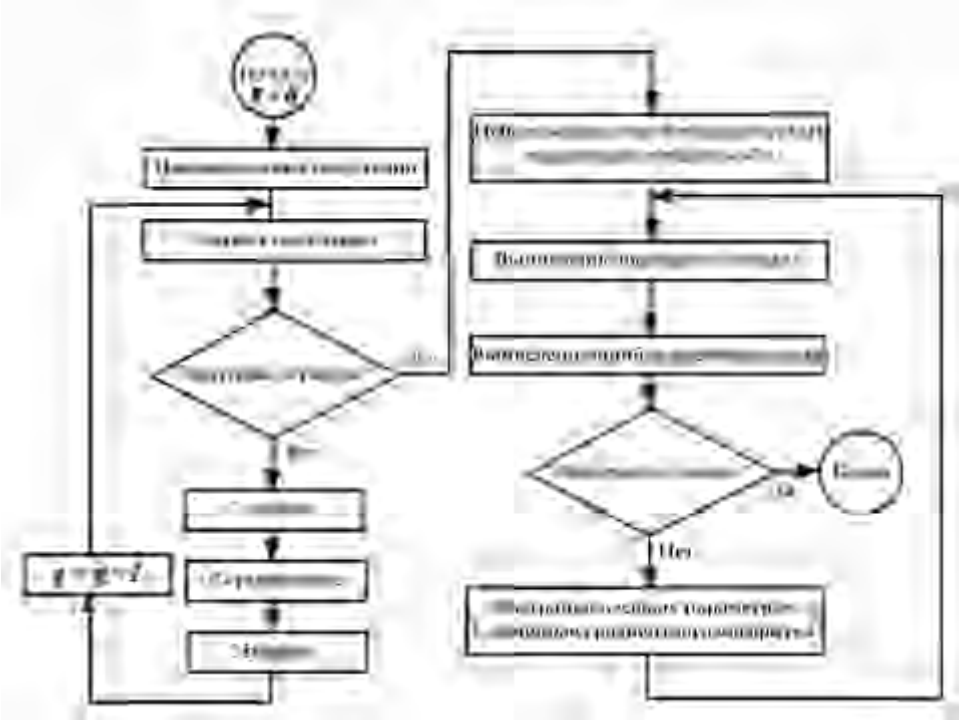


Рисунок 2.2 – Схема эволюционного алгоритма настройки ИНС

Подобный подход был использован в гибридном алгоритме, предложенном в [132] для синтеза эволюционирующих сетей прямого распространения. На каждой итерации в процессе эволюции гибридный алгоритм выбирает сеть из популяции и обучает ее с помощью алгоритма ОР определенное количество эпох. Если обучение повышает производительность сети, обученная сеть вместе со связанным с нею значением фитнесс-функции будет возвращена обратно в популяцию для дальнейшей эволюции. Этот механизм сохраняет приобретенные характеристики, полученные на основе обучения, что соответствует эволюции Ламарка. Такой подход использовался в [129] при построении сети обратного распространения для решения задач классификации. Здесь показано, что тонкая настройка каждой сети с помощью алгоритма ускоренного ОР позволяет уменьшить пространство поиска до набора локальных оптимальных (седловых) точек, в результате чего поиск глобального оптимума становится более эффективным.

Впервые использование эффекта Болдуина для обучения эволюционирующих нейронных сетей было предложено в [110]. В этой работе случайный поиск применялся к каждой нейронной сети, полученной с помощью эволюционного поиска. Экспериментальные результаты показывают, что данный гибридный алгоритм может найти глобальный оптимум, который является недостижимым при использовании лишь эволюционного поиска.

Как отмечается в [112], эффект Болдуина в некоторых случаях может приводить к неэффективным гибридным алгоритмам, в то время как использование эволюции Ламарка является весьма эффективным в нестационарных условиях функционирования сети.

ГА совместно с алгоритмом ОР использовался в [133] для одновременной настройки топологии и весов МСП.

В [134] был предложен нейроэволюционный метод растущей топологии, использующий генетическое кодирование топологии сети и ее весов. При этом генетические операторы применяются как для введения нового нейрона, так и для удаления старого.

В работе [133], улучшенный ГА используется для обучения трехслойной полносвязной сети прямого распространения, которая после обучения может стать частично связанной. В [134] ГА совместно с алгоритмом ОР применяется для настройки как архитектуры, так и весов многослойного персептрона, а в работе [135] алгоритм ускоренного ОР используется для улучшения решения и достижения ближайшего локального минимума от найденного с помощью ГА решения. При этом популяция инициализируется особями, представляющими из себя ИНС с различным числом скрытых слоев. Мутация, многоточечный кроссовер, добавление/удаление нейрона и ускоренное ОР обучение используются в качестве генетических операторов. ГА осуществляет поиск и оптимизацию архитектуры, производит первоначальную настройку весов для этой архитектуры, а также определяет параметры алгоритма обучения.

Операторы ускоренного ОР обучения и процедуру удаления нейрона можно рассматривать как элементы стратегии эволюции Ламарка. В генетическом методе обратного распространения [133, 136] ГА выбирает начальные веса и изменяет количество нейронов в скрытом слое с помощью пяти генетических операторов: мутации, многоточечного кроссовера, добавления, удаления и замены нейронов скрытого слоя. Алгоритм ОР используется затем для настройки этих весов, т.е. четко разделяется глобальный и локальный поиск. Таким образом, эта стратегия избегает использования ламаркизма. Данный метод модифицирован в [137] путем интеграции ускоренного ОР как оператора обучения и реализует эволюцию Ламарка.

В [138] ЭА применялся для одновременной оптимизации структуры и весов РБС при использовании двоичного кодирования.

2.3 Эволюционирующие ИНС

Рассмотрим классический ГА, содержащий следующие шаги:

1. Создание начальной популяции.

1.1. Инициализация хромосомы каждой особи.

1.2. Оценивание начальной популяции.

2. Этап эволюции - построение нового поколения.

2.1. Отбор кандидатов на скрещивание (селекция).

2.2 Скрещивание, т.е. порождение каждой парой отобранных кандидатов новых индивидов.

2.3. Мутация.

2.4. Оценивание новой популяции.

Критерием останова алгоритма обычно являются либо прохождение максимально допустимого количества поколений, либо достижение функцией

приспособленности какой-либо особи некоторого заранее заданного порогового значения.

При переходе от ИНС к ЭИНС для всех типов сетей используются общие эволюционные процедуры (инициализация популяции, оценка популяции, селекция, скрещивание, мутации), а различия заключаются лишь в способе кодирования структуры и параметров той или иной ИНС в виде хромосомы. Схема работы такого эволюционного алгоритма настройки ИНС приведена на рис.2.2.

2.3.1 Инициализация популяции.

В начале работы НА случайным образом инициализируется популяция P_0, состоящая из N особей (ИНС): $P_0 = \{H_1, H_2, .., H_N\}$. Каждая особь в популяции при этом получает свое уникальное описание, закодированное в хромосоме $H_j = \left\{ h_{1j}, h_{2j}, .. h_{Lj} \right\}$, которая состоит из L генов, где $h_{ij} \in [w_{\min}, w_{\max}]$ - значение i-го гена j-ой хромосомы ($w_{\min}$ - минимальное, и $w_{\max}$ - максимальное допустимые значения соответственно). На рис.2.3 и 2.4 показаны примеры МП и РБС, форматы хромосом и соответствие между параметрами сетей и хромосомами (представление их параметров в хромосоме). Следует отметить, что длина хромосомы зависит от размерности идентифицируемого объекта и максимально допустимого количества нейронов.

Существуют, однако, ЭА, которые используют хромосомы переменной длины (не имеют фиксированного числа генов), где каждый ген кодирует лишь часть решения, и кодирование не зависит от взаимного расположения генов (только от их значений). Целесообразность применения хромосом переменной длины в основном зависит от области использования эволюционных вычислений (ЭВ).

При использовании хромосом переменной длины для кодирования структуры МП их длину можно вычислить с помощью следующей формулы:

$$S = (I + D) H_1 + (H_1 + D) H_2 + ... + (H_n + D) O,$$

где I – число входов модели; H_j – число активных нейронов j- го скрытого слоя; O – число выходов модели; n – число скрытых слоев; D – число дополнительных генов, активирующих нейроны, кодирующих их смещения и вид базисной функции. При наличии помех измерений длина хромосомы также может быть увеличена в целях хранения параметров фильтра помех.

Если число нейронов в каждом скрытом слое ограничено и не может превышать некоторого количества $H_{\max}$, то максимальная длина хромосомы вычисляется с помощью формулы:

$$S_{\max} = (I + D) H_{\max} + (n - 1) * H_{\max} * (H_{\max} + D) + (H_{\max} + D) O.$$

Для РБФ длина хромосомы может быть определена следующим образом:

$$S = H * (2R + \mathrm{D} + 1) + 1,$$

где H – число активных нейронов шаблонного слоя; R – число входов модели; D – число дополнительных генов, отвечающих за активацию нейронов и вид их базисных функций.

Максимальная длина хромосомы вычисляется соответственно с помощью формулы:

$$S = H_{max} * (2R + D + 1) + 1.$$

В результате использования хромосом переменной длины возникают особи с особыми генетическими сегментами кода (интронами), которые не используются для кодирования характеристик [139, 140].

Обычно интроны используются в ЭА:

1. Как некодирующие биты, равномерно добавленные в генетический код (в этом случае интроны лишь заполняют пространство между активными генами хромосомы).

2. Как нефункциональные части генетического кода, т.е. части решения, которые фактически ничего не делают, тем самым не влияют на приспособленность хромосомы (обычно это происходит в генетическом программировании и в хромосомах, которые подвергаются циклу развития после своего рождения).

3. Как апостериорно бесполезные части хромосомы, т.е. части генетического представления, которые не принимают участия в вычислении ее приспособленности (как правило, это проявляется в некоторых видах конкуренто-обучаемых нейронных сетей, в которых только нейроны-победители в отличие от остальных нейронов, являющихся апостериорно бесполезными, влияют на результат работы сети).

Интроны появляются и в других видах нейронных сетей, где их называют потенциально полезным мусором.

Контроль количества интронов в популяции осуществляется путем:

- использования специальных операторов, которые изменяют длину хромосомы и добавляют или удаляют интроны (экспериментальные результаты показывают, что добавление некоторого числа интронов в целом улучшает свойства ЭА);

- использования оператора селекции: в зависимости от числа интронов, значение фитнесс функции особи будет увеличиваться или уменьшаться.

Как видно из рисунков, каждая хромосома состоит из генов, в которых хранится информация о соответствующих параметрах сети. В начале хромосомы идут гены, которые содержат информацию о параметрах помехи и являются активными лишь в случае идентификации зашумленных объектов. Следующий ген кодирует информацию о смещении нейрона выходного слоя сети (b_5 для МП и w_0 для РБС). Затем идут блоки генов, кодирующие параметры соответствующих нейронов скрытого слоя. Первый ген каждого такого блока (1/0) определяет, присутствует ли соответствующий нейрон в структуре сети, т.е. участвует он или нет в вычислении выходной реакции сети на поступивший входной сигнал. Так как в качестве $f(\bullet)$ в МП обычно используют сигмоидальные АФ, например, вида

$$f(x) = (1 + e^{-\alpha x})^{-1}, \tag{2.1}$$

то следующий ген в его хромосоме (α) определяет форму АФ данного нейрона.

Наиболее часто в качестве БФ выбираются функции, приведенные в табл.2.1.

Если же при исследовании объектов осуществляется нормализация сигналов, целесообразным является применение модифицированной гауссовской функции (МГБФ) вида

$$\Phi(x, \lambda_1, \lambda_2) = \begin{cases} \exp\left\{ -\dfrac{(\lambda_2 - \lambda_1)^2 / 4}{(x - \lambda_1)(\lambda_2 - x)} \right\} & \textit{при } x \in (\lambda_1, \lambda_2); \\ 0 & \textit{в противном случае,} \end{cases} \tag{2.2}$$

которая, в отличие от классической гауссовской функции, равна нулю при $x \notin (\lambda_1, \lambda_2)$.

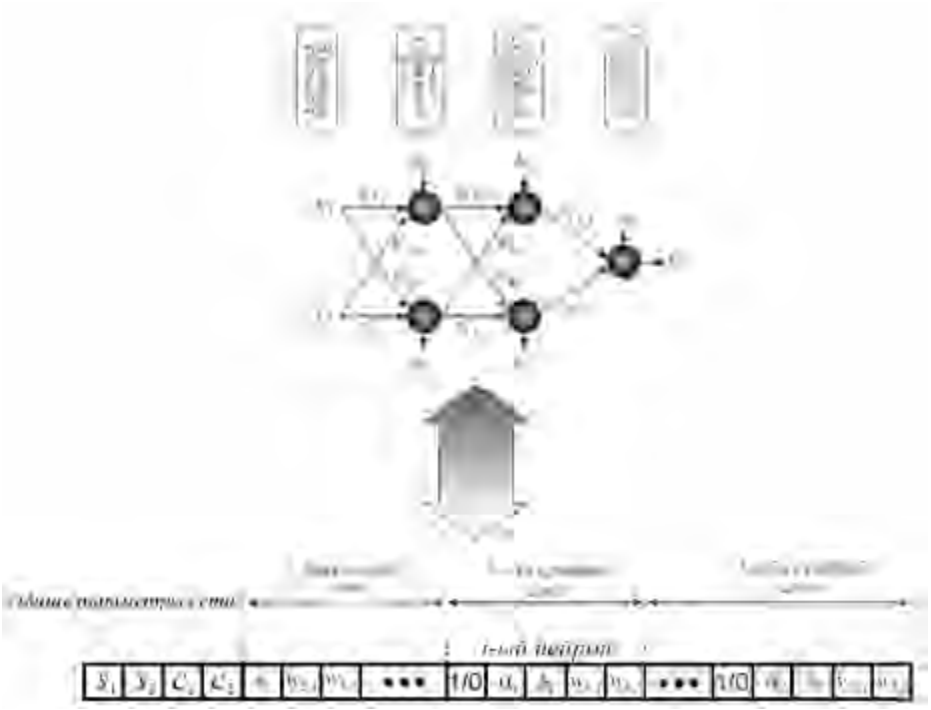


Рисунок 2.3 – Форматы хромосомы МП

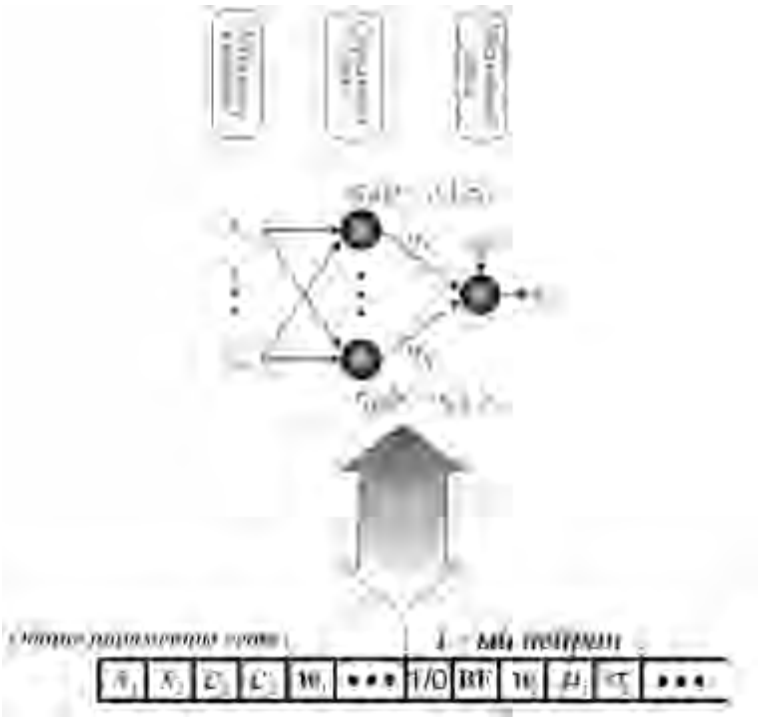


Рисунок 2.4 – Форматы хромосомы РБС

В ряде случаев, в частности, когда центры БФ распределены равномерно, а радиусы отличаются незначительно, весьма эффективным является использование нормализованных БФ (НБФ) [141]

$$\bar{\phi}_i(x,\mu_i)=\frac{\phi_i(x,\mu_i)}{\sum_{m=1}^{N}\phi_m(x,\mu_m)}. \tag{2.3}$$

Свойство данной сети

$$\sum_{m=1}^{N}\phi_m(x,\mu_m)=1$$

делает ее привлекательной для многих приложений.

Отметим, что в случае M-мерного сигнала $\mathbf{x}[k]$ БФ i-го нейрона будет иметь вид

$$\Phi_i(\mathbf{x},\boldsymbol{\mu})=\prod_{m=1}^{M}\phi_{im}\left(x_m,\mu_{im}\right), \tag{2.4}$$

т.е.

$$\Phi_i(\mathbf{x}[k],\boldsymbol{\mu})=\prod_{j=1}^{M}\exp\left\{-\frac{\left(x_j[k]-\mu_{ij}\right)^2}{\sigma_{ij}^2}\right\} \tag{2.5}$$

при выборе гауссовской БФ и

Таблица 2.1 – Наиболее часто выбираемые БФ

№ п.п.	Наименование	Вид функции
1.	Гауссовская	$\Phi(x)=\exp\left\{-\frac{\|x-\mu\|^2}{\sigma^2}\right\}$
2.	«Мексиканская шляпа»	$\Phi(x)=\left(1-\frac{(x-\mu)^2}{\sigma^2}\right)e^{-\frac{(x-\mu)^2}{\sigma^2}}$
3.	Лапласа	$\Phi(x)=\exp\left\{-\frac{\|x-\mu\|}{\sigma}\right\}$
4.	Релея	$\Phi(x)=\frac{2(x-\mu)}{\sigma}\exp\left\{-\frac{(x-\mu)^2}{\sigma^2}\right\}$
5.	Обобщенная гауссовская функция	$\Phi(\boldsymbol{x})=e^{-(\boldsymbol{x}-\mu)^T R^{-1}(\boldsymbol{x}-\mu)}$ где $R^{-1}=\left[r_{ij}^k\right], i,j=\overline{1,M}, k=\overline{1,N}$ (M – размерность входного сигнала, N – количество нейронов) – масштабирующая матрица;
6.	Косинусоидальная	$\Phi(x)=\cos\left\{\frac{2\pi(x-\mu)}{\sigma}\right\}$
7.	Параболическая	$\Phi(x)=1-\frac{(x-\mu)^2}{\sigma^2}$

*В таблице приняты следующие обозначения: μ, σ – центры и радиусы базисных функций соответственно; $\|\cdot\|$ - евклидова норма.

$$\hat{\Phi}_i(\mathbf{x}[k],\lambda_1,\lambda_2)=\begin{cases}\prod_{m=1}^{M}\exp\left\{-\dfrac{(\lambda_{im,2}-\lambda_{im,1})^2/4}{(x_m[k]-\lambda_{im,1})(\lambda_{im,2}-x_m[k])}\right\} & \textit{при } \mathbf{x}[k]\in(\boldsymbol{\lambda}_{i1},\boldsymbol{\lambda}_{i2});\\ 0 & \textit{в противном случае,}\end{cases} \tag{2.6}$$

при выборе модифицированной гауссовской функции.

Нормализованная БФ будет выглядеть следующим образом:

$$\bar{\Phi}_i(\mathbf{x}[k],\boldsymbol{\mu}_i)=\frac{\Phi_i(\mathbf{x}[k],\boldsymbol{\mu}_i)}{\sum_{m=1}^{N}\Phi_m(\mathbf{x}[k],\boldsymbol{\mu}_m)}\ . \tag{2.7}$$

Применение нормализованных РБС (НРБС) обеспечивает, в отличие от РБС, в равной степени покрытие всех точек входного пространства, делая сеть менее чувствительной к неудачному выбору центров. Однако нормализация существенно изменяет форму БФ, причем на эту форму оказывает влияние как ширина исходных БФ (при уменьшении σ_i^2 форма нормализованных функций становится более прямоугольной), так и их взаимное расположение, характеризуемое параметрами μ_i. При неравномерном же распределении центров исходных БФ или при их различной ширине, максимумы НБФ могут быть опущены по отношению к центрам, а сами НБФ могут стать немонотонно убывающими. Последнее приводит к неправильной реакции нейрона на входные сигналы.

Как следует из приведенных формул, при реализации ИНС необходимо вычисление экспоненциальных БФ и их производных, что требует значительных временных затрат, существенно возрастающих с ростом размерности задачи и увеличением количества БФ. Одним из возможных

способов упрощения вычислений является использование кусочно-линейной аппроксимации БФ.

Достаточно простой и в то же время эффективной является следующая аппроксимация [142-143]:

- гауссовская БФ (см.табл.2.1)

$$\phi(x)=\begin{cases}\dfrac{0,77}{1,2\sigma}\left(x-\mu+1,7\sigma\right) \textit{ при } \mu - 1.7\sigma \le x < \mu - 0.5\sigma;\\ 0,77+\dfrac{0,23}{0,5\sigma}\left(x-\mu+0,5\sigma\right) \textit{ при } \mu - 0.5\sigma \le x < \mu;\\ 0,77-\dfrac{0,23}{0,5\sigma}\left(x-\mu-0,5\sigma\right) \textit{ при } \mu \le x < \mu + 0.5\sigma;\\ -\dfrac{0,77}{1,2\sigma}\left(x-\mu-1,7\sigma\right) \textit{ при } \mu + 0.5\sigma \le x < \mu + 1.7\sigma;\end{cases} \tag{2.8}$$

- модифицированная гауссовская БФ

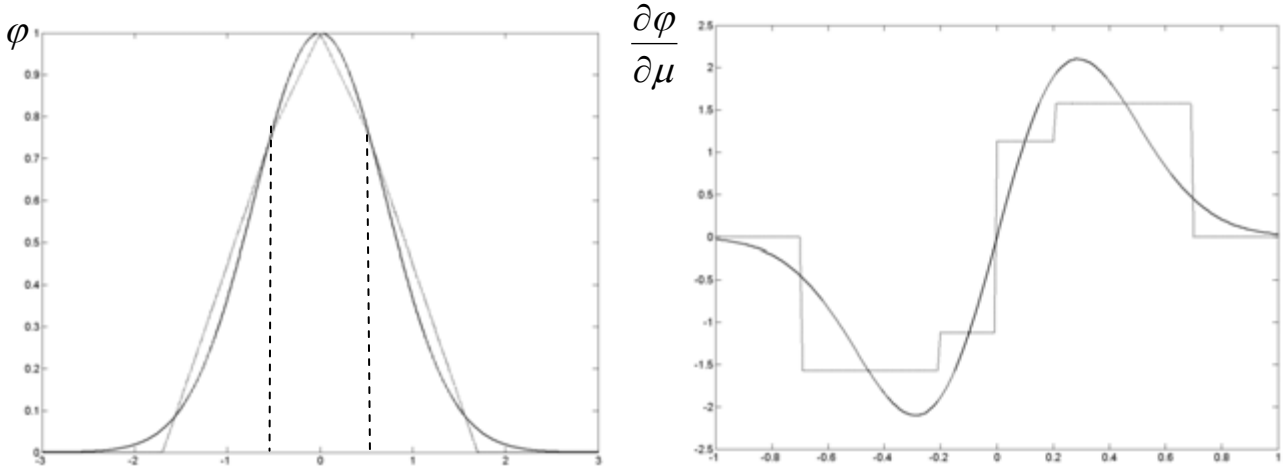
$$\hat{\varphi}(x)=\begin{cases}0 & \text{при } \mu-\sigma \le \mathrm{x} < \mu-0.9\sigma;\\ 0,73+\dfrac{0,73}{0,4\sigma}\left(x-\mu+0,5\sigma\right) & \text{при } \mu-0.9\sigma \le \mathrm{x} < \mu-0.5\sigma;\\ 0,91+\dfrac{0,18}{0,2\sigma}\left(x-\mu+0,3\sigma\right) & \text{при } \mu-0.5\sigma \le \mathrm{x} < \mu-0.3\sigma;\\ 0,99+\dfrac{0,08}{0,2\sigma}\left(x-\mu+0,1\sigma\right) & \text{при } \mu-0.3\sigma \le \mathrm{x} < \mu-0.1\sigma;\\ 0,99 & \text{при } \mu-0.1\sigma \le \mathrm{x} < \mu+0.1\sigma;\\ 0,99-\dfrac{0,08}{0,2\sigma}\left(x-\mu-0,1\sigma\right) & \text{при } \mu+0.1\sigma \le \mathrm{x} < \mu+0.3\sigma;\\ 0,91-\dfrac{0,18}{0,2\sigma}\left(x-\mu-0,3\sigma\right) & \text{при } \mu+0.3\sigma \le \mathrm{x} < \mu+0.5\sigma;\\ 0,73-\dfrac{0,73}{0,4\sigma}\left(x-\mu-0,5\sigma\right) & \text{при } \mu+0.5\sigma \le \mathrm{x} < \mu+0.9\sigma;\\ 0 & \text{при } \mu+0.9\sigma \le \mathrm{x} < \mu+\sigma.\end{cases} \tag{2.9}$$

В (2.9) $\sigma = \lambda_2 - \lambda_1$.

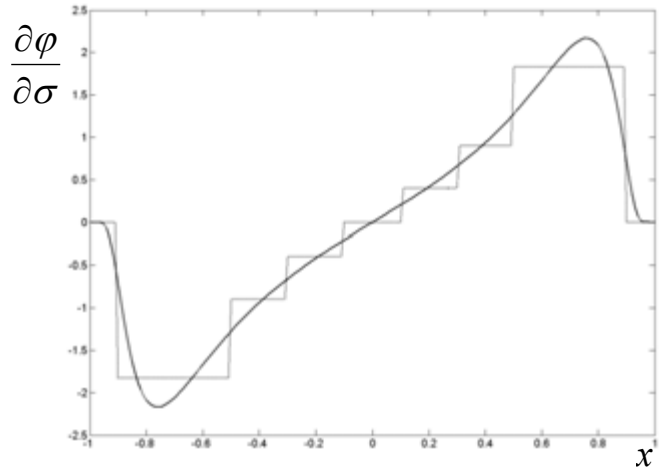
Аппроксимация НБФ осуществляется с использованием соотношений (2.8), (2.9), а также выражения (2.7).

На рис.2.5. сплошными линиями показаны: 2.5-а) – ГБФ (8) и ее производные по центрам 2.5-б), и радиусам 2.5-в), прерывистыми – соответствующие аппроксимации. Аналогично на рис. 2.6 сплошные линии отражают зависимости МГБФ (2.2) (рис. 2.6 -а)) и ее производных (рис. 2.6 -б) и 2.6 -в)), а прерывистые – соответствующие аппроксимации. На рис. 2.7 -а) и 2.7-б) сплошными линиями показаны виды двух и трех НБФ соответственно, а пунктирными линиями обозначены их аппроксимации.

В связи с большим количеством БФ, которые могут применяться в РБС, в ее хромосому вводится специальный ген BF, отвечающий за кодирование вида используемой функции. Так как в МП и РБС используются нелинейные функции преобразования с различным количеством параметров, то и кодирующие их хромосомы несколько отличаются [144-149].

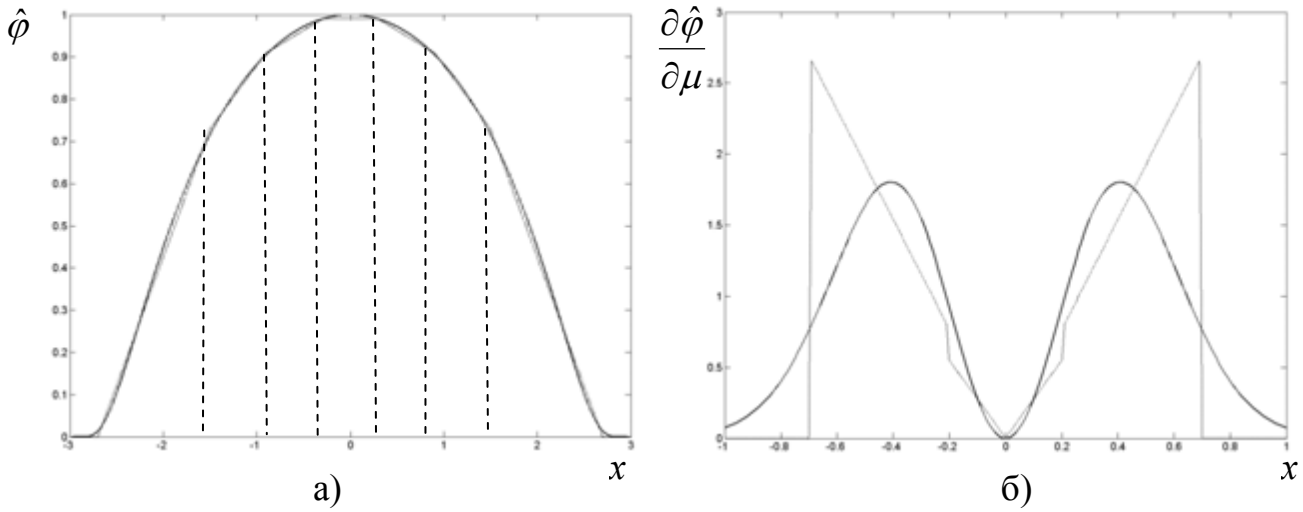


а) б)

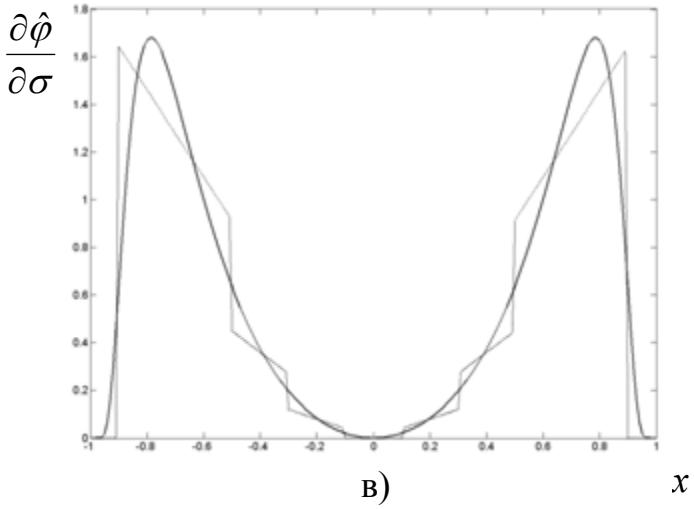


в)

Рисунок 2.5 – ГБФ и ее производные: б - по центрам; в – радиусам



а) б)



в)

Рисунок 2.6 – МГБФ и ее производные: б - по центрам; в - радиусам

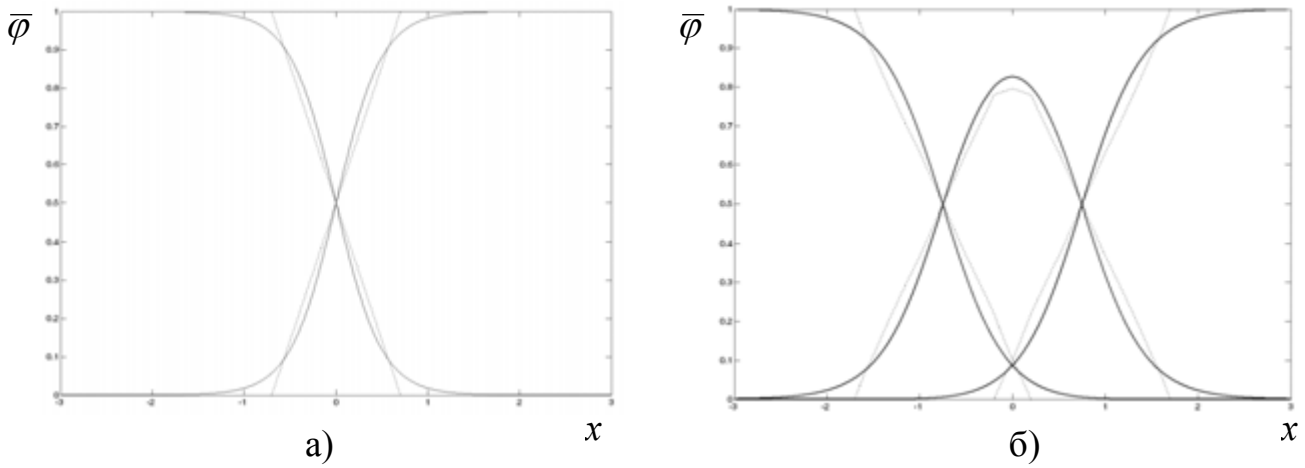


Рисунок 2.7 – Виды НБФ: а – две функции; б – три функции

Затем в хромосоме идет группа генов, кодирующая непосредственно весовые параметры соответствующего нейрона. На этапе инициализации всем этим параметрам с помощью датчика случайных чисел присваивают начальные значения, находящиеся в некотором допустимом диапазоне.

2.3.2 Оценка приспособленности.

После того как начальная популяция сформирована, производится оценка приспособленности каждой входящей в нее особи, определяемой некоторой функцией приспособленности (фитнесс-функцией). При идентификации параметров нелинейных систем с помощью ГА в них традиционно используется в качестве фитнесс-функции квадратичный функционал

$$\rho\left(e\left(i,\theta\right)\right)=0.5e^{2}\left(i,\theta\right). \tag{2.10}$$

При наличии же выбросов в обучающей выборке применение данной функции приспособленности приводит к тому, что решения, даваемые как алгоритмами обучения, использующими вычисления производных, так и ЭА, являются неудовлетворительными [150].

Выбор функции потерь, отличной от квадратичной, позволяет обеспечить робастность оценок, т.е. их работоспособность практически для всех распределений помех. Так для ε-засоренных вероятностных распределений, описываемых моделью Тьюки-Хьюбера [151]

$$p(x) = (1-\varepsilon)p_0(x) + \varepsilon q(x), \tag{2.11}$$

где $p_0(x)$ – плотность соответствующего основного распределения $N(0,\sigma_1^2)$; $q(x)$ – плотность засоряющего (произвольного) распределения $N(0,\sigma_2^2)$, $\sigma_1^2 << \sigma_2^2$; $\varepsilon \in [0,1]$ – параметр, характеризующий степень засорения основного распределения, на основе минимизации функционала

$$\rho(e(k)) = \begin{cases} \dfrac{e^2(k)}{\tau}, |e(k)| \le \tau; \\ \tau|e(k)| - \dfrac{e^2}{2}, |e(k)| > \tau; \end{cases} \tag{2.12}$$

были получены нелинейные оценки огрубленного или робастного метода максимального правдоподобия [151-153].

В настоящее время существует достаточно много функций $\rho(e(i))$, обеспечивающих получение робастных оценок, например [56, 144, 154-159],

$$\rho[e(k)] = |e(k)|^{\lambda}, \tag{2.13}$$

$$\rho[e(k)] = \ln\cosh(e(k)), \tag{2.14}$$

$$\rho[e(k)] = \sqrt{1+e^2(k)} - 1, \tag{2.15}$$

$$\rho[e(k)]=\begin{cases} 1-\cos\left(\dfrac{\pi e(k)}{\tau}\right), |e(k)|\le\tau; \\ 2, |e(k)|>\tau, \end{cases} \tag{2.16}$$

$$\rho[e(k)]=\tau^2\left(\frac{|e(k)|}{\tau}-\ln\left(1+\frac{|e(k)|}{\tau}\right)\right). \tag{2.17}$$

Необходимо отметить, что при использовании функционалов типа (2.12), (2.16)-(2.17) возникает проблема выбора (оценивания) *параметра* τ, который также может быть внесен в качестве дополнительного гена в хромосому и оценен с помощью ГА.

При использовании модели засорения (2.11) можно с помощью ГА настроить величины s_1^2 и s_2^2 (как дополнительные параметры сети, см. рис.2.3, 2.4), являющиеся оценками σ_1^2 и σ_2^2, и учесть эти оценки при вычислении значения фитнесс-функции, которая в этом случае будет иметь вид

$$f_i(x_j)=\frac{1}{M}\sum_{j=1}^{M}\hat{e}_j^2(k), \tag{2.18}$$

где $\hat{e}_j^2(k)=\begin{cases} \dfrac{\left(y_j^*(x_j)-\hat{y}_j(x_j)\right)^2}{s_1^2(k)}, \text{если } \left|y_j^*(x_j)-\hat{y}_j(x_j)\right|\le 3s_1(k); \\ \dfrac{\left(y_j^*(x_j)-\hat{y}_j(x_j)\right)^2}{s_2^2(k)}, \text{в противном случае.} \end{cases}$

2.3.3 Селекция.

После вычисления для каждой особи (сети) ее функции приспособленности в популяции необходимо провести отбор особей,

хромосомы которых будут принимать участие в формировании нового поколения [160]. Для этого производится вычисление среднего значения фитнесс-функции f_{av} популяции как среднее арифметическое значений фитнесс-функций всех входящих в нее особей:

$$f_{cp} = \frac{1}{N} \sum_{j=1}^{N} f_i \,. \tag{2.19}$$

Затем для каждой особи вычисляется следующее соотношение:

$$P_s(j) = \frac{f_i}{f_{av}},$$

и в зависимости от значения $P_s(j)$ формируется массив особей, хромосомы которых будут участвовать в процессе скрещивания.

Возможен также вариант селекции с использованием порогового значения функции приспособленности. При этом производится сортировка особей по убыванию нормированного значения функции приспособленности (2.18), задается порог $\theta \in [0,1]$, и для участия в процедуре скрещивания будут отобраны лишь те особи, для которых выполняется условие $f_i \le \theta$.

Операторы селекции

Свойства оператора селекции оказывают существенное влияние на среднее время получения решений заданного качества и на разброс результатов при независимых реализациях эволюционного процесса в ГА. В ряде случаев оператор пропорциональной селекции может быть заменен другими операторами. При этом выбор каждой родительской особи не обязательно осуществляется независимо от выбора других особей на данной итерации *k*.

Для сравнения различных операторов селекции необходимо выбрать некоторые характеристики, имеющие одинаковый смысл для всех из них. Возьмем в качестве таких характеристик математическое ожидание и дисперсию числа событий, состоящих в выборе особи i в качестве родительской в процессе построения очередного поколения. Число таких случайных событий будет обозначаться через $Z(i, \Pi^k)$ и формально определяется следующим образом. Пусть $K, \ldots, K_N$ - номера родительских особей, выбранных из П при построении очередной популяции в ГА. Тогда $Z(i, \Pi^k)$ – число таких j, что $K_j = i$.

Стохастическая универсальная селекция Бэкера. При построении очередной популяции оператор селекции ГА выбирает особь i в среднем $P_s(i, \Pi^k)$ N раз, однако, случайная величина $Z(i, \Pi^k)$ может существенно отклоняться от этого среднего значения. Большие отклонения от среднего могут приводить к неустойчивой работе ГА в целом. Для снижения таких колебаний Дж. Бэкер предложил следующую модификацию оператора селекции, называемую *стохастической универсальной селекцией* [13, 161]: каждая особь гарантированно выбирается $\lfloor P_s(i,\Pi^k)N \rfloor$ раз в качестве родителя, а дробная часть $\{P_s(i,\Pi^k)N\}$ используется, как вероятность отбора этой особи еще один раз. Очевидно, такой оператор, как и оператор селекции ГА, осуществляет пропорциональную селекцию.

Действие оператора стохастической универсальной селекции можно представлять себе с помощью колеса рулетки, разделенного на N секторов как при селекции в ГА. Пусть в центре колеса через равные угловые промежутки, закреплено N указателей. Поворот колеса рулетки на случайно выбранный угол (с равномерным распределением от 0 до 2π) относительно указателей дает назначение родительских особей $K_1, \ldots, K_N$. В завершение, элементы последовательности $K_1, \ldots, K_N$ переупорядочиваются случайным образом, чтобы снизить вероятность совпадения родительских генотипов на входе

следующего за селекцией оператора кроссинговера.

Исследования показали, что оператор стохастической универсальной селекции имеет преимущество по сравнению с классической селекцией, обеспечивая меньшие колебания численности потомства каждой особи и, тем самым, более устойчивую работу алгоритма.

Ранговая селекция. Оба из рассмотренных выше операторов селекции имеют общее проблемное свойство: как правило, в процессе развития популяции ГА происходит приближение приспособленности особей к максимальному значению, и вместе с этим распределение вероятностей $P_s(i,\Pi^k)$ приближается к равномерному. Таким образом, снижается «чувствительность» операторов пропорциональной селекции к различиям в приспособленности особей текущей популяции, что представляется нецелесообразным при решении задачи оптимизации. Один из способов решения этой проблемы был предложен Д. Голдбергом [13] и состоит в преобразовании значений функции приспособленности. Альтернативный подход состоит в использовании *ранжирования* особей.

Ранговая селекция состоит в применении оператора пропорциональной селекции с подстановкой рангов особей вместо значений их приспособленности. Такая селекция не теряет «чувствителвности» при сколь угодно малых различиях приспособленности особей.

Турнирная селекция. В операторе турнирной селекции [13] с размером турнира S (иное название: оператор S-турнирной селекции) для отбора очередного родительского решения из текущей популяции независимо извлекаются S особей с равномерным распределением и из них выбирается особь с наибольшим рангом.

Оператор S-турнирной селекции близок по своим свойствам к ранговой селекции.

Элитизм. Иногда хорошие решения могут быть потеряны при скрещивании или мутациях, приводящих к получению потомства, являющегося слабее, чем родителей. Обычно ГА вновь генерирует эти потерянные решения в последующих поколениях, но никаких гарантии восстановления потерянных особей с высокой приспособленностью не существует. Для борьбы с этим нежелательным эффектом можно использовать подход, известный как элитарность [21]. Элитарность включает в себя копирование небольшой группы сильнейший кандидатов без каких либо изменений в следующем поколении. Такой подход оказывает значительное влияние на производительность, гарантируя, что ГА не будет тратить время на повторное обнаружение ранее потерянных оптимальных решений. Особи, которые сохранились неизменными благодаря функции элитарности, сохраняют право на выбор их в качестве родительских особей при генерации следующего поколения. Эволюционная стратегия, применяющая функцию элитарности, приведена на рис.2.8. При такой стратегии особи с лучшими значениями функции копируются в новое поколение без каких-либо изменений, особи с более худшими показателями фитнесс-функции скрещиваются, но их потомки не мутируют, особи же с худшими значениями фитнесс-функций скрещиваются и хромосомах их потомков с некоторой вероятностью происходят мутации.

Рисунок 2.8 – Эволюционная стратегия

2.3.4 Скрещивание.

После того как родительские особи были отобраны методом селекции, осуществляется их скрещивание (кроссовер) [162]. Кроссовер применяется с целью воспроизведения потомства, и заключается в обмене генетической информацией между родительскими особями. Пусть родительские особи описываются выражениями

$$H^{(1)} = \left\{ h_1^{(1)} \ ,\ldots, h_i^{(1)}, .. h_L^{(1)} \right\};$$

$$H^{(2)} = \left\{ h_1^{(2)}, \ldots, h_i^{(2)} \ , .. h_L^{(2)} \right\},$$

а их потомки

$$Y^{(1)} = \left\{ y_1^{(1)} \ ,\ldots, y_i^{(1)}, .. y_L^{(1)} \right\};$$

$$Y^{(2)} = \left\{ y_1^{(2)} \ ,\ldots, y_i^{(2)}, .. y_L^{(2)} \right\}.$$

Количество родителей и потомков зависит от выбора оператора кроссовера и может меняться. Математически оператор кроссовера может быть записан как некоторый оператор (функция) ψ: $H \times H \rightarrow Y$, такой, что $\psi(H^{(1)}, H^{(2)}) \neq H^{(1)} \vee \psi(H^{(1)}, H^{(2)}) \neq H^{(2)}$, если $H^{(1)} \neq H^{(2)}$. Здесь предполагается, что для кодирования фенотипа в генотип используется порядково-зависимое вещественное кодирование. Как правило, оператор рекомбинации гарантирует, что потомки наследуют общие для обоих родителей аллеи. Формально это можно записать следующим образом:

$$H^{(1)}(i) = H^{(2)}(i) \Rightarrow Y(i) = H^{(1)}(i) = H^{(2)}(i),\ i = 1, 2, ..., n,$$

где $H^{(1)}$, $H^{(2)}$ и Y – родители и потомок соответственно.

Наследование оставшихся генов может быть осуществлено различными способами. В настоящее время существует достаточно большое количество различных операторов скрещивания: равномерное скрещивание, точечное скрещивание, однородное скрещивание, сравнительное скрещивание, нечеткое скрещивание, диагональное скрещивание и т.д.

В 1995 г. Тэйт и Смит [163] предложили модификацию равномерного кроссовера для хромосом с порядково-зависимым кодированием, который в работе [162] получил название *uniform like crossover* (ULX). Работает он следующим образом (рис 2.9). На первом этапе в хромосомах родителей определяются одинаковые значения генов, находящихся на одних и тех же позициях в их хромосомах. Значения этих генов копируются в те же позиции хромосомы потомка. Затем, на втором этапе, происходит сканирование хромосомы потомка слева направо и для еще неопределенных генов случайным образом с равной вероятностью выбирается либо значение гена с той же позиции либо первого, либо второго родителя, или происходит переход к следующей позиции (пропуск гена). Затем, на третьем этапе, все еще не определенные гены потомка заполняются случайными значениями в определенных диапазонах.

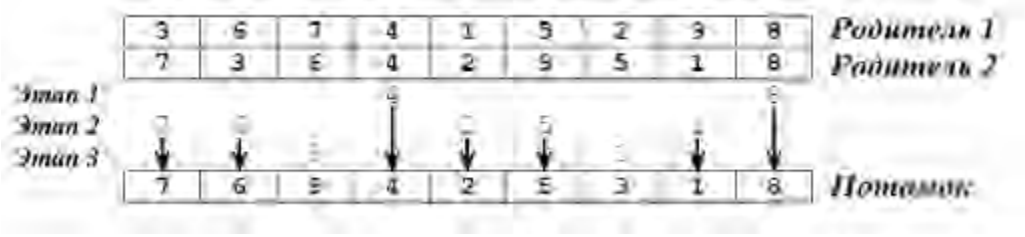


Рисунок 2.9 – Пример работы оператора скрещивания ULX

Равномерный кроссовер допускает некоторую гибкость в связи с чем возможны некоторые модификации базисной процедуры. Рассмотрим некоторые модификации (расширения) базисного ULX оператора. В некотором смысле эти модификации могут быть рассмотрены как отдельные типы оператора кроссовера. Назовем их: рандомизированый ULX (RULX), модифицированный ULX (или блоковый кроссовер (BX)), и расширенный ULX (или восстанавливающий кроссовер (RX)).

Единственная разница между описанным выше стандартным равномерным кроссовером и рандомизированным ULX заключается в порядке сканирования позиций хромосом. В стандартном ULX порядок сканирования является фиксированным - слева направо. В RULX анализ позиций (как уже определенных так и неопределенных) осуществляется случайным образом. Цель такого подхода заключается во внесении большей случайности и разнообразия в процесс генерации потомство (что является важным для избегания преждевременной сходимости).

Другая модификация ULX, названная в [162] блоковым кроссовером, отличается от ULX тем, что в процессе формирования хромосомы потомка принимают участия некоторые блоки генов (сегменты) хромосом родителей, вместо отдельных их элементов. Размер блока лежит в диапазоне [1, $\lfloor n/2 \rfloor$]. Следует отметить, что блоки могут быть различных размеров. Так, например, если $n = 9$, а число блоков равно 4, то получается три блока размером в 2 гена и один блок размером в 3 гена. При копировании блоков их позиции должны сохраняться (рис. 2.10). При размере блока, равном $\lfloor n/2 \rfloor$, первый сегмент хромосомы размером $\lfloor n/2 \rfloor$ копируется от одного из родителей (скажем, $H^{(1)}$) в хромосому потомка, а остальные элементы копируются из хромосомы "противоположного" родителя ($H^{(2)}$)) таким образом, что декодирование полученной хромосомы потомка остается возможным. Если после всех

копирований все еще остаются неопределенные гены в хромосоме потомка, то недостающие элементы могут быть случайным образом взяты либо у первого или второго родителя.

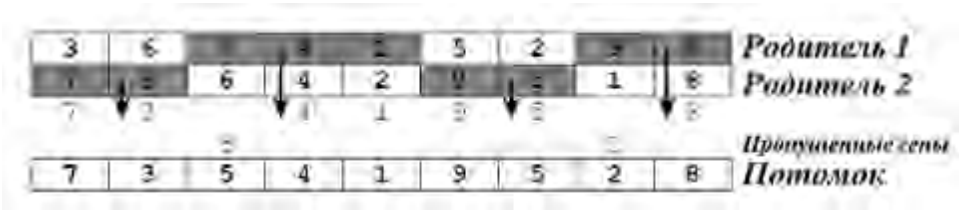


Рисунок 2.10 – Пример работы блокового кроссовера

Основная идея расширенного кроссовера (RX) является сочетание ULX и некоторой восстанавливающей процедуры (коррекции), которая применяется для хромосом потомства, полученного с помощью ULX. Данная процедура способна улучшить потомство с помощью попарной перестановки элементов - но только тех, которые не наследуются от родителей. Вначале создается список кандидатов *CL* следующим образом

$$CL(i) = Y(i),\ CL(i) \neq H^{(1)}(i),\ CL(i) \neq H^{(2)}(i), \tag{2.20}$$

где $H^{(1)}$, $H^{(2)}$ и Y – родители и потомок соответственно.

Затем кандидаты из списка принимают участие в процессе коррекции (если размер списка кандидатов нулевой ($|CL| = 0$), то процесс коррекции опускается).

Эксперименты показали, что размер списка кандидатов в большинстве случаев гораздо меньше n. Таким образом, локальная процедура коррекции вносит незначительный вклад в общее время выполнения ГА.

Далее рассмотрим вариант широко известного оператора кроссовера с частичным отображением (PMX), который был предложен в [164]. Основная

идея (PMX) заключается в том, что операции осуществляются над частью хромосомы – секцией отображения – расположенной между двумя точками кроссовера. Оператор PMX является весьма эффективным при решении задачи коммивояжера, однако при решении ряда других прикладных задач с данным оператором возникают существенные трудности. В связи с этим в [165] была предложена модификация данного оператора, использующая несколько случайных точек отображения вместо одного сегмента отображения. Данный кроссовер получил название равномерного PMX (UMPX). Оператор UMPX состоит из следующих шагов: 1) клонирование потомка Y из первого родителя $H^{(1)}$; 2) выбор случайным образом позиции pos_1 в хромосоме потомка; 3) поиск позиции pos_2 в хромосоме потомка в которой значении соответствующего гена равно значению гена с позицией pos_1 в хромосоме второго родителя $H^{(2)}$, т.е. $Y(pos_2) = H^{(2)}(pos_1)$; 4) обменять содержимое $Y(pos_1)$ и $Y(pos_2)$; 5) повторить шаги 1-5 k раз, где $k = \lfloor n/3 \rfloor$ (получено в [165]). Пример работы UPMX показан на рис. 2.11.

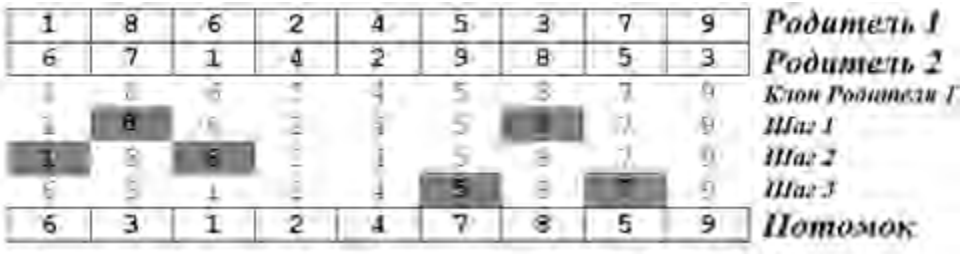


Рисунок 2.11 – Пример работы оператора UMPX

В [166] предложен кроссовер с сохранением дистанции (DPX) (более ранняя версия данного кроссовера для решения задачи коммивояжера была предложена в [167]. Основная идея DPX в том, что данный оператор скрещивания генерирует потомство, которое имеет такое же расстояние до каждого из своих родителей, и это расстояние равно расстоянию между самими родителями. Здесь под "расстоянием" ρ между двумя хромосомами $H^{(1)}$ и $H^{(2)}$

определяется как число различных элементов хромосом, находящихся на одинаковых позициях в хромосомах, т.е. $\rho\left(H^{(1)},H^{(2)}\right)=\left|\left\{i \mid H^{(1)}(i)\neq H^{(2)}(i)\right\}\right|$. Таким образом, элементы, которые идентичны и находятся на одних и тех же позициях в хромосомах обоих родителей ($H^{(1)}$ и $H^{(2)}$) будут скопированы в хромосому потомка *Y*. Значения же оставшихся генов хромосомы потомка будут присвоены случайным образом, с условием, что они не будут совпадать со значениями соответствующих ген ни одного из родителей. Пример работы DPX показан на рис.2.12.

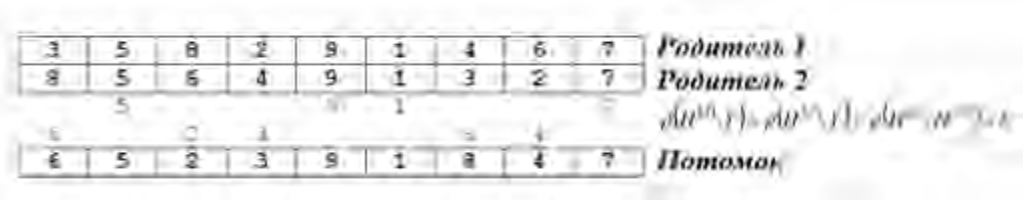


Рисунок 2.12 – Пример работы оператора DPX

Рассмотрим оператор циклического кроссовера (CX) [164, 168-169]. Особенность этого оператора заключается в том, что CX сохраняет информацию, содержащуюся в обоих родителях, т.е., все аллели потомства взяты либо от одного либо от второго родителя. Описать оператор CX можно следующим образом. Начиная с первой позиции, случайным образом выбирается ген одного из двух родителей и копируется в ту же позицию хромосомы потомка. Очевидно, что одинаковые гены будут при этом сохранены. Пример работы оператора CX представлен на рисунке 2.13

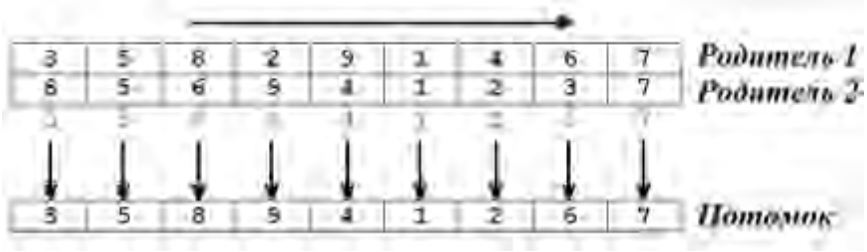


Рисунок 2.13 – Пример работы оператора CX

В работе [170] авторы для "жадного генетического алгоритма" предложили использовать оператор кроссовера с перестановкой путей (SPX). Пусть $H^{(1)}$ и $H^{(2)}$ являются парой родительских особей. Алгоритм SPX начинает работу с первого (или какого-то случайного) гена, и хромосомы родителей рассматриваются слева направо, до тех пор, пока все гены не будут рассмотрены. Если аллели в текущей рассматриваемой позицц обоих родителей одинаковые, то алгоритм переходит к следующей позиции; в противном случае, выполняется обмен (перестановка) двух аллелей в $H^{(1)}$ или в $H^{(2)}$, так что аллели в текущей позиции становятся подобными. (Например, если текущий ген i, $a = H^{(1)}(i)$, $b = H^{(2)}(i)$, затем, после обмена, либо $H^{(1)}(i)$ становится b, или $H^{(2)}(i)$ становится a.) Более подробно оператор SPX описан в [170].

Наиболее простым оператором является оператор одноточечного скрещивания (OPX) [13], при котором две родительские особи $H^{(1)}$ и $H^{(2)}$ формируют хромосомы двух потомков следующим образом

$$Y^{(1)} = \left\{ h_1^{(1)} \ ,\ldots, h_i^{(1)}, h_{i+1}^{(2)}, .. h_L^{(2)} \right\},$$

$$Y^{(2)} = \left\{ h_1^{(2)} \ ,\ldots, h_i^{(2)}, h_{i+1}^{(1)}, .. h_L^{(1)} \right\},$$

где i - случайная величина из интервала $[1, L]$.

Пример операции скрещивания двух РБС с помощью оператора OPX показан на рис.2.14-а),б). Из рисунков видно, что при скрещивании происходит не только обмен информацией о параметрах сети, но и образуются сети с новой для популяции структурой. Также из рисунков видно, что наличие интронов позволяет защитить целостность хромосом потомков и сохранить их логический формат и длину.

Так называемый порядковый оператор кроссовера (OBX) [171] был успешно применен для задач планирования, в которых целью является получить порядок (последовательность) в котором задания назначены планировщиком. Основной особенностью порядкового кроссовера является то, что он сохраняет относительный порядок аллелей в хромосомах. Таким образом, некоторое количество аллелей выбирается у одного из родителей и копируется в хромосому потомка. Пропущенные аллели копируются из следующих по порядку аллелей хромосомы другого родителя. Пример такого кроссовера показан на рис.2.15 (более подробная информация о данном операторе представлена в [171-172].

В работе [173] был предложен весьма оригинальный оператор кроссовера – так называемый связующий кроссовер (COHX), который генерирует потомство в несколько этапов. Вначале формируется маска $\boldsymbol{M}$ размерностью $n_1 \times n_2$. Где n_1 и n_2 выбираются таким образом, что $n_1 \cdot n_2 = n \wedge n_1 + n_2 \rightarrow \min$. Начальная позиция маски фиксируется в точке (i_0, j_0), где $i_0 \in \{1, 2, ..., n_1\}$, $j_0 \in \{1, 2, ..., n_2\}$. Затем маска заполняется в соответствии с принципом распространения волны. Таким образом получается n различных масок $\boldsymbol{M}^{(1)}$, $\boldsymbol{M}^{(2)}$, ..., $\boldsymbol{M}^{(k)}$, ..., $\boldsymbol{M}^{(n)}$, где k, i_0, and j_0 находятся в следующей зависимости: $k = n_2(i_0 - 1) + j_0$, $i_0 = 1, 2, ..., n_1$, $j_0 = 1, 2, ..., n_2$. Затем k-ый потомок $Y^{(k)}$ ($k \in \{1, 2,..., n\}$) генерируется с помощью следующих шагов:

$$1)\quad Y^{(k)}(i) = \begin{cases} H_{better}(i), M^{(k)}\left(\left((k-1)\operatorname{div} n_2\right)+1, \left((k-1)\bmod n_2\right)+1\right) \le \eta; \\ 0, \textit{в противном случае}; \end{cases}$$

а)

б)

Рисунок 2.14 – Примеры оператора одноточечного скрещивания

8	6	4	2	1	5	9	3	7
2	3	4	6	7	1	5	9	8
0	1	0	1	1	0	0	0	1
	6							7
		1			5	9	8	
3	6	4	2	1	5	9	8	7

Родитель 1
Родитель 2
Случайный шаблон
Элементы Родитель 1
Пропущенные элементы
Потомок

Рисунок 2.15 – Пример работы оператора OBX

где $i = 1, 2\ldots, n,\ H_{better} = \arg\min\left\{z(H^{(1)}), z(H^{(2)})\right\}$, $H^{(1)}$ и $H^{(2)}$ являются парой родительских особей, η - медиана матрицы $M^{(k)}$, т.е. $\eta = \dfrac{\sum_{i}^{n_1}\sum_{j}^{n_2} M^{(k)}(i,j)}{n_1 n_2}$;

$$2)\ \ Y^{(k)}(i) = \begin{cases} H^{(k)}(i), H^{(k)}(i) > 0; \\ H_{worse}(i), H^{(k)}(i) = 0 \wedge H_{worse}(i) \text{ не в } H^{(k)}; \\ 0, \textit{в противном случае}; \end{cases}$$

где $i = 1, 2\ldots, n,\ H_{worse} = \arg\max\left\{z(H^{(1)}), z(H^{(2)})\right\}$.

Для каждой неопределенной позиции i $\left(Y^{(k)}(i) = 0\right)$ значение выбирается случайным образом.

Следует отметить, что в результате работы данного оператора генерируется n различных решений, однако только лучшее из них рассматривается как потомок, т.е. $Y = \arg\min_{k=1,2,\ldots n} z\left(H^{(k)}\right)$.

Оператор кроссовера с множеством родителей (MPX) был предложен в работе [174]. MPX отличается от остальных операторов тем, что потомок наследует информацию от многих родителей. Это является главной отличительной характеристикой и главным преимуществом данного алгоритма над традиционными операторами в которых полезная информация может быть потеряна в следствие использования лишь двух родителей.

В MPX i-ый элемент (аллель потомка Y) получается путем выбора такого числа j (среди еще не выбранных значений), что вероятность $Y(i) = j$ является максимальной. Здесь вероятность того, что $Y(i) = j$, вычисляется следующим

образом: $\frac{d_{ij}}{\sum_{j}^{n} d_{ij}}$, где d_{ij} - компонента матрицы $\boldsymbol{D} = (d_{ij})_{n\times n}$, равная количеству раз, когда i-ый элемент был присвоен j-ой позиции $j = \pi(i)$ в хромосомах μ родителей (которые участвуют в процессе генерации потомка). Процесс повторяется до тех. Пор, пока не будут определены значения всех генов потомка.

Сравнительная характеристика всех рассмотренных выше операторов приведена в табл. 2.2.

Таблица 2.2 – Сравнительная характеристика операторов скрещивания

Кроссовер / **Характеристики**	**ULX, RULX, BX, OBX**	**RX**	**UPMX**	**DPX**	**CX**	**SPX**	**OPX**	**COHX**	**MPX**
Количество родителей	2	2	2	2	2	2	2	2	>2
Уровень рандомизации (явная мутация (ЯМ), скрытая мутация (СМ))	СМ	СМ	СМ	СМ	ЯМ	СМ	СМ	СМ	СМ
Сложность алгоритма	$O(n)$	$O(n + w^2)$	$O(n)$	$O(n^2)$	$O(n^2)$	$O(n^2)$	$O(n^2)$	$O(n^2)$	$O(n^2)$
Требование к памяти	$O(n)$	$O(n)$	$O(n)$	$O(n)$	$O(n)$	$O(n)$	$O(n)$	$O(n)$	$O(n^2)$
Сложность программной реализации (П – простой в реализации, С – сложный)	П	С	П	С	С	П	П	С	С
Априорная информация о системе (требуется (Т), не требуется (НТ))	НТ	Т	НТ	НТ	НТ	Т	НТ	Т	Т
Дополнительные свойства			$\rho(H^{(1)}, Y) \le 2k$, где k - число перестановок	Высокая степень изменения хромосомы потомка (не похож на родителей)		Низкая степень изменений	Относительно низкая степень изменений; $\rho(H^{(1)}, Y) < n - c$, где c позиция разрыва хромосомы	Относительно низкая степень изменений;	

2.3.5 Мутация.

После применения оператора кроссовера, хромосомы подвергаются мутации. Мутация предотвращает застревание алгоритма обучения в ловушке

локального минимума и позволяет создать новый генетический материал в популяции для поддержания ее разнообразия. Мутация - это не что иное, как случайное изменение части хромосомы, представляющей отдельную особь. Количество мутаций в популяции регулируется параметром p_m, который определяет вероятность мутации. Обычно вероятность выбирается равной $1/L$, где L – длина хромосомы. Таким образом, только $p_m \times L$ случайных хромосом в популяции могут мутировать. Вероятность мутации также может быть привязана к значению фитнесс-функции, т.е. чем хуже значение фитнесс-функции, тем выше вероятность мутации. Пример оператора мутации в хромосоме ЭРБС показан на рис.2.16.

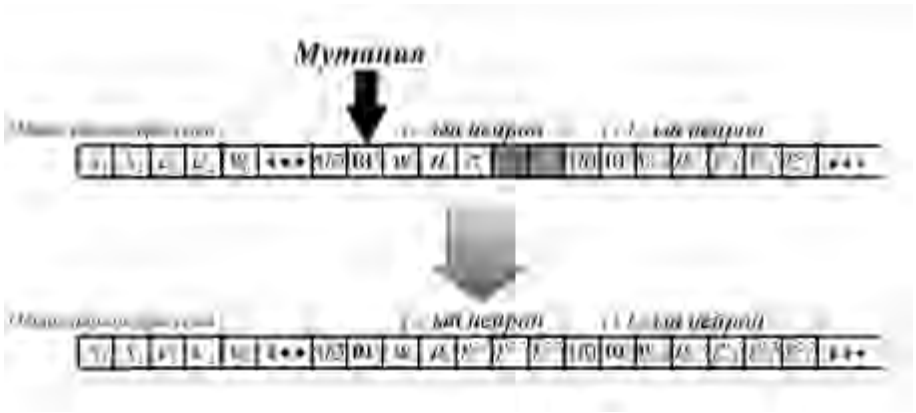


Рисунок 2.16 – Пример оператора мутации

Как видно из рисунка, в случае мутации гена, отвечающего за вид базисной функции нейрона, гены, кодирующие параметры БФ могут менять свою роль и переходить из экзонов (кодирующих областей хромосомы) в интроны (некодирующие области) и наоборот. Такой подход позволяет более эффективно настраивать структуру сети.

Оператор мутации осуществляет возможные мутации на некоторых генах некоторой хромосомы. Если хромосома перед тем как мутировать, имела вид

$$H_j = \{h_{1j},\ldots,h_{ij},..h_{Lj}\},$$

где h_{ij} - ген, в котором должна произойти мутация,

то после мутации ее можно записать следующим образом:

$$H'_j = \{h_{1j},\ldots,h'_{ij},..h_{Lj}\}.$$

Обычно мутировавшая хромосома сохраняется в популяции только в том случае, если мутация привела к улучшению приспособленности особи, что позволяет значительно ускорить поиск оптимального решения. Однако при таком подходе происходит уменьшение разнообразия в популяции, в результате чего алгоритм обычно сходится в некоторые локальные оптимумы.

Следует отметить тот факт, что с одной стороны мутации могут привести к ухудшению приспособленности данной особи, однако с другой – они являются единственным способом внесения новой информации в ее хромосому.

2.3.6 Операторы мутации и стратегии их применения.

Существует большое количество операторов мутации для различных способов кодирования хромосом. Поскольку в хромосоме ЭИНС используется гибридная кодировка, при мутации необходимо осуществлять различные операции для различных методов кодирования. Основные операторы мутаций и стратегии их применения подробно рассмотрены в Приложении А. Анализ проведен на основе работ [175-181].

2.3.7 Критерии останова.

После окончания формирования нового поколения осуществляется его оценка. В случае если выполняется критерий останова ГА, начинает работу

градиентный алгоритм настройки весовых параметров, который осуществляет «тонкую» настройку наилучшей сети, отобранной с помощью ГА. В качестве такого алгоритма может быть взята какая-либо модификация алгоритма ОР, алгоритм Гаусса-Ньютона, Левенберга-Марквардта и т.п.

Выводы по разделу 2

1. Основой построения и функционирования ЭИНС являются генетические алгоритмы (ГА), которые абстрагируют фундаментальные процессы дарвиновской эволюции: естественного отбора и генетических изменений вследствие рекомбинации и мутации. Однако помимо дарвиновских ГА могут реализовывать механизмы эволюции Ламарка, изменяющие (улучшающие) хромосомы, и Болдуина, улучшающие приспособляемость хромосомы без ее изменения.

2. В отличие от дарвинизма, где эволюция является результатом конкуренции и отбора, в ламаркизме сами организмы контролируют эволюцию. Это достигается с помощью практики, обучения и частого использования специальных органов. Хотя ламаркизм в природе не встречается, его идеи нашли применение при построении ЭИНС в алгоритмах, содержащих механизм сохранения приобретенных характеристик.

3. Анализ работ по применению эффекта Болдуина в ЭИНС показал, что обучение Болдуина можно рассматривать как своего рода изменчивость, направляемую фенотипом, позволяющую увеличить дисперсию процесса отбора и делающую рельеф фитнес-функции вокруг оптимальных областей более плоским, что обеспечивает улучшение процесса эволюции. Общий эффект от обучения Болдуина заключается в том, что оно позволяет найти глобальный оптимум. Однако в некоторых случаях эффект Болдуина может приводить к неэффективным гибридным алгоритмам, в то время как

использование эволюции Ламарка является весьма эффективным в нестационарных условиях функционирования сети.

4. Анализ работ по построению ЭИНС показал, что эволюция в ЭИНС может касаться архитектуры сети, ее весов, вида и параметров АФ (БФ), алгоритма обучения. В связи с этим получила дальнейшее развитие архитектура эволюционирующей ИНС прямого распространения, которая помимо эволюции структуры сети и ее параметров учитывает также эволюцию алгоритма обучения и помехи. Такая ИНС обладает существенно улучшенными аппроксимирующими и экстраполирующими свойствами, что дает возможность за счет адаптации структуры, параметров и процедуры обучения эффективно обрабатывать информацию в нестационарных условиях и наличии неопределенности.

5. При переходе от ИНС к ЭИНС для всех типов сетей используются общие эволюционные процедуры (инициализация популяции, оценка популяции, селекция, скрещивание, мутации), а различия заключаются лишь в способе кодирования структуры и параметров той или иной ИНС в виде хромосомы. Приведена схема работы такого эволюционного алгоритма настройки ИНС и предложены различны форматы хромосомы для МП и РБС.

6. При использовании ГА для построения ЭИНС основными проблемами являются выбор метода кодирования возможного решения и генетических операторов; в различных задачах используются различные методы кодирования и генетические операторы, основные из которых рассмотрены в разделе.

7. Предложены новые методы аппроксимации гауссовских базисных функций в РБС нулевого и первого порядков. Использование кусочно-линейной аппроксимации позволяет существенно упростить вычисления, сопутствующие процессам построения нейросетевой модели исследуемых объектов и управления ими.

РАЗДЕЛ 3
МЕТОДЫ ОБУЧЕНИЯ ЭВОЛЮЦИОНИРУЮЩИХ НЕЙРОННЫХ СЕТЕЙ ПРЯМОГО РАСПРОСТРАНЕНИЯ

В настоящее время существует большое число методов настройки параметров сети, отличающихся объемом используемой информации, влияющим как на динамические свойства алгоритмов, так и на их вычислительную сложность.

Как отмечалось выше, задача обучения сетей сводится к минимизации функционала (1.54).

При таком подходе качество обучения зависит от того, насколько удачно была выбрана функция $\rho(e(i),\theta)$. Одним из основных факторов, влияющих на эффективность выбора $\rho(e(i),\theta)$, является наличие информации о свойствах помехи ξ. В настоящее время существует два принципиально различных подхода, основанных на том, что

1) существует информация либо о некоторых статистических свойствах помехи, либо о принадлежности ее распределения помехи некоторому известному классу;
2) несмотря на природу помехи, она предполагается ограниченной.

В первом случае для оценивания (обучения) используются статистические методы.

Если помеха имеет нормальное распределение, эти методы имеют аналитическое решение, а получаемые путем минимизации квадратичного функционала, т.е. функционала (1.54) с $\rho(i,\theta)=e^2(i,\theta)$, оценки являются оптимальными (так для линейного случая МНК-оценка оптимальна и обладает минимальной дисперсией в классе несмещенных оценок).

Если же такой информации нет, используют другой подход, при котором задача обучения заключается в поиске θ, принадлежащего некоторому классу моделей, для которых абсолютное значение $\left|f(x)-\hat{f}(x,\theta)\right|\le\delta$ для всех моментов времени *k*. В этом случае алгоритмы обучения, которые также имеют структуру, близкую к структуре алгоритма рекуррентного МНК, будут содержать зону нечувствительности, определяемую величиной δ. Следует отметить, что эта зона нечувствительности может служить величиной, ограничивающей точность получаемого решения, т.е. определяющей его допустимую погрешность.

3.1 Одношаговые процедуры обучения

3.1.1 Одношаговые процедуры обучения при наличии информации о статистических свойствах помех.

Одношаговые процедуры обучения минимизируют не функционал (1.54), а функцию потерь $\rho(e(i,\theta))$ на одном такте.

а) Обучение линейной Адалины.

Если в качестве $\rho(e(i,\theta))$ выбрать квадратичную функцию $E(e(k,\theta))=e^2(k)$, то оптимальная одношаговая процедура обучения будет иметь вид

$$\theta(k)=\theta(k-1)+\gamma\frac{e(k)}{\left\|x(k)\right\|^2}x(k), \qquad (3.1)$$

где $\gamma\in(0,2)$ – коэффициент обучения.

Данная процедура представляет собой алгоритм Качмажа [92], известный в теории ИНС как алгоритм Уидроу-Хоффа [93]. В теории же оценивания и

фильтрации (3.1) называют нормализованным алгоритмом наименьших квадратов.

Начиная с работы Чадеева В.М. [182], исследованию свойств этой процедуры посвящено огромное число работ (см., например [183]).

Традиционно схема доказательства сходимости сводится к анализу каким-либо образом выбранной функции Ляпунова. В наиболее простом случае она представляет собой разность ошибок обучения на двух соседних тактах, т.е.

$$V(k) = \left\|\tilde{\theta}(k)\right\|^2 - \left\|\tilde{\theta}(k-1)\right\|^2, \tag{3.2}$$

либо

$$V(k) = M\left\{\left\|\tilde{\theta}(k)\right\|^2\right\} - M\left\{\left\|\tilde{\theta}(k-1)\right\|^2\right\}, \tag{3.3}$$

а сходимость обеспечивается при

$$\Delta V(k) = V(k) - V(k-1) \le 0. \tag{3.4}$$

В этом случае процедура обучения записывается относительно ошибок обучения $\tilde{\theta}(i) = \theta^* - \theta(i)$ и производится ее анализ.

Поступая таким образом, несложно показать, что процедура (3.1) сходится в среднем при

$$0 < \gamma < M\left\{\frac{x(k)x^T(k)}{\left\|x(k)\right\|^2}\right\} = 1,$$

а в среднеквадратичном – при

$$0 < \gamma < \frac{2}{\|\sigma_x\|^2}.$$

Оптимальное же значение γ^{onm}, обеспечивающее максимальную скорость сходимости, будет равно 1.

При этом

$$\lim_{k\to\infty} M\left\{\left\|\tilde{\theta}(k)\right\|^2\right\} = \frac{\gamma^2\sigma_\xi^2}{N\sigma_x^2} \neq 0, \tag{3.5}$$

т.е. для сходимости в точку параметр γ должен удовлетворять условиям Дворецкого [184].

Следует отметить, что определение условий сходимости процедуры (3.1) и выбор оптимального шага обучения γ^{onm} зависят от специфики решаемой задачи (идентификация или фильтрация, стационарные или нестационарные условия, наличие или отсутствие помех, и т.д.). Практически во всех работах исследуются вопросы сходимости в среднем и среднеквадратичном. Например в [185] получены условия сходимости при отсутствии помех

$$0 < \gamma < \frac{1}{\lambda_{\max}}, \tag{3.6}$$

в [186]

$$0 < \gamma < \frac{2}{\mathrm{tr}[R_{xx}]}; \tag{3.7}$$

в [187]

$$0 < \gamma < \frac{1}{3\,\mathrm{tr}\left[R_{xx}\right]}; \tag{3.8}$$

а, например, в [188] – при наличии помех

$$\frac{2^{-b}}{4\sigma_x\sqrt{\sigma_\xi^2+\sigma_e^2}} < \gamma < \frac{1}{\mathrm{tr}\left[R_{xx}\right]}, \tag{3.9}$$

где b – количество бит в цифровом представлении сигнала.

В различное время были предложены и исследованы многочисленные модификации (3.1), среди которых наибольший интерес представляют его регуляризированный вариант [182], получающийся путем использования в знаменателе правой части (3.1) регуляризатора $\beta > 0$,

$$\theta(k) = \theta(k-1) + \gamma(k)\frac{e(k)}{\left\|x(k)\right\|^2 + \beta}x(k) \tag{3.10}$$

и алгоритм Гудвина-Рэмеджа-Кейнеса [189]

$$\theta(k) = \theta(k-1) + \gamma(k)\frac{e(k)}{r(k)}x(k), \tag{3.11}$$

где $r(k) = r(k-1) + \left\|x(k)\right\|^2$; $r(0) = r_0 = 1$, сочетающий свойства нормализованного алгоритма (3.1) и алгоритма стохастической аппроксимации.

Регуляризованная процедура (3.10) изучалась во многих работах, например, в работе [190], помимо условий сходимости, была определена и ее скорость, которая при $\gamma = 1$ определяется следующим образом:

$$M\left\{\tilde{\theta}(k)\right\}=\left(1-\frac{1}{N}\left(1-\frac{\beta N}{(N-2)\sigma_x^2+\beta}\right)\left(1-\frac{\beta\sigma_x^2}{\left((N-2)\sigma_x^2+\beta\right)^2}\right)\right)\tilde{\theta}(0). \qquad (3.12)$$

Кроме того, легко можно показать, что для данной регуляризованной процедуры справедливо

$$M\left\{\left\|\tilde{\theta}(k)\right\|^2\right\}=\left(1-\gamma\frac{2\sigma_x^2}{N\sigma_x^2+\beta}+\gamma^2\frac{\sigma_x^4}{\left(N\sigma_x^2+\beta\right)^2}\right)M\left\{\left\|\tilde{\theta}(k-1)\right\|^2\right\}+\gamma^2\frac{N\sigma_x^2\sigma_\xi^2}{\left(N\sigma_x^2+\beta\right)^2}, \qquad (3.13)$$

т.е. процедура сходится в среднеквадратичном при

$$\left|1-\gamma\frac{2\sigma_x^2}{N\sigma_x^2+\beta}+\gamma^2\frac{(\mathrm{N}+2)\sigma_x^4}{\left(N\sigma_x^2+\beta\right)^2}\right|<1$$

или

$$0<\gamma<\frac{2(N\sigma_x^2+\beta)}{(N+2)\sigma_x^2} \qquad (3.14)$$

к области, определяемой помехой

$$\lim_{k\to\infty}M\left\{\left\|\theta(k)\right\|^2\right\}=\gamma^2\frac{N\sigma_x^2\sigma_\xi^2}{\left(N\sigma_x^2+\beta\right)^2}. \qquad (3.15)$$

Из последнего выражения следует, что для сходимости в точку параметр γ должен удовлетворять условиям Дворецкого.

Модификацией (3.1) является предложенная в [191] процедура

$$w(k) = w(k-1) + \gamma \frac{G(k-1)x(k)e(k)}{x^T(k)G(k-1)x(k)}, \tag{3.16}$$

где $G(k) = diag\left[g_0(k-1) \quad g_1(k-1) \quad ... \quad g_{N-1}(k-1)\right]$ в – матричный коэффициент усиления *N*x*N*;

$$g_i(k-1) = \frac{1-k}{2N} + (1-k)\frac{|w_i(k-1)|}{2|w(k-1)|}, \quad 0 \le i < N-1, \tag{3.17}$$

$k \in [-1,1)$.

Так как можно показать, что

$$x^T(k)G(k-1)x(k) = \frac{1-k}{2N}\|x(k)\|^2 + \frac{1+k\sum_{i=0}^{N-1} x^2(k-i)|w_i(k-1)|}{2|w(k-1)|} \approx$$

$$\approx \frac{1-k}{2}\sigma_x^2 + \frac{1+k}{2}\sigma_x^2 \approx \sigma_x^2,$$

процедура (3.16) мало чем отличается от (3.1).

Объединить преимущества процедур Качмажа и стохастической аппроксимации удается в (3.11). Данная процедура изучалась в [189].

Выше отмечалось, что алгоритм (3.1) получен путем минимизации квадратичного функционала. Выбор минимизируемого функционала другого вида приводит к получению алгоритма, отличного от алгоритмов, минимизирующих квадратичный функционал (типа МНК), и, следовательно,

обладающего другими свойствами. Как показывает практика, весьма перспективным является применение оценок (алгоритмов) метода наименьших модулей (МНМ), получающихся в результате минимизации модульного критерия. Если эффективность МНМ при гауссовых распределениях помехи ниже, чем МНК, то при распределениях помех с более тяжелыми хвостами (например, Лапласа или Коши) эффективность МНМ существенно превышает эффективность МНК. Таким образом, алгоритмы МНМ обладают лучшим по сравнению с алгоритмами МНК робастными свойствами. Улучшение робастных свойств алгоритмов (3.1), (3.10) может быть достигнуто введением в данные алгоритмы нелинейности. Кроме того, использование в них нелинейности типа sign(x) должно уменьшить количество вычислений, необходимых для реализации данного алгоритма, так как при вычислении скалярных произведений типа $x^T \operatorname{sign}(x)$ исчезает операция умножения.

К числу таких алгоритмов относится алгоритм Нагумо-Ноды [192]

$$\theta(k) = \theta(k-1) + \gamma(k)\frac{e(k)}{|x(k)|}\operatorname{sign} x(k), \tag{3.18}$$

где $|x(k)| = x^T(k)\operatorname{sign} x(k)$.

Регуляризованный алгоритм Нагумо-Ноды

$$\theta(k) = \theta(k-1) + \gamma(k)\frac{e(k)}{|x(k)| + \delta}\operatorname{sign} x(k), \tag{3.19}$$

был получен и исследован в [190].

Процедуры (3.18) и (3.19) изучались в работах [192, 193]. Так в [192] для случая $\gamma = 1$ было показано, что

$$M\left\{\tilde{\theta}(k)\right\}=\left(1-\frac{1}{N}\left(1-\frac{\beta\tilde{\alpha}N}{N+\beta\tilde{\alpha}}\right)\right)M\left\{\tilde{\theta}(k-1)\right\}+O\left(\frac{1}{N^2}\right), \tag{3.20}$$

$$M\left\{\left\|\tilde{\theta}(k)\right\|^2\right\}=\left(1-\frac{1}{N+\beta\tilde{\alpha}}\left(2-\frac{\pi N}{2\left(N+\beta\tilde{\alpha}\right)}\right)\right)M\left\{\left\|\tilde{\theta}(k-1)\right\|^2\right\}+\frac{\tilde{\alpha}^2N\sigma_\xi^2}{\left(N+\beta\tilde{\alpha}\right)^2}+O\left(\frac{1}{N^2}\right), \tag{3.21}$$

где $\tilde{\alpha}=\sqrt{\dfrac{\pi}{2\sigma_x^2}}$.

Таким образом, данные процедуры, как и рассмотренные выше, обеспечивают получение асимптотически несмещенных оценок. Область же сходимости определяется выражением

$$\lim_{k\to\infty}M\left\{\left\|\tilde{\theta}(k)\right\|^2\right\}=\frac{\pi N}{4\left(N+\beta\tilde{\alpha}\right)-\pi N}\frac{\sigma_\xi^2}{\sigma_x^2}\neq 0. \tag{3.22}$$

Из последнего выражения следует, что для обеспечения сходимости в точку следует брать в (3.18), (3.19) $\gamma\neq 1$, а переменное γ, удовлетворяющее условиям Дворецкого.

В [194] предложена процедура обучения, основанная на (3.16) и имеющая вид

$$\theta(k)=\theta(k-1)+\gamma\frac{G(k-1)\operatorname{sign}x(k)e(k)}{\beta+x^T(k)G(k-1)\operatorname{sign}x(k)}, \tag{3.23}$$

где матрица $G(k-1)$ имеет такой же вид, как и в (3.16).

По аналогии с (3.11) может быть предложена следующая модификация алгоритма (3.18):

$$\theta(k) = \theta(k-1) + \gamma(k)\frac{e(k)}{\tilde{r}(k)}\operatorname{sign}\nabla x(k), \tag{3.24}$$

где $\tilde{r}(k) = \tilde{r}(k-1) + \nabla x^T(k)\operatorname{sign}\nabla x(k)$; $\tilde{r}(0) = r_0 > 0$.

б) Обучение нелинейной Адалины

Следует отметить, что при нейросетевой аппроксимации нелинейности $y(k) = f(x(k),\theta)$ или $y(k) = f(\theta^T x(k))$ и выборе квадратичного функционала обучения $E(k) = 0.5e^2(k)$, градиентная процедура обучения будет иметь вид

$$\theta(k) = \theta(k-1) + \Delta\gamma\Delta\theta(k), \tag{3.25}$$

где

$$\Delta\theta(k) = \gamma e(k)\Delta f(x(k),\theta(k-1))x(k) \tag{3.26}$$

Для простоты вместо $f(\theta^T x(k))$ будем писать $f(k)$.

Разложение ошибки в ряд Тейлора

$$\begin{aligned} e(k) &= e(k-1) + \sum_{i=1}^{N}\frac{\partial e(k-1)}{\partial\theta_i(k-1)}\Delta\theta_i(k) + \\ &+ \sum_{i=1}^{N}\frac{\partial e(k-1)}{\partial x_i(k-1)}\Delta x_i(k) + \frac{\partial e(k-1)}{\partial y^*(k)}\Delta y^*(k) + \ldots \end{aligned} \tag{3.27}$$

и ограничение членами разложения

$$e(k) = e(k-1) + \sum_{i=1}^{N} \frac{\partial e(k-1)}{\partial \theta_i(k-1)} \Delta\theta_i(k) + \ \varepsilon,$$

где ε – остальные члены разложения, позволяет записать с учетом (3.26) следующее соотношение для $e(k)$:

$$\begin{aligned} e(k) &\approx e(k-1) - e(k-1)\gamma \sum_{i=1}^{N} x_i(k)\nabla f(k)\nabla f(k)x_i(k) = \\ &= \left(1 - \gamma \left\|\nabla f(k)x(k)\right\|^2\right) e(k). \end{aligned} \tag{3.28}$$

Рассмотрим величину

$$\left|e(k)\right| = \left|1 - \gamma \left\|\nabla f(k)x(k)\right\|^2\right| \left|e(k-1)\right|. \tag{3.29}$$

Из (3.29) вытекает, что процедура будет сходиться, если

$$\left|1 - \gamma \left\|\nabla f(k)x(k)\right\|^2\right| < 1, \tag{3.30}$$

т.е. при

$$0 < \gamma < \frac{2}{\left\|\nabla f(k)x(k)\right\|^2}, \tag{3.31}$$

а оптимальное значение параметра γ, обеспечивающее максимальную скорость сходимости, определяется выражением

$$\gamma(k)=\frac{1}{\left\|\nabla f(k)x(k)\right\|^2}. \qquad (3.32)$$

Таким образом, данная процедура соответствует процедуре Качмажа (Уидроу-Хоффа) для нелинейного случая.

Повышение вычислительной устойчивости (3.26) можно добиться путем использования регуляризующего параметра $\beta>0$, т.е.

$$\gamma(k)=\frac{1}{\left\|\nabla f(k)x(k)\right\|^2+\beta}. \qquad (3.33)$$

Ограничения, налагаемые на величину β, и рекуррентная процедура настройки β рассмотрены в следующем подразделе.

в) Обучение модели персептронного типа.

Рассмотрим обучение многослойного персептрона (МП) с *M* входами и *L* выходами, числом нейронов выходного слоя *P*, скрытого – *N*, все нейроны которого имеют активационную функцию вида $f(x)=\frac{1}{1+e^{-\alpha x}}$.

Введем следующие обозначения:

$V^*(k)=\left(V_1^*(k),V_2^*(k),...,V_p^*(k)\right)^T$ – матрица оптимальных весов выходного слоя *P*x*L*;

$V_i^*(k)=\left(V_{i1}^*(k),V_{i2}^*(k),...,V_{iL}^*(k)\right)^T$ – вектор оптимальных весов *i*-го нейрона выходного слоя *P*x1;

$V(k)=\left(V_1(k),V_2(k),...,V_p(k)\right)^T$ – матрица оценок весов выходного слоя МП *P*x*L*;

$V_i\ (k) = \left(V_{i1}(k), V_{i2}(k), ..., V_{iL}(k)\right)^T$ – вектор оценок весов *i*-го нейрона выходного слоя *P*x1;

$W^*(k) = \left(W_1^*(k), W_2^*(k), ..., W_p^*(k)\right)^T$ – матрица оптимальных весов нейронов скрытого слоя *N*x*P*;

$W_i^*(k) = \left(W_{i1}^*(k), W_{i2}^*(k), ..., W_{iN}^*(k)\right)^T$ – вектор оптимальных весов нейронов скрытого слоя *N*x1;

$F(x,w) = \left(f_1(w_1^T(k)x(k)) \quad f_2(w_2^T(k)x(k)) \quad ... \quad f_L(w_L^T(k)x(k))\right)^T$ – вектор активационных функций нейронов выходного слоя *L*x1;

$\nabla f(x,w) = diag\left[\nabla f(w_1^T(k), x(k)) \quad \nabla f(w_2^T(k), x(k)) \quad ... \quad \nabla f(w_L^T(k), x(k))\right]$ – матрица производных активационных функций *L*x*L*;

$\nabla f(w_i^T(k), x(k))$ – первая производная активационной функции *i*-го нейрона.

Тогда выходные сигналы МП и ошибка аппроксимации могут быть представлены следующим образом:

- требуемый вектор выходных сигналов

$$y^*(k) = V^*(k)F\left(x, w^*(k)\right) + \xi_1(k); \tag{3.34}$$

- реальный вектор выходных сигналов

$$y(k) = V(k)F\left(x, w(k)\right) + \xi_2(k); \tag{3.35}$$

- вектор ошибок реакции сети

$$e(k) = y^*(k) - y(k) + \xi(k) = \tilde{\theta}^T(\mathrm{k})F\left(x, w(k)\right) + V^*(k)\tilde{F}\left(x, w(k)\right) + \xi_1(k) + \xi_2(k); \quad (3.36)$$

где $\tilde{\theta}(k) = V^*(k) - V(k)$ – матрица ошибок обучения PxL;

$\tilde{F}(k) = F\left(x, w^*(k)\right) - F\left(x, w(k)\right)$ – вектор ошибок активационных функций;

$\xi_1(k)$, $\xi_2(k)$ – вектор ошибок выходного сигнала и нейросетевой аппроксимации соответственно.

С учетом этих обозначений градиентные процедуры обучения матриц весов выходного и скрытого слоя МП, минимизирующие квадратичный функционал и представляющие собой процедуру обратного распространения ошибки, могут быть записаны следующим образом:

$$V(k) = V(k-1) - \gamma_V(k)\frac{1}{\left\|F\left(x, w(k-1)\right)\right\|^2}\frac{\partial E(k)}{\partial V(k-1)} = V(k-1) + \gamma_V(k)\frac{F\left(x, w(k-1)\right)e^T(k)}{\left\|F\left(x, w(k-1)\right)\right\|^2}; \quad (3.37)$$

$$W(k) = W(k-1) - \gamma_W(k)\frac{1}{\left\|\nabla f\left(x(k), V(k-1)\right)x(k)\right\|^2}\frac{\partial E(k)}{\partial W(k-1)} = W(k-1) + \gamma_W(k)\frac{\nabla f\left(x(k), w(k-1)\right)e^T(k)x^T(k)}{\left\|\nabla f\left(x(k), V(k-1)\right)x(k)\right\|^2}; \quad (3.38)$$

т.е. являются матричными вариантами процедуры Качмажа (Уидроу-Хоффа).

Здесь $\gamma_V(k)$ и $\gamma_W(k)$ – коэффициенты обучения.

Исследование вопросов сходимости данных процедур трудностей не вызывает и осуществляется стандартными способами.

Для повышения вычислительной устойчивости (3.37), (3.38) они могут быть модифицированы аналогично (3.10), т.е.

$$V(k) = \mathrm{V}(k-1) + \gamma_V(k)\frac{F\left(x, w(k-1)\right)e(k)}{\beta_V(k) + \left\|F\left(x, w(k-1)\right)\right\|^2}; \tag{3.39}$$

$$W(k) = W(k-1) + \gamma_W(k)\frac{\nabla f\left(x(k), w(k-1)\right)e(k)x^T(k)}{\beta_W(k) + \left\|\nabla f\left(x, V(k-1)\right)x(k)\right\|^2}, \tag{3.40}$$

где $\beta_V(k)$, $\beta_W(k)$ – параметры регуляризации.

Условия сходимости данных процедур, ограничения, которым должны удовлетворять параметры $\gamma_i(k)$, $\beta_i(k)$ $\left(i = V, W\right)$, определяются способом, аналогичным рассмотренному выше.

3.1.2 Одношаговые процедуры обучения при наличии ограниченных помех.

Если относительно помехи известно только, что она ограничена по амплитуде

$$\left|\xi(k)\right| < \Delta, \tag{3.41}$$

то для оценивания параметров применяются иные методы, которые не используют никакой информации о статистических свойствах возмущений кроме их принадлежности некоторому ограниченному интервалу.

В настоящее время для оценивания параметров при наличии ограниченных по амплитуде помех наиболее широкое распространение получили алгоритмы, в основе построения которых лежит метод политопов (и как частный случай – ортотопов), алгоритмы, содержащие зону нечувствительности и алгоритмы на основе построения эллипсоидов.

Если о помехах известно, что они удовлетворяют условию (3.41), то эта информация может быть учтена в алгоритме, содержащем зону нечувствительности. При этом снижаются требования относительно знания свойств помехи, однако и огрубляется сам алгоритм обучения.

Целесообразность использования в алгоритме зоны нечувствительности основана на том, что даже в случае точного определения параметров модели остается ошибка (рассогласование между выходными сигналами объекта и модели), величина которой определяется величиной помехи (3.41).

Различные алгоритмы, включающие нелинейность типа зоны нечувствительности

$$\xi(\varphi)=\begin{cases}\varphi-\Delta & при \quad \varphi>\Delta;\\ 0 & при \quad |\varphi|\le\Delta;\\ \varphi+\Delta & при \quad \varphi<-\Delta,\end{cases}$$

рассматривались в работах [195-197].

Наиболее часто используются зоны нечувствительности, описываемые соотношениями

$$g(e(k),\Delta)=\begin{cases}e(k), & если \quad |e(k)|\ge\Delta;\\ 0, & если \quad |e(k)|<\Delta.\end{cases} \qquad (3.42)$$

и

$$g(e(k),\Delta)=\begin{cases} e(k)-\Delta(k)signe(k), & если \quad |e(k)|\geq\Delta; \\ 0, & если \quad |e(k)|<\Delta. \end{cases} \tag{3.43}$$

Рассмотрим некоторые процедуры, содержащие зону нечувствительности. Пусть задана градиентная процедура вида

$$\theta(k)=\theta(k-1)+\gamma g(e(k),\Delta)x(k), \tag{3.44}$$

где $\gamma>0$ – параметр обучения.

Поступая стандартно, т.е. вычитая из обеих частей (3.44) w^*, записывая процедуру относительно ошибок обучения и возводя обе части полученного выражения в квадрат, имеем

$$\left\|\tilde{\theta}(k)\right\|^2=\left\|\tilde{\theta}(k-1)\right\|^2-2\gamma g(e(k),\Delta)\tilde{\theta}^T(k-1)x(k)+\gamma^2 g^2(e(k),\Delta)\left\|\mathrm{x}(k)\right\|^2. \tag{3.45}$$

Тогда приращение функции Ляпунова

$$\Delta V(k)=-2\gamma g(e(k),\Delta)\left(\tilde{\theta}^T(k-1)x(k)\right)+\gamma^2 g^2(e(k),\Delta)\left\|x(k)\right\|^2 \tag{3.46}$$

будет отрицательным при

$$g(e(k),\Delta)\leq\frac{2e(k)}{\gamma\left\|x(k-1)\right\|^2}. \tag{3.47}$$

Принимая во внимание, что

$$g(e(k),\Delta) = \left|g(e(k),\Delta)\right| \operatorname{sign} g(e(k),\Delta),$$

а

$$\operatorname{sign} g(e(k),\Delta) = \operatorname{sign} e(k),$$

вследствие ограниченности помехи (3.41), условие сходимости (3.4) будет иметь вид

$$\left|g(e(k),\delta)\right| \le \frac{2}{\gamma \left\|x(k)\right\|^2}[\left|e(k)\right| - \Delta(k)],$$

или

$$\left|g(e(k),\Delta)\right| \le \frac{2}{\gamma \left\|x(k)\right\|^2}[\left|e(k)\right| - \Delta)]. \qquad (3.48)$$

Таким образом, значение $g(e(k),\Delta)$ зависит как от величины свободно выбираемого параметра γ, так и от значений $x(k)$.

Использование зоны нечувствительности в алгоритме Качмажа и регуляризированном алгоритме Качмажа приводит соответственно к процедурам

$$\hat{\theta}(k) = \hat{\theta}(k-1) + \gamma \frac{g(e(k),\Delta)}{\left\|x(k)\right\|^2} x(k). \qquad (3.49)$$

и

$$\hat{\theta}(k) = \hat{\theta}(k-1) + \gamma \frac{g(e(k),\Delta)}{\beta + \left\|x(k)\right\|^2} x(k). \qquad (3.50)$$

Проанализируем более общую процедуру (3.50). Записав (3.50) относительно ошибок обучения $\tilde{\theta}(k)$, возведя обе части полученного выражения в квадрат и подставив в (3.4), имеем следующее условие сходимости процедуры:

$$\Delta V(k) = \gamma^2 g^2(e(k),\Delta)\|x(k)\|^2 - 2\gamma g(e(k),\Delta)(e(k)-\xi(k))(\beta+\|x(k)\|^2) \le 0. \qquad (3.51)$$

Отсюда по аналогии с проведенными выше преобразованиями, получаем следующее условие сходимости (3.50):

$$|g(e(k),\Delta)| \le \frac{2A}{\gamma}(|e(k)-\Delta)| \quad , \qquad (3.52)$$

где $A = 1 + \dfrac{\beta}{\|x(k)\|^2}$.

Таким образом, как и в случае процедуры (3.44), значение $g(e(k),\Delta)$ для (3.50) зависит от величины $x(k)$.

Модифицированная процедура обучения (3.11) с нелинейным преобразованием невязки будет иметь вид

$$\hat{\theta}(k) = \hat{\theta}(k-1) + \gamma \frac{g(e(k),\Delta)}{r(k)} x(k), \qquad (3.53)$$

где $r(k)$ вычисляется как и в (3.11).

Повторяя проведенные выше выкладки, нетрудно получить, что для сходимости данной модификации необходимо выполнение условия

$$\left|g(e(k),\Delta)\right| \le \frac{2}{\gamma}\frac{r(k)}{\left\|x(k)\right\|^2}(\left|e(k)\right| - \Delta). \qquad (3.54)$$

Как следует из (3.54), с ростом k требование к зоне нечувствительности для процедуры (3.53) ослабляется, и при $k \to \infty$ сходимость будет наблюдаться практически при любых $g(e(k),\Delta)$.

Следует отметить, что при наличии информации лишь об ограниченности помех $\xi_1(k)$ и $\xi_2(k)$ по амплитуде, процедуры (3.37)-(3.40) следует модифицировать путем использования в них зон нечувствительности так, как это было сделано для случая линейной Адалины, например, следующим образом:

$$V(k) = \mathrm{V}(k-1) + \gamma_V(k)\frac{\alpha_V(k)F\left(x, w(k-1)\right)e(k)}{\beta_V(k) + \left\|F\left(x, w(k-1)\right)\right\|^2}, \qquad (3.55)$$

$$\text{где } \alpha_V(k) = \begin{cases} g\left(e(k), \Delta_V(k)\right), \ \textit{если } \left\|e(k)\right\| > \Delta_V(k); \\ 0, \ \textit{если } \left\|e(k)\right\| \le \Delta_V(k). \end{cases} \qquad (3.56)$$

Так как при вычислении оценок матрицы W используется производная функции активации, например (1.4), то параметр зоны нечувствительности должен содержать коэффициент α, определяющий наклон сигмоиды

$$W(k) = W(k-1) + \gamma_W(k)\frac{\alpha_W(k)\nabla f\left(x(k), V(k-1)\right)e(k)x^T(k)}{\beta_W(k) + \left\|\nabla f\left(x, V(k-1)\right)x(k)\right\|^2}, \qquad (3.57)$$

$$\text{где } \alpha_W(k) = \begin{cases} g\left(e(k), \Delta_W(k)\right), \text{ если } \dfrac{\nabla f_{\min}}{\alpha}\left\|e(k)\right\| > \Delta_W(k); \\ 0, \text{ если } \dfrac{\nabla f_{\min}}{\alpha}\left\|e(k)\right\| \le \Delta_W(k), \end{cases} \tag{3.58}$$

где $\nabla f_{\min} = \min\left[\nabla f_1(k), \nabla f_2(k), ..., \nabla f_L(k)\right] > 0.$

3.2 Адаптация параметров процедур обучения

3.2.1 Адаптация параметра регуляризации

Как следует из рассмотрения одношаговых процедур, повышение их вычислительной устойчивости связано с введением параметра регуляризации β. Очевидно, что данный параметр влияет на свойства процедур обучения, так как при таком подходе вместо обычного квадратичного функционала используется регуляризованный, т.е. решается задача

$$\min_{\theta(k)}\left\{\left(y(k) - \theta^T(k)x(k)\right)^2 + \beta\left\|\theta(k)\right\|^2\right\}. \tag{3.59}$$

Как и для всякого искусственно вводимого в критерий или оценку параметра, для параметра регулярзации β не существует общих рекомендаций по его выбору. Нетрудно видеть, что с целью увеличения скорости сходимости данный параметр должен стремиться к нулю, а с целью повышения устойчивости оценки должен быть отличным от нуля.

Выбор оптимального значения β зависит от наличия априорной информации о свойствах исследуемого объекта, статистических свойствах полезных сигналов и помех и т.д. Как правило, на практике такая информация

отсутствует, поэтому при практической реализации регуляризованных алгоритмов применяют различного рода допущения и аппроксимации.

Так, например, в [198] предлагается аппроксимировать регуляризованный квадратичный функционал $E(\beta)$ параболой в окрестности точки β=0 и воспользоваться методом Ньютона-Рафсона.

Полученное в этой работе значение $\beta_{опт}$ определяется выражением

$$\beta_{опт} = \frac{\sigma_\xi^2 \sum_{i=1}^{N} \frac{1}{\mu_i^2}}{\sum_{i=1}^{N} \frac{3\sigma_\xi^2 + \gamma_i^2 \mu_i}{\mu_i^2}} \qquad (3.60)$$

где μ_i и γ_i соответственно характеристические числа матриц $x^T x$ и P - ортогональной матрицы, столбцы которой составлены из характеристических чисел матрицы $x^T x$.

Однако, как отмечается в данной работе, воспользоваться выражением для $\beta_{опт}$ на практике не представляется возможным, т.к. это выражение зависит от неизвестных параметров $\gamma_i \quad (i = 1 \div N)$ и $\mu_i \quad (i = 1 \div N)$ и зачастую неизвестной σ_ξ^2.

В связи с этим предлагается довольно сложная итеративная процедура оценивания β, неприменимая при on-line обучении.

Отметим, что достаточно удобной on-line процедурой настройки параметра β для нелинейной адалины является следующая, получаемая путем минимизации функционала $E(k)$ и записывающаяся в виде

$$\beta(k) = \beta(k-1) + \alpha \nabla_\beta E(k)\,|_{\beta=\beta(k-1)}, . \qquad (3.61)$$

где α – некоторый параметр, влияющий на скорость сходимости;

$$\nabla_{\beta} E(k) = \frac{\partial E(k)}{\partial \beta(k-1)} = e(k) \frac{\partial e(k)}{\partial \beta(k-1)}. \tag{3.62}$$

Так как

$$\frac{\partial e(k)}{\partial \beta(k-1)} = -\frac{\partial w(k)}{\partial \beta(k-1)} \nabla f(k) x(k), \tag{3.63}$$

а

$$\frac{\partial w(k)}{\partial \beta(k-1)} = -\frac{e(k-1)\nabla f(k-1)x(k-1)}{\left[\left\|\nabla f(k-1)x(k-1)\right\|^2 + \beta(k-1)\right]^2}, \tag{3.64}$$

то

$$\frac{\partial E(k)}{\partial \beta(k-1)} = -\frac{e(k)e(k-1)\nabla f(x(k))\nabla f(x(k-1))x^T(k)x(k-1)}{\left[\left\|\nabla f(k-1)x(k-1)\right\|^2 + \beta(k-1)\right]^2}. \tag{3.65}$$

Таким образом, окончательная рекуррентная процедура настройки параметра $\beta(k)$ примет вид

$$\beta(k) = \beta(k-1) - \alpha \frac{e(k)e(k-1)\nabla f(k)\nabla f(k-1)x^T(k)x(k-1)}{\left[\left\|\nabla f(k)x(k-1)\right\|^2 + \beta(k-1)\right]^2}. \tag{3.66}$$

Учитывая (3.32) и (3.38-65), можно записать

$$0 < \frac{1}{\left\|\nabla f(k)x(k)\right\|^2 + \beta(k)} < \frac{2}{\left\|\nabla f(k)x(k)\right\|^2}, \qquad (3.67)$$

откуда получаем ограничение для параметра регуляризации

$$-0.5\left\|\nabla f(k)x(k)\right\|^2 < \beta(k). \qquad (3.68)$$

Для повышения устойчивости оценок можно применить более грубые значения параметра регуляризации, получаемые, например, путем построения так называемых гребневых следов [198-200], представляющих собой графики зависимостей суммы квадратов отклонений, средней суммы квадратов ошибок, координат вектора гребневой оценки и т.д. от параметра регуляризации. Сопоставление графиков гребневых следов для разных показателей помогает установить в каких пределах должен лежать параметр регуляризации β и к каким последствиям приводит тот или иной выбор β.

При использовании регуляризованных оценок следует очитывать следующее. При выборе достаточно малых значений β значение суммы квадратов отклонений увеличивается незначительно по сравнению с суммой квадратов отклонений МНК-оценок, в то время как сами регуляризованные оценки будут отличаться от МНК-оценок более существенно. Поэтому выбор параметра регуляризации может осуществляться также, исходя из задания максимально допустимого значения суммы квадратов отклонений [198-200]. Если гребневый след дает качественную информацию о поведении важнейших характеристик оценок в зависимости от изменения β, то сумма квадратов отклонений обладает хорошей интерпретацией.

Однако практически всем известным классическим процедурам выбора параметра регуляризации присущ общий недостаток – они не учитывают

статистических свойств получаемых оценок. Поэтому на практике следует, скорее всего, исходить из конкретной ситуации и задавать параметр β достаточно малым, но в то же время позволяющим получать оценки в условиях мультиколлинеарности.

Попытка косвенного учета этих свойств была предпринята в работе [191], в которой в качестве критерия предложено рассматривать отношение сигнал-шум (ENR – echo-to-noise ratio), т.е. нормализованную величину

$$ENR = \frac{\sigma_y^2}{\sigma_\xi^2} = \frac{w^T R_x w}{\sigma_\xi^2}, \tag{3.69}$$

где $R_x = M\left\{x(k)x^T(k)\right\}$.

Минимизация $M\left\{\tilde{e}^2(k)\right\} = \sigma_\xi^2$, где $\tilde{e}(k) = y^*(k) - w^T(k-1)x(k)$, в предположении, что $\left\|x(k)\right\|^2 \approx N\sigma_x^2$ и

$$M\left\{\left|x(k)\right|\right\} = \sqrt{\frac{2}{\pi\sigma_x^2}}\sigma_x^2 = \alpha_x\sigma_x^2, \tag{3.70}$$

где $\alpha_x = \sqrt{\frac{2}{\pi\sigma_x^2}}$,

позволяет получить следующие выражения для нормализованного параметра регуляризации

- для процедуры (3.10)

$$\beta = \frac{N\left(1+\sqrt{1+ENR}\right)}{ENR}\sigma_x^2; \tag{3.71}$$

- для процедуры (3.19)

$$\beta = \frac{N\alpha_x\left(1+\sqrt{1+ENR}\right)}{ENR}\sigma_x^2; \tag{3.72}$$

- для процедуры (3.16)

$$\beta = \frac{1+\sqrt{1+ENR}}{ENR}\sigma_x^2; \tag{3.73}$$

- для процедуры (3.23)

$$\beta = \frac{1+\sqrt{1+ENR}}{ENR}. \tag{3.74}$$

Следует отметить, что полученные выражения для параметра регуляризации также трудно применимы, т.к. требуют значения σ_y^2 и σ_ξ^2, что не представляется возможным. Поэтому для вычисления данных значений также следует использовать оценки σ_x^2 и σ_ξ^2, полученные с помощью некоторых рекуррентных процедур (получение таких оценок будет рассмотрено ниже).

Обычно в работах, имеющих практическую направленность, рекомендуется выбирать β достаточно малым положительным числом и не рассматривается вопрос коррекции этого параметра в процессе обучения сети.

Можно рассмотреть задачу определения $\beta_{опт}$, обеспечивающего минимальное значение ошибки аппроксимации, которая в итоге будет определяться величиной σ_ξ^2.

3.2.2 Адаптация зоны нечувствительности.

Результаты проведенного анализа свидетельствуют о том, что для монотонной сходимости процедур должны выполняться соответствующие условия на каждом такте процесса обучения. Так как эти условия сходимости налагают вполне определенные ограничения на величину зоны нечувствительности, целесообразно ввести коррекцию зон нечувствительности рассмотренных алгоритмов на каждом такте [201]. С другой стороны, параметры зон нечувствительности зависят от характеристик присутствующих помех. Если наличие информации о характеристиках помехи (в частности, Δ, входящая в (3.41)) упрощает выбор зоны нечувствительности, то ее отсутствие существенно затрудняет ее выбор. Поэтому целесообразным представляется адаптивная настройка величины зоны нечувствительности, осуществляемая по мере поступления новой информации.

Настройка зон нечувствительности для процедур (3.55) и (3.56) может осуществляться следующим образом:

$$\Delta_V(k) = \Delta_V(k-1) + \frac{\alpha_V(k)\left\|e(k-1)\right\|}{\beta_V(k-1) + \left\|F\left(x, V(k-1)\right)\right\|^2}; \tag{3.75}$$

$$\Delta_W(k) = \Delta_W(k-1) + \frac{\alpha_W(k)\left\|e(k-1)\right\|}{\beta_W(k-1) + \left\|\nabla f\left(x, W(k-1)\right)x(k)\right\|^2}. \tag{3.76}$$

Пусть $\xi(k)-$ независимые случайные величины, симметрично расположенные относительно нуля, с функцией распределения $F(\xi)$. Тогда, если в пределе оценка $\hat{w}(k)$ сходится с w, то и величина зоны нечувствительности $\Delta(k)$ с ростом числа наблюдений должна сходиться к предельному значению Δ^*. В этом случае ошибка обучения стремится к

нулю. Обозначим через p вероятность попадания оценки вне отрезка $(f(k)-\hat{w}^T(k-1)x(k)-\Delta^*,\ f(k)-\hat{w}^T(k-1)x(k)+\Delta^*)$ при $\Delta=\Delta^*$, т.е. выберем величину зоны нечувствительности из условия $F(y(k)-\hat{w}^T(k-1)x(k)-\Delta=0.5p.$ В этом случае соответствующая рекуррентная процедура оценки Δ будет иметь вид

$$\Delta(k)=\Delta(k-1)-\frac{\gamma(k)}{2}[sign(y(k)-\hat{w}^T(k-1)x(k)+\Delta(k-1))-$$
$$-sign(y(k)-\hat{w}^T(k-1)x(k)-\Delta(k-1))-2(1-p)].$$

Здесь

$$\gamma(k)=\frac{1}{k}\frac{\Delta(k-1)}{1-p}\ ,\ p\in(0,1).$$

Как отмечается в [202], для распределения типа «равномерное с выбросами» при известном уровне выбросов p величина Δ^* получается при $p=0.5$. Величина зоны нечувствительности, рекуррентно вычисляемая в соответствии с полученным правилом, используется в модифицированной градиентной процедуре (3.10), представляющей собой ее персептронный вариант. Полученные выше условия сходимости процедур могут быть использованы для коррекции на каждом такте величин их зон нечувствительности. При этом правила настройки будут иметь следующий вид:

– для процедуры (3.44)

$$g(e(k),\Delta)=\begin{cases}e(k), & если\ \ |e(k)|\ge\dfrac{\Delta}{1-0.5\gamma\|x(k-1)\|^2};\\ \dfrac{2}{\gamma\|x(k-1)\|^2}(|e(k)|-\Delta), & если\ \ \Delta\le|e(k)|\le\dfrac{\Delta}{1-0.5\lambda\|x(k-1)\|^2};\\ 0, & если\ \ e(k)\le\Delta;\end{cases}$$

– для процедуры (3.49)

$$g(e(k),\Delta)=\begin{cases} e(k), & если \ \ |e(k)|\geq\dfrac{\Delta}{1-0.5\lambda};\\ \dfrac{2}{\gamma}(|e(k)|-\Delta), & если \ \ \delta\leq|e(k)|\leq\dfrac{\Delta}{1-0.5\lambda};\\ 0, & если \ \ e(k)\leq\Delta; \end{cases}$$

– для процедуры (3.50)

$$g(e(k),\Delta)=\begin{cases} e(k), & если \ \ |e(k)|\geq\dfrac{A\Delta}{1-0.5\gamma};\\ \dfrac{2A}{\gamma}(|e(k)|-\Delta), & если \ \ \Delta\leq|e(k)|\leq\dfrac{A\Delta}{1-0.5\gamma};\\ 0, & если \ \ e(k)\leq\Delta; \end{cases}$$

– для процедуры (3.53)

$$g(e(k),\Delta)=\begin{cases} e(k), & если \ \ |e(k)|\geq\dfrac{r(k)\Delta}{1-0.5\gamma\|x(k-1)\|^2};\\ \dfrac{2r(k)}{\gamma\|x(k-1)\|^2}(|e(k)|-\Delta), & если \ \ \Delta\leq|e(k)|\leq\dfrac{r(k)\Delta}{1-0.5\gamma\|x(k-1)\|^2};\\ 0, & если \ \ e(k)\leq\Delta; \end{cases}$$

3.3 Рекуррентные процедуры обучения, основанные на методе наименьших квадратов

Оценка МНК имеет вид

$$\Theta(k) = X^{+}(k)Y(k), \tag{3.77}$$

где $X^{+}(k)$ - матрица, псевдообратна X. В задачах обучения вместо вектора $Y(k)$ размерности kx1 используется вектор ошибок $E(k)$ той же размерности. Если матрица $X(k)$ имеет размерность Nxk, то в зависимости от количество уравнений (K) и числа неизвестных параметров (N) $X^{+}(k)$ принимает вид

$$X^{+}(k) = \begin{cases} \left[X(k)X^{T}(k) \right]^{-1} X(k), \text{ если } k \geq N; & (3.78) \\ X^{T}(k)\left[X^{T}(k)X(k) \right]^{-1}, \text{ если } k < N. & (3.79) \end{cases}$$

В традиционном МНК $X^{+}(k)$ имеет вид (3.78). В данной оценке используются все имеющиеся наблюдения K.

3.3.1 Рекуррентная форма МНК.

В настоящее время рекуррентный МНК (РМНК) получил, пожалуй, наиболее широкое распространение при решении в реальном времени задач идентификации, фильтрации, прогнозирования, адаптивного управления и т.п. Следует отметить, что существуют различные подходы к получению РМНК, среди которых наиболее распространенным является построение РМНК из обычного МНК на основании блочного представления матрицы наблюдений и использовании леммы об обращении матриц. Модификация минимизируемого квадратного функционала (введение весовых параметров, механизма экспоненциального сглаживания и т.д.) приводит к соответствующей модификации РМНК.

Необходимо отметить, что его применение предполагает линейность (псевдолинейность) используемой модели, т.е.

$$y(k) = f(\theta^T(k)\psi(k)), \tag{3.80}$$

поэтому РМНК применим в ИНС прямого распространения типа ADALINE, а также для настройки параметров выходного слоя многослойных сетей. При использовании модели псевдолинейной регрессии в подобных сетях в качестве вектора $\psi(k)$ в сети ADALINE, как и в классическом МНК, выбирается вектор входных сигналов, а в многослойных ИНС – вектор входных сигналов выходного слоя, т.е. вектор выходных сигналов предыдущего скрытого слоя. Следует, однако, заметить, что аналогами МНК в нелинейных моделях является метод Ньютона, Гаусса-Ньютона, Левенберга-Марквардта, а РМНК – соответствующие процедуры указанных методов. В связи с этим представляет интерес исследования вопросов получения рекуррентных процедур РМНК и изучение их свойств.

Однако РМНК может быть получен непосредственно путем минимизации некоторого квадратичного функционала. Так как для анализа вопросов сходимости рекуррентных алгоритмов широко используются критерии (функция Ляпунова) вида $\tilde{\Theta}^T P^{-1}\tilde{\Theta}$, где $\tilde{\Theta}$ - вектор ошибок оценивания, P^{-1}-положительно определенная матрица, то целесообразно воспользоваться таким критерием для получения самого алгоритма РМНК.

Такой подход был применен в работах [196, 203-204]. Как показано в [196], минимизация функционала

$$I(\theta) = (\theta - \hat{\theta}(k-1))^T P(k-1)(\theta - \hat{\theta}(k-1) + \gamma(y(k) - \theta^T\psi(k-1))^2 \tag{3.81}$$

приводит к следующей процедуре:

$$\hat{\theta}(k) = \hat{\theta}(k-1) + \gamma(k)P(k)\psi(k-1)(y(k) - \hat{\theta}^T(k-1)\psi(k-1)); \qquad (3.82)$$

$$P^{-1}(k) = P^{-1}(k-1) + \gamma(k)\psi(k-1)\psi^T(k-1). \qquad (3.83)$$

Здесь $\gamma(k) > 0$ - некоторый параметр.

Применяя к соотношению (3.83) лемму об обращении матриц и подставляя полученное выражение в (3.82), получаем рекуррентный аналог (3.82), (3.83).

$$\hat{\theta}(k) = \hat{\theta}(k-1) + \gamma(k)\frac{P(k-1)\psi(k-1)}{1+\gamma(k)\psi^T(k-1)P(k-1)\psi(k-1)}(y(k) - \hat{\theta}^T(k-1)\psi(k-1)). \qquad (3.84)$$

$$P(k) = P(k-1) - \gamma(k)\frac{P(k-1)\psi(k-1)\psi^T(k-1)P(k-1)}{1+\gamma(k)\psi^T(k-1)P(k-1)\psi(k-1)}. \qquad (3.85)$$

Выбирая различные значения параметра $\gamma(k)$, получаем различные модификации РМНК, в частности, при $\gamma(k) = 1$ имеем оптимальный РМНК.

Вопросы сходимости процедуры (3.84)-(3.85) рассмотрены в Приложении Б.

3.3.2 Рекуррентная форма МНК со скользящим окном

Оценка МНК со скользящим окном описывается соотношениями (3.77) и (3.78) при фиксированном числе наблюдений, т.е. при $K = const$. В дальнейшем будем обозначать это фиксированное число L, подразумевая, что $L \geq N$.

Особенностью алгоритмов с конечной памятью *L=const* является то, что на каждом шаге матрица наблюдений $X_L(k)$ формируется следующим образом: отбрасывается (*n-L*)-е наблюдение (последний столбец матрицы $X_L(k-1)$) и в матрицу включается новое, (*n*)-е наблюдение. В зависимости от того, как формируется новая матрица наблюдений (добавляется ли сначала новое наблюдение, а затем отбрасывается старое, либо же сначала отбрасывается старое наблюдение, а затем добавляется новое) возможны две рекуррентные форму МНК с окном. Вывод этих форм приведен в Приложении В.

Рекуррентная форма МНК со скользящим окном, получаемая путем добавления нового (*n*)-го наблюдения и последующего исключения старого (*n-L*+1)-го, описывается соотношениями

$$\theta(k)=\hat{\theta}(k-1)-\frac{U_{L+1}(k)x(k-L+1)}{1+x^T(k-L+1)U_{L+1}(k)x(k-L+1)}\left(y(k-L+1)-\hat{\theta}^T(k)x(k-L+1)\right); \tag{3. 86}$$

$$\hat{\theta}(k)=\theta(k-1)+\frac{S_L(k-1)x(k)}{1+x^T(k)S_L(k-1)x(k)}\left(y(k)-c^T(k-1)x(k)\right); \tag{3.87}$$

$$U_{L+1}(k)=S_L(k-1)-\frac{S_L(k-1)x(k)x^T(k)S_L(k-1)}{1+x^T(k)S_L(k-1)x(k)}; \tag{3. 88}$$

$$S_L(k)=U_L(k-1)-\frac{U_{L+1}(k)x(k-L+1)x^T(k-L+1)U_{L+1}(k)}{1-x^T(k-L+1)U_{L+1}(k)x(k-L+1)}. \tag{3.89}$$

Если же при построении оценки $\theta(k)$ сначала отбрасывается самое старое, (*n-L*+1)-е, наблюдение, а затем добавляется новое, то, как нетрудно показать, рекуррентная форма будет иметь вид

$$\theta k) = \hat{\theta}(k-1) + \frac{U_{L-1}(k-1)x(k)}{1 + x^T(k)U_{L-1}(k-1)x(k)}\Big(y(k) - \hat{\theta}^T(k)x(k)\Big); \tag{3.90}$$

$$\hat{\theta}(k) = \theta(k-1) - \\ -\frac{S_L(k-1)x(k-L+1)}{1 - x^T(k-L+1)U_{L-1}(k-1)x(k-L+1)}\Big(y(k-L+1) - \hat{\theta}^T(k-1)x(k-L+1)\Big); \tag{3.91}$$

$$U_{L-1}(k-1) = S_L(k-1) - \frac{S_L(k-1)x(k-L+1)x^T(k-L+1)S_L(k-1)}{1 - x^T(k-L+1)S_L(k-1)x(k-L+1)}; \tag{3.92}$$

$$S_L(k) = U_{L-1}(k-1) - \frac{U_{L-1}(k-1)x^T(k)x(k)U_{L-1}(k-1)}{1 + x^T(k)U_{L-1}(k-1)x(k)}. \tag{3. 93}$$

3.3.3 Рекуррентные формы проекционных процедур с $S < N$

Вывод рекуррентной формы проекционных процедур с $S < N$ приведен в Приложении В.

Рекуррентная форма многошаговых проекционных методов, описываемых соотношениями (3.77), (3.79) с $S < N$, может принимать тот или иной вид (по аналогии с рекуррентной формой МНК со скользящим окном) в зависимости от того, каким образом вычисляется используемая в них обратная матрица размерности sxs. На каждом шаге матрица наблюдений $X_s(k)$ формируется следующим образом: отбрасывается (n-s)-е наблюдение (последний столбец матрицы $X_s(k-1)$) и в матрицу включается последнее, (n)-е наблюдение, т.е.

$$\begin{aligned} X_s(k-1) &= \big(x(k-1), x(k-2), ..., x(k-s)\big) = X_{s-1}(k-1) \,|\, x(k-\mathrm{s}); \\ X_s(k) &= \big(x(k), x(k-1), ..., x(k-s+1)\big) = x(k) \,|\, X_{s-1}(k-1). \end{aligned} \tag{3.94}$$

Таким образом, наличие у $X_s(k-1)$ и $X_s(k)$ общих подматриц $X_{s-1}(k-1)$ позволяет производить вычисление $X_s(k)$ по $X_{s-1}(k-1)$, а значит и $\left(X_s^T(k)X_s(k)\right)^{-1}$ по $\left(X_s^T(k-1)X_s(k-1)\right)^{-1}$.

В окончательном виде рекуррентная форма может быть записана так (с учетом того, что $w^T(k-1)x(k-1)=\left(1-\gamma(k-1)\right)w^T(k-2)x(k-1)$)

$$\theta(k)=\theta(k-1)+\gamma(k)r_s(k), \tag{3.95}$$

$$\begin{aligned} &R_i\,(k-s+i)=R_{i-1}(k-s+i-1)- \\ &-\frac{R_{i-1}(k-s+i-1)x(k-s+i)x^T(k-s+i)R_{i-1}(k-s+i-1)}{\alpha(k-s+i)}, \end{aligned} \tag{3.96}$$

$$r_s(k)=\frac{R_{s-1}(k-1)x(k)}{\alpha(k)}e(k)+\left(1-\gamma(k-1)\right)\left(I-\frac{R_{s-1}(k-1)x(k)x^T(k)}{\alpha(k)}\right)r_{s-1}(k-1), \tag{3.97}$$

где $r(0)=0$; $\alpha(k)=x^T(k)R_{s-1}(k-1)x(k)$;

$\alpha(k-s+i)=x^T(k-s+i)R_{i-1}(k-s+i-1)x(k-s+i)$; $R_0(k-s)=I$.

Таким образом, применение рекуррентной формы (3.95) - (3.97) приводит к тому, что вместо непосредственного обращения матрицы $X_s^T(k-1)X_s(k-1)$ на каждом шаге процесса идентификации выполняется *s*-1 вычисление матрицы *R* в соответствии с (3.96).

Процедура (3.95) - (3.97) будет работоспособной только после $n \geq s$ шагов. До этого момента следует применять процедуру, у которой глубина памяти s будет переменной, т.е.

$$c(k) = c(k-1) + \gamma(k) r_s(k); \tag{3.98}$$

$$r(k) = \frac{R(k)x(k)}{\alpha(k)} e(k) + \left(1 - \gamma(k-1)\right)\left(I - \frac{R(k-1)x(k)x^T(k)}{\alpha(k)} \right) r(k-1), \tag{3.99}$$

$$R(k) = R(k-1) - \frac{R(k-1)x(k)x^T(k)R(k-1)}{\alpha(k)}, \tag{3.100}$$

где $R(0) = I, r(0) = 0$.

В этом случае на первом шаге алгоритм (3.98)-(3.100) совпадает с алгоритмом Качмажа, на втором – с двухшаговым проекционным алгоритмом и т.д.

Следует отметить, что полученная форма алгоритма (3.95)-(3.97) является достаточно общей. Как частный случай, при $\gamma = I$ из нее следует оптимальный конечно-шаговый алгоритм, рассмотренный в работах [205-206].

Заметим, что стоящие в знаменателях приведенных выше соотношений величины $\alpha(k) = x^T(k)R(k-1)x(k)$ на некоторых шагах процесса идентификации могут обращаться в нуль. Величины $\alpha(k)$ являются отношением определителей Грама на двух соседних шагах процесса идентификации и характеризует степень линейной зависимости $x(k)$ с векторами $x(k-1),...,x(k-s+1)$. Поэтому если на некоторых шагах процесса идентификации эти величины близки к нулю, то на этих шагах коррекция

оценки $c(k)$ не производится и данное измерение входного сигнала $x(k)$ в алгоритм не включается. Это можно объяснить следующим фактом. Так как очередная оценка вектора искомых параметров $c(k)$, построенная с помощью данного алгоритма, будет являться проекцией $w(k-1)$ на $L_s(k)$ и проходящую через $c(k-1)$, т.е. она удовлетворяет s последним уравнениям $y(i)=c^T(k-1)x(i)$, то она будет удовлетворять и новому, так как вновь поступивший вектор $x(k)$ является линейной комбинацией предыдущих векторов.

Поэтому для обеспечения вычислительной устойчивости алгоритма следует наложить ограничения на величину $\alpha(k)$. В качестве такого ограничения можно взять, например, следующее: настройка коэффициентов происходит только при выполнении условия $\alpha(k)\geq\delta$, где δ — величина, играющая роль регуляризатора.

Еще одна рекуррентная форма, приведенная в Приложении В, может быть получена методами, рассмотренными в работе [207].

3.3.4 Модификации РМНК с зоной нечувствительности.

Как отмечалось выше, использование зоны нечувствительности предполагает коррекцию вектора оцениваемых параметров θ, если ошибка сети $e(k)=y(k)-\theta^T(k-1)\psi(k)$ превышает некоторый порог δ, определяющий величину зоны нечувствительности.

В настоящее время существует целый ряд процедур, основанных на РМНК и использующих зону нечувствительности. В частности, такие процедуры реализованы в работах [208-209]. Несмотря на эффективность этих процедур, их использование в критических системах наталкивается на серьезные затруднения.

Так, сложность оптимального алгоритма Фогеля-Хуанга [208] обусловлена необходимостью отыскания на каждой итерации глобального минимума многоэкстремальной функции и не позволяет использовать его для обучения в реальном времени.

Условия сходимости алгоритма Лозано-Лила-Ортеги, имеющем вид

$$\hat{\theta}(k)=\hat{\theta}(k-1)+\frac{\alpha P(k)\psi(k-1)}{1+\psi^T(k-1)P(k-1)\psi(k-1)}(|e(k)|-\delta_l)\operatorname{sign}e(k); \qquad (3.101)$$

$$P^{-1}(k)=\begin{cases}P^{-1}(k-1)+\dfrac{\alpha P(k-1)\psi(k-1)}{(1+\psi^T(k-1)P(k-1)\psi(k-1))e(k)}(|e(k)|-\delta_l)sign\,e(k), & |e(k)|>\delta_l;\\ P^{-1}(k-1), \quad |e(k)|\le\delta_l\ . \end{cases} \qquad (3.102)$$

$$\delta_1=\sqrt{1+\alpha} \qquad \alpha\in(0,1), \qquad (3.103)$$

требует ограниченности значений $\psi^T(k\text{-}1)P(k\text{-}1)\psi(k\text{-}1)$. При этом

$$\lim_{k\to\infty}|e(k)|=\sqrt{1+\alpha};\ \alpha\in(0,1), \qquad (3.104)$$

т.е. ошибка $e(k)$ никогда не может быть по модулю меньше заданных ограничений δ.

В алгоритме Канудас де Вита-Каррильо [209]

$$\hat{\theta}(k)=\hat{\theta}(k-1)+\frac{\alpha P(k-1)\psi(k-1)}{\psi^T(k-1)P(k-1)\psi(k-1))}(|e(k)|-\delta)sign\,e(k); \qquad (3.105)$$

$$P(k)=\gamma^{-1}(P(k-1)-\frac{\alpha(k)P(k-1)\psi(k-1)\psi^T(k-1)P(k-1)}{\psi^T(k-1)P(k-1)\psi(k-1)}(1-\frac{\delta}{|e(k)|}),\ \gamma\in(0,1];$$

(3.106)

$$\alpha(k)=\begin{cases}1, |e(k)|>\delta \quad или \quad \psi^T(k-1)P(k-1)\psi(k-1)=0;\\ 0, |e(k)|\le\delta \quad ,\end{cases} \tag{3.107}$$

при $|e(k)\not>\delta$ и малых значениях $\psi^{\mathrm{T}}(k\text{-}1)\mathrm{P}(k\text{-}1)\psi(k\text{-}1)$ может возникнуть режим неустойчивости. Кроме того, в ситуации, когда $\alpha(k)=0$, невозможно гарантировать выполнение условия $|e(k)|\le\delta$ в предположении, что $\psi^T(k\text{-}1)P(k\text{-}1)\psi(k\text{-}1)$ ограничено.

Для устранения недостатков рассмотренных процедур в [210] предложен алгоритм, являющийся своеобразной комбинацией РМНК и процедур (3.101)-(3.103) и (3.105)-(3.107)

$$\hat{\theta}(k)=\hat{\theta}(k\text{-}1)+\frac{\alpha(k)P(k\text{-}1)\psi(k\text{-}1)}{1+\psi^{\mathrm{T}}(k\text{-}1)P(k\text{-}1)\psi(k\text{-}1)}(|e(k)|-\delta)\mathrm{signe}(k); \tag{3.108}$$

$$P(k)=P(k\text{-}1)-\frac{\alpha(k)P(k\text{-}1)\psi(k\text{-}1)\psi^T(k\text{-}1)P(k\text{-}1)}{|e(k)|+(2|e(k)|-\delta)\psi^T(k)P(k\text{-}1)\psi(k\text{-}1)}(|e(k)|-\delta); \tag{3.109}$$

$$\alpha(k)=\begin{cases}1, \text{ если } |e(k)|>\delta,\\ 0, \text{ если } |e(k)|\le\delta.\end{cases} \tag{3.110}$$

Рассмотрим некоторые процедуры, основанные на РМНК и использующие зону нечувствительности. В [211] была предложена процедура (3.84), (3.85) с $\gamma(k)$, определяемым по правилу

$$\gamma(k)=\alpha\frac{g(e(k),\beta\delta)}{1+\psi^T(k-1)P(k-1)\psi(k-1)}, \tag{3.111}$$

где

$$\alpha\in(0,1];\beta=\sqrt{1+\alpha};$$

$$g(e(k),\beta\delta)=\begin{cases}\dfrac{f(e(k),\beta\delta)}{e(k)} & , \quad если \quad |e(k)|>\beta\delta;\\ 0, & если \quad |e(k)|\le\beta\delta.\end{cases} \tag{3.112}$$

В этом случае

$$\begin{aligned}&1+\gamma(k)\psi^T(k-1)P(k-1)\psi(k-1)=\\&=\frac{1+(1+\alpha g(e(k)\ ,\beta\delta))\psi^T(k-1)P(k-1)\psi(k-1)}{1+\psi^T(k-1)P(k-1)\psi(k-1)}\end{aligned} \tag{3.113}$$

и подстановка данного выражения и значения γ, определяемого в соответствии с (3.111) в (3.84), (3.85) приводит к следующей процедуре

$$\hat\theta(k)=\begin{cases}\hat\theta(k-1)+\dfrac{\alpha P(k-1)g(e(k),\beta\delta)}{1+(1+\alpha g(e(k),\beta\delta))\psi^T(k-1)P(k-1)\psi(k-1)} & \text{если } |e(k)|>\beta\delta;\\ \hat\theta(k-1), & в\ противном\ случае\ .\end{cases} \tag{3.114}$$

$$P(k)=$$

$$=\begin{cases}P(k-1)-\alpha g(e(k),\beta\delta)\dfrac{P(k-1)\psi(k-1)\psi^T(k-1)P(k-1)}{1+(1+\alpha g(e(k),\beta\delta))\psi^T(k-1)P(k-1)\psi(k-1)} & \text{если } |e(k)|>\beta\delta;\\ P(k-1), & в\ противном\ случае\ .\end{cases} \tag{3.115}$$

В работах [212-213] была предложена процедура (3.84), (3.85) с $\gamma(k)$, определяемым выражением

$$\gamma(k) = \alpha \frac{g(e(k), \beta\delta)}{1 + (1 - \alpha g(e(k), \beta\delta)\psi^T(k-1)P(k-1)\psi(k-1)}, \qquad (3.116)$$

где $\alpha \in (0,1]; \quad \beta = \sqrt{\beta_0 + \frac{1}{1-\alpha}}; \quad \beta_0 > 0$.

Подставляя данное выражение для $\gamma(k)$ в (3.84), (3.85), получаем

$$\hat{\theta}(k) = \begin{cases} \hat{\theta}(k-1) + \dfrac{\alpha(k)P(k-1)\psi(k-1)f(e(k), \beta\delta)}{1 + \psi^T(k-1)P(k-1)\psi(k-1)} & \text{если} \quad |e(k)| > \beta\delta; \\ \hat{\theta}(k-1), & \textit{в противном случае}; \end{cases} \qquad (3.117)$$

$$P(k) = \begin{cases} P(k-1) - \alpha g(e(k), \beta\delta)\dfrac{P(k-1)\psi(k-1)\psi(k-1)^T P(k-1)}{1 + \psi^T(k-1)P(k-1)\psi(k-1)} & \text{, если} \quad |e(k)| > \beta\delta; \\ P(k-1), & \textit{в противном случае} \quad . \end{cases} \qquad (3.118)$$

Отметим, что вопросы сходимости различных модификаций РМНК с зоной нечувствительности рассмотрены в Приложении Г.

3.4 Традиционное обучение, основанное на методах Гаусса-Ньютона и Левенберга-Марквардта

3.4.1 Обучение в режиме off-line.

При обучении в режиме off-line на заданном количестве примеров $\{y(k), x(k)\}$, $k = \overline{1, M}$ для определения оценки параметров $\hat{\theta}$ обычно используют квадратичный критерий

$$F(M,\theta)=\frac{1}{2M}\sum_{l=1}^{M}e^2(l,\theta). \qquad (3.119)$$

Задавая некоторые начальные значения $\hat{\theta}_0^M$, многие процедуры, минимизирующие (3.119), можно описать соотношением [99, 214]

$$\hat{\theta}_i^M=\hat{\theta}_{i-1}^M-\gamma_i^M\left[R_i^M\right]^{-1}\nabla F_i^M\left(\hat{\theta}_{i-1}^M\right), \qquad (3.120)$$

где $\hat{\theta}_i^M$ – оценка вектора параметров на *i*-ой итерации; R_i^M – матрица усиления; γ_i^M – некоторый параметр, влияющий на длительность процесса обучения.

Выбирая различные γ_i^M и R_i^M, получаем соответствующие процедуры обучения.

Так при $R_i^M=I$ и $\gamma_i^M=const$ имеем традиционный алгоритм обратного распространения ошибки.

Если $\gamma_i^M=1$ и

$$R_i^M=\frac{1}{M}\sum_{l=1}^{M}\nabla f(l,\hat{\theta}_i)\nabla f^T(l,\hat{\theta}_i), \qquad (3.121)$$

где $\nabla f(l,\hat{\theta}_i)=\dfrac{\partial f(l,\theta)}{\partial\theta}$,

имеем алгоритм Гаусса-Ньютона [99, 215].

Выбор $\gamma_i^M=1$ и

$$R_i^M=\frac{1}{M}\sum_{l=1}^{M}\nabla f(l,\hat{\theta}_i)\nabla f^T(l,\hat{\theta}_i)+\delta_i I, \qquad (3.122)$$

где $\delta_i \geq 0$, приводит к процедуре Левенберга- Марквардта [99-100].

Как отмечалось выше, данные процедуры предназначены для работы в режиме off-line.

3.4.2 Рекуррентные формы процедур обучения: обучение в режиме on-line.

Для режима on-line характерно пошаговое поступление информации и соответствующее ему потактовое обучение. Для получения соответствующих рекуррентных форм процедур обучения поступим следующим образом.

Рассмотрим взвешенный квадратичный функционал

$$F\left[e(k)\right] = \frac{1}{2}\sum_{i=1}^{K} \lambda^{K-i} e^2(i,\theta), \tag{3.123}$$

где $\lambda \in (0,1]$ – параметр взвешивания.

Процедура (Д.2) может быть записана следующим образом (получение процедуры дано в Приложении Д):

$$\hat{\theta}(k) = \hat{\theta}(k-1) + H^{-1}(k)\nabla f(k,\theta)e(k,\theta), \tag{3.124}$$

$$H(k) = \lambda H(k-1) + \nabla f(k,\theta)\nabla f^T(k,\theta). \tag{3.125}$$

Как и (3.120), процедуры (3.124)-(3.125) являются достаточно общими и описывают различные рекуррентные процедуры.

Для режима on-line процедура обучения (3.120) может быть записана так:

$$\hat{\theta}(k) = \hat{\theta}(k-1) + \gamma(k)\left[R\left(k\right)\right]^{-1}\nabla f(k,\hat{\theta}(k-1))e(k,\hat{\theta}(k-1)). \tag{3.126}$$

Из (3.126) при выборе $R(k)=I$ получаем градиентный алгоритм (алгоритм обратного распространения ошибки)

$$\hat{\theta}(k)=\hat{\theta}(k-1)+\gamma(k)\nabla f(k,\hat{\theta}(k-1))e(k,\hat{\theta}(k-1)), \qquad (3.127)$$

где $\gamma(k)$ - параметр, влияющий на скорость сходимости (отметим, что выбор $\gamma(k)=\gamma\left\|\nabla f(k,\hat{\theta}(k-1))\right\|^{-2}$ приводит к алгоритму (3.1)).

При $R(k)=(1-\lambda)H(k)$ и $\gamma(k)=(1-\lambda)$ из (3.124) , (3.125) следует процедура Гаусса-Ньютона

$$\hat{\theta}(k)=\hat{\theta}(k-1)+(1-\lambda)\left[R(k)\right]^{-1}\nabla f(k,\theta)e(k,\theta), \qquad (3.128)$$

$$R(k)=\lambda R(k-1)+(1-\lambda)\nabla f(k,\hat{\theta})\nabla f^{T}(k,\hat{\theta}). \qquad (3.129)$$

Выбор $R(k)=(1-\lambda)H(k)+\delta I$ и $\gamma(k)=(1-\lambda)$ приводит к процедуре Левенберга-Марквардта

$$\hat{\theta}(k)=\hat{\theta}(k-1)+(1-\lambda)\left[R(k)\right]^{-1}\nabla f(k,\theta)e(k,\theta), \qquad (3.130)$$

$$R(k)=\lambda R(k-1)+(1-\lambda)\left(\nabla f(k,\hat{\theta})\nabla f^{T}(k,\hat{\theta})+\delta I\right). \qquad (3.131)$$

По аналогии с РМНК, входящие в данные процедуры обратные матрицы также могут быть вычислены рекуррентно. Если прямое обращение матрицы $R(k)$ обладает вычислительной сложностью $O(d^3)$, где d – размерность θ, то рекуррентные вычисления $\left[R(k)\right]^{-1}$ понижает эту сложность до $O(d^2)$.

Для процедуры Гаусса-Ньютона это получается просто. Для этого введем обозначение $P(k) = (1-\lambda)R^{-1}(k)$ и применим лемму об обращении матрицы к (3.129).

Тогда окончательно рекуррентная процедура Гаусса-Ньютона (3.128), (3.129) может быть записана следующим образом:

$$\hat{\theta}(k) = \hat{\theta}(k-1) + P(k)\nabla f(k,\theta)e(k,\theta); \qquad (3.132)$$

$$P(k) = \frac{1}{\lambda}\left(P(k-1) - \frac{P(k-1)\nabla f(k,\theta)\nabla f^T(k,\theta)P(k-1)}{\lambda + \nabla f^T(k,\theta)P(k-1)\nabla f(k,\theta)} \right). \qquad (3.133)$$

Наличие в обращаемой матрице $Q(k)$ в алгоритме Левенберга- Марквардта слагаемого δI несколько усложняет задачу.

Для квадратичного функционала рекуррентный алгоритм Левенберга-Марквардта был получен в [216] и применен для решения задачи нейросетевой нелинейной адаптивной фильтрации в [217]. В этих работах используется аппроксимация δI матрицей $I_N(k)$ размерности NxN, все элементы которой равны нулю за исключением диагонального, $(k \bmod N + 1)$-го, равного единице.

В этом случае после N шагов имеем

$$\sum_{k=i+1}^{i+N} N\delta I_N(k) = N\delta I.$$

Если воспользоваться данной аппроксимацией, то после несложных преобразований в окончательном виде рекуррентный алгоритм Левенберга-Марквардта можно записать следующим образом:

$$\hat{\theta}(k+1) = \hat{\theta}(k) + P(k)\nabla\hat{f}(k)e(k); \tag{3.134}$$

$$P(k) = P(k-1) - P(k-1)\nabla\hat{f}^*(k)K^{-1}(k)\nabla^T\hat{f}^*(k)P(k-1), \tag{3.135}$$

где

$$K(k) = \nabla^T\hat{f}^*(k)P(k-1)\nabla\hat{f}^*(k) + \Lambda^*(k);$$

$$\Lambda^{*-1}(k) = \begin{pmatrix} 1 & 0 \\ 0 & \delta N \end{pmatrix};$$

$$\nabla\hat{f}^*(k) = \begin{pmatrix} \nabla^T\hat{f}^*(k) \\ 0\ldots0 \quad 1\ldots0 \end{pmatrix}^T.$$

$$\uparrow$$

$$\text{позиция} = (k \bmod N + 1)$$

Таким образом, аппроксимация имеет вид

$$R(k) = \lambda R(k-1) + (1-\lambda)\left(\nabla f(k,\hat{\theta})\nabla f^T(k,\hat{\theta}) + \delta I_N(k)\right). \tag{3.136}$$

или

$$R(k) = \lambda R(k-1) + (1-\lambda)\left(\nabla f^*(k,\hat{\theta})\Lambda^{*-1}(k)\nabla f^T(k,\hat{\theta})\right). \tag{3.137}$$

Применение леммы об обращении матрицы, с учетом того, что $P(k) = (1-\lambda)R^{-1}(k)$, и подстановка полученного соотношения в (3.130) позволяют записать окончательно следующие соотношения, описывающие рекурентную процедуру Левенберга-Марквардта:

$$\hat{\theta}(k+1) = \hat{\theta}(k) + P(k)\nabla\hat{f}(k,\theta)e(k,\theta), \tag{3.138}$$

$$P(k)=\frac{1}{\lambda}\Big[P(k-1)-P(k-1)\nabla f^{*}(k,\theta)K^{-1}(k)\nabla^{T}f^{*}(k,\theta)P(k-1)\Big] \qquad (3.139)$$

$$K(k)=\nabla^{T}f^{*}(k,\theta)P(k-1)\nabla f^{*}(k,\theta)+\lambda\Lambda^{*}(k)\,. \qquad (3.140)$$

Как следует из (3.134)-(3.135), в этом алгоритме также необходимо обращать матрицу $K(k)$. Однако размерность ее равна 2x2, что значительно меньше размерности матрицы $Q(k)$.

Начальное значение матрицы $P(0)$ как в алгоритме (3.132), (3.133), так и в (3.136), (3.137) выбирается по аналогии с рекуррентным МНК (РМНК), т.е. $P(0)=\lambda I$, где $\lambda>>1$, а начальная размерность единичной матрицы I задается $D\times D$, где $D=1+(M^{2}+M+1)$ - количество настраиваемых параметров сети, содержащей 1 нейрон. Так как после введения в сеть нового, n-го, нейрона размерность $P(k)$ увеличивается, то значения элементов матрицы $P(k)$ сбрасываются и инициализируются заново, при этом D становится равным $D=1+n(M^{2}+M+1)$, где n - текущее количество нейронов в сети.

3.4.3 Модификация процедур обучения Гаусса-Ньютона и Левенберга-Марквардта с зоной нечувствительности.

Алгоритм обучения Гаусса-Ньютона с зоной нечувствительности имеет вид

$$\hat{\theta}(k)=\hat{\theta}(k-1)+\frac{\alpha P(k-1)\nabla\hat{f}(k)e(k,\theta)}{1+\nabla^{T}\hat{f}(k)P(k-1)\nabla\hat{f}(k)}, \qquad (3.141)$$

$$P(k)=P(k-1)-\frac{\alpha P(k-1)\nabla\hat{f}(k)\nabla^{T}\hat{f}(k)P\big(k-1\big)}{1+\nabla^{T}\hat{f}(k)P(k-1)\nabla\hat{f}(k)}, \qquad (3.142)$$

$$\alpha = \begin{cases} 1, если \ |e(k)| > \Delta; \\ 0, если \ |e(k)| \le \Delta. \end{cases} \qquad (3.143)$$

Рекуррентную же процедуру Левенберга-Марквардта с зоной нечувствительности можно записать следующим образом:

$$\hat{\theta}(k+1) = \hat{\theta}(k) + \alpha P(k) \nabla \hat{f}(k,\theta) e(k,\theta), \qquad (3.144)$$

$$P(k) = \frac{1}{\lambda}\left[P(k-1) - \alpha P(k-1) \nabla f^*(k,\theta) K^{-1}(k) \nabla^T f^*(k,\theta) P(k-1) \right] \qquad (3.145)$$

где α вычисляется в соответствии с (3.143), а $K(k)$ – с помощью (3.140).

3.5 Исследование скорости сходимости и надежности процедур обучения

Большое количество существующих алгоритмов минимизации функционала, используемых при обучении сетей, затрудняет их выбор. Конкретные практические рекомендации могут быть получены путем имитационного моделирования различных алгоритмов. Следует отметить, что выбор конкретной стратегии обучения и структуры сети определяется решаемой задачей. В данном разделе проведен сравнительный анализ эффективности алгоритмов обучения МП при решении тестовых задач [218, 219] аппроксимации тригонометрических функций sin(x), tan(x), sin(x)cos(2x) (рис.3.1). Решение рассмотренных задач осуществлялось с использованием персептронных архитектур типа 1-3-1, 1-10-1 и 1-5-1 соответственно. Для обучения сетей использовались следующие алгоритмы:

- традиционный алгоритм обратного распространения ошибки (A);
- алгоритм Качмажа (B);

– Нагумо-Ноды (C);

– проекционные процедуры обучения (D);

– РМНК(E);

– Гудвина-Рэмеджа-Кейнеса (F)

– метод Левенберга-Марквардта (G).

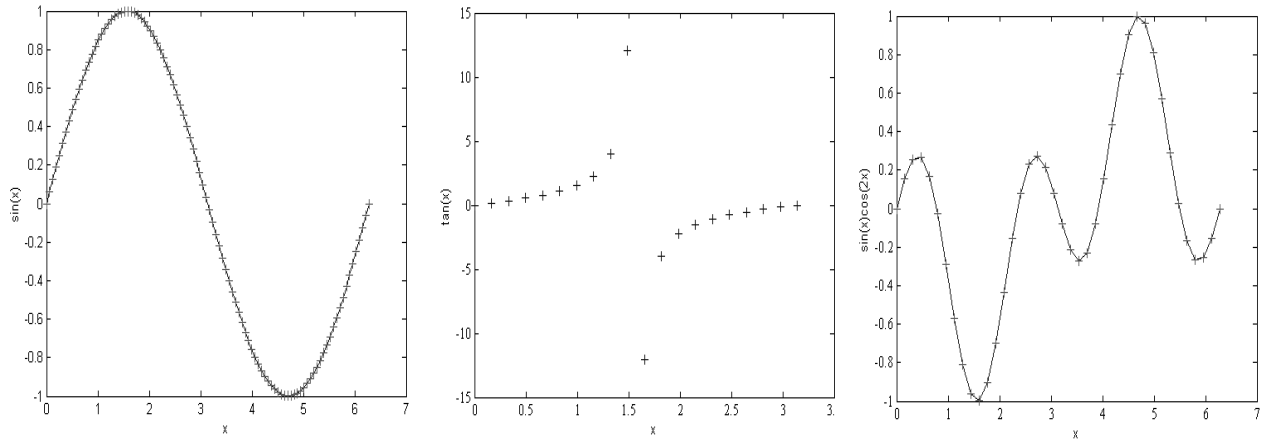


Рисунок 3.1 – Задачи аппроксимации тригонометрических функций

Все алгоритмы выполнялись в пакетном режиме по 100 раз с новыми начальными значениями весов и смещений. Максимальное число допустимых итераций – 100000. При решении тестовых задач использовалась логистическая функция активации для нейронов во входном и скрытом слоях, а для нейронов в выходном слое – линейная. В качестве оценки использовались два аспекта работы алгоритмов – “глобальная надежность” (т.е. насколько алгоритм склонен останавливаться в локальных минимумах) и скорость обучения [220].

Численной оценкой глобальной надежности алгоритма является процент успешной сходимости (ПС) к решению с заданной степенью точности, а скорости сходимости – среднее число итераций (СЧИ), необходимое для достижения решения при заданной точности. Результаты моделирования сведены в таблицы (табл.3.1-табл.3.3) и представлены на графиках (рис.3.2-рис.3.7).

Классические многомерные методы оптимизации можно расположить в следующем порядке (в порядке уменьшения вычислительной устойчивости и скорости сходимости): проекционные процедуры обучения; метод Левенберга-Марквардта; РМНК; метод Качмажа.

Однако этот порядок не учитывает вычислительной стоимости каждой итерации или метода выбора длины шага. Метод Качмажа оценен как худший, различие между другими менее четко.

Исследования показали, что методы второго порядка сходятся значительно быстрее методов первого порядка. Самым быстрым оказался метод Левенберга-Марквардта, но его нельзя рекомендовать для любых задач из-за чувствительности к ошибкам округления. Эффективное выполнение проекционных процедур обучения заслуживает самой высокой оценки. Он требует меньших вычислительных затрат на каждой итерации, чем метод РМНК, и, в отличие от метода Левенберга-Марквардта, не чувствителен к ошибкам округления. Кроме того, теоретическая скорость сходимости данного метода выше, чем у РМНК. К сожалению, высокие затраты, необходимые для хранения данных, не позволяют использовать метод Левенберга-Марквардта для решения задач большой размерности. Далее идут методы модификации метода Качмажа, которые можно рекомендовать для решения задач средней и большой степени сложности ввиду простоты реализации, невысоких вычислительных затрат и малого объема требуемой памяти.

Что касается сложности реализации, то рассмотренные многомерные алгоритмы должны быть расположены в порядке, обратном представленному выше.

Выводы по разделу 3

1. Анализ методов обучения ЭИНС показал, что В настоящее время существует большое число методов настройки параметров сети, отличающихся объемом используемой информации, влияющим как на динамические свойства

алгоритмов, так и на их вычислительную сложность. качество обучения зависит от того, насколько удачно был выбран функционал ошибки. Одним из основных факторов, влияющих на эффективность выбора функционала, является наличие информации о свойствах помехи. В настоящее время существует два принципиально различных подхода, основанных на том, что

- существует информация либо о некоторых статистических свойствах помехи, либо о принадлежности ее распределения помехи некоторому известному классу;
- несмотря на природу помехи, она предполагается ограниченной.

2. Рассмотрены наиболее простые в вычислительном отношении одношаговые градиентные процедуры обучения, минимизирующие квадратичный функционал и включающие алгоритмы Качмажа (Уидроу-Хоффа), Нагумо-Ноды, Гудвина-Рэмеджа-Кейнеса. Получены их модификации, содержащие зону нечувствительности, введение которой, огрубляя процедуру обучения, обеспечивает их работу при наличии ограниченных помех. Рассмотрены вопросы сходимости, обоснована необходимость адаптации величины зоны нечувствительности при отсутствии информации о свойствах помех и получены соответствующие процедуры ее настройки.

3. Рассмотрены модификации РМНК, в частности, содержащие зону нечувствительности. Получено неравенство, к проверке которого сводится анализ сходимости процедур.

4. Рассмотрены проекционные методы обучения, занимающие промежуточное положение между одношаговыми процедурами и РМНК.

5. Рассмотрены рекуррентные процедуры Гаусса-Ньютона и Левенберга-Марквардта и предложены их модификации, содержащие зону нечувствительности.

Таблица 3.1 – Оценка методов обучения при моделировании функции sin(x)

Метод	E=0.1		E=0.01		E=0.001		E=0.0001	
	СЧИ	ПС	СЧИ	ПС	СЧИ	ПС	СЧИ	ПС
A	43.31	100	2574.3	84	14093	89	38511	76
B	46.16	100	1201.0	88	14512	79	29837	68
C	17.8	100	1212.7	90	4395.8	87	15737	78
D	3.11	100	29.35	100	49.36	99	114.30	99
E	4.71	100	98.98	89	201.18	89	1756.6	88
F	4.29	100	319.92	88	3004.7	91	7238.4	48
G	2.56	100	10.23	88	10.734	94	11.51	91

Таблица 3.2 – Оценка методов обучения при моделировании функции tan(x)

Метод	E=0.1		E=0.01		E=0.001		E=0.0001	
	СЧИ	ПС	СЧИ	ПС	СЧИ	ПС	СЧИ	ПС
A	10567	82	57821	56	-	-	-	-
B	9567.6	86	44813	79	-	-	-	-
C	7206.0	82	30672	71	-	-	-	-
D	67.138	87	141.23	88	577.55	62	366.28	32
E	615.91	100	1279.0	97	4635.1	18	-	-
F	615.91	100	1279.0	97	5861.4	23	-	-
G	69.51	100	117.54	99	154.8	95	433.24	76

Таблица 3.3 – Оценка методов обучения при моделировании функции sin(x)cos(2x)

Метод	E=0.1		E=0.01		E=0.001		E=0.0001	
	СЧИ	ПС	СЧИ	ПС	СЧИ	ПС	СЧИ	ПС
A	57.16	100	615.99	100	11549	45	38511	41
B	57.9	100	616.76	100	13371	39	42871	37
C	25.24	100	282	100	10883	57	26374	48
D	4.69	100	14.56	100	136.79	100	271.36	100
E	7.72	100	33.08	100	595.86	92	4830.5	37
F	11.07	100	118.6	100	3748.6	74	4968.1	20
G	1.13	100	3.87	100	9.02	99	40.21	96

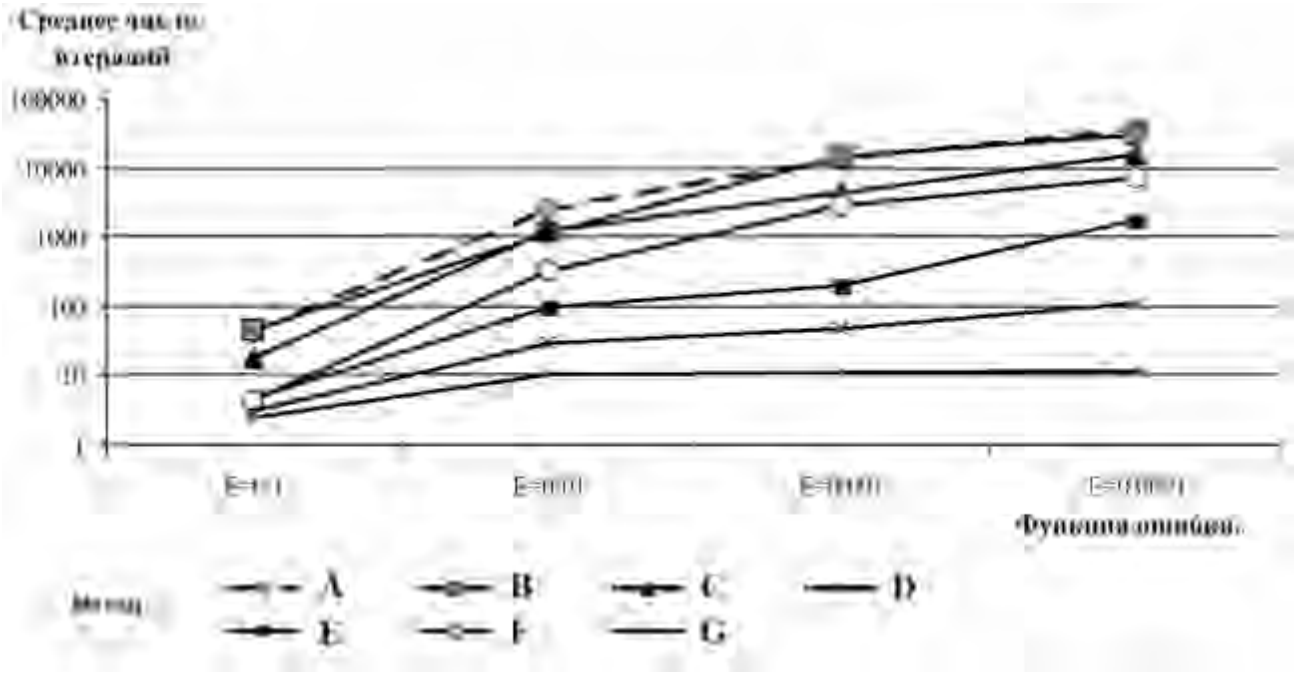


Рисунок 3.2 – Оценка скорости сходимости алгоритмов обучения при моделировании функции sin(x)

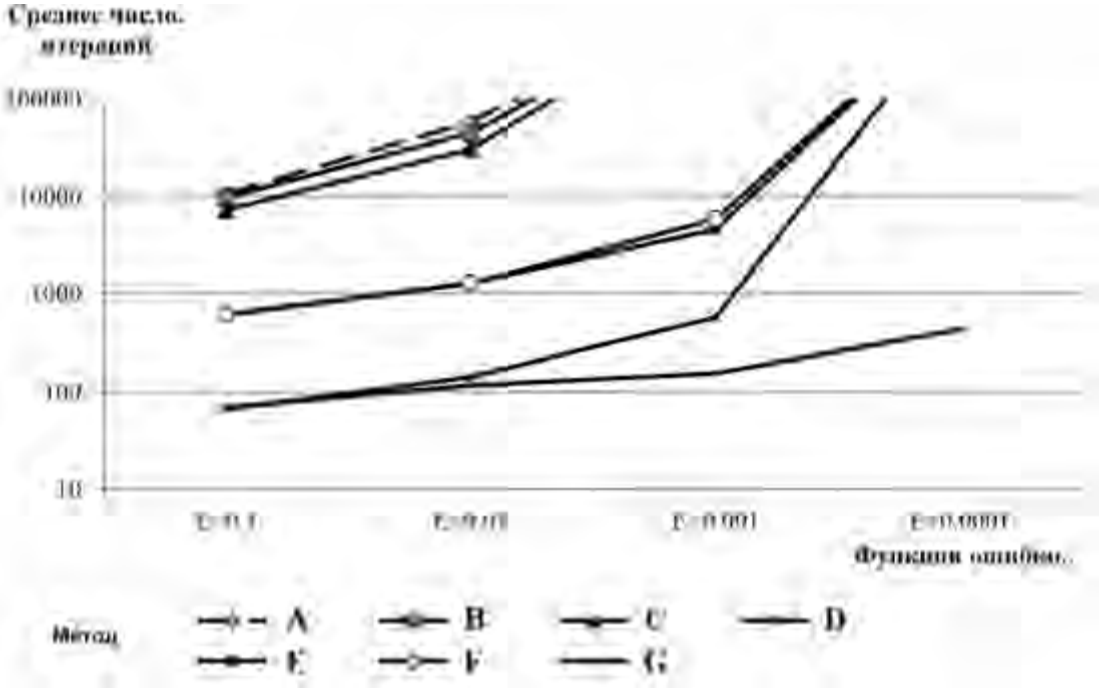


Рисунок 3.3 – Оценка скорости сходимости алгоритмов обучения при моделировании функции tan(x)

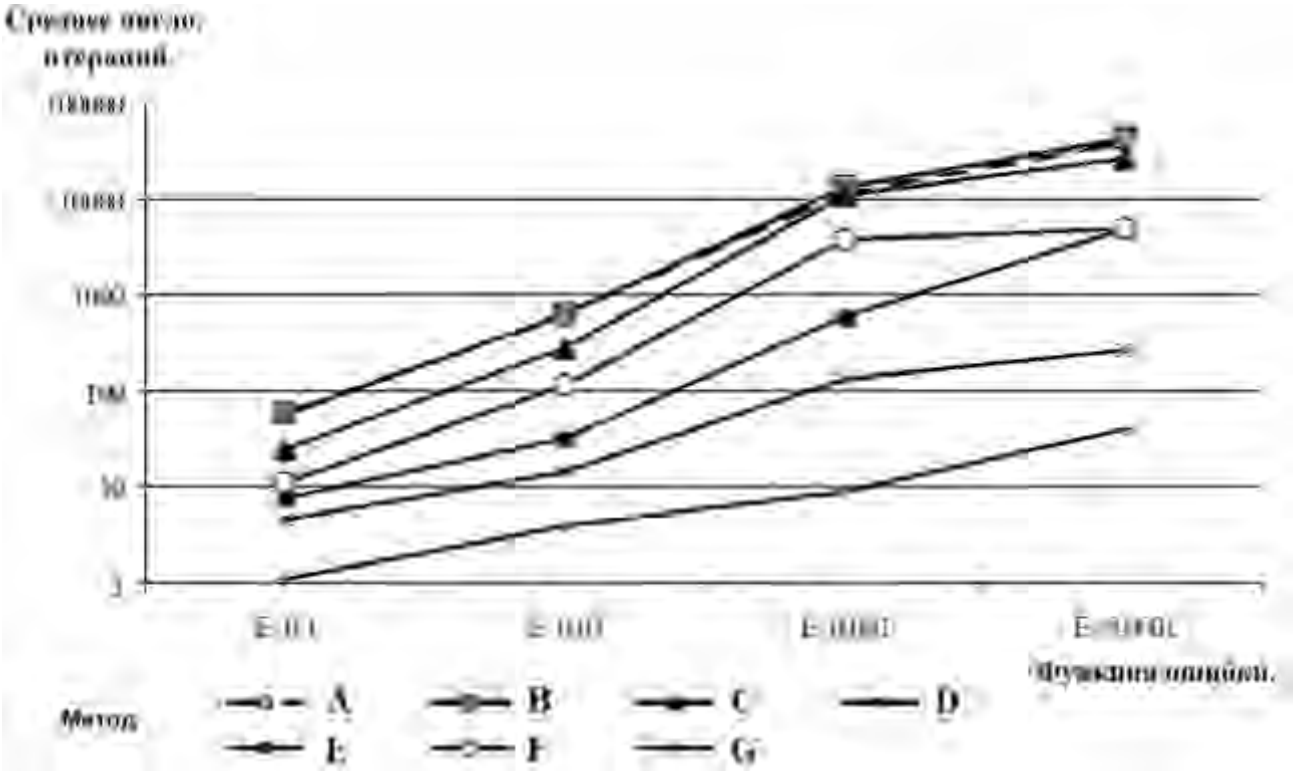


Рисунок 3.4 – Оценка скорости сходимости алгоритмов обучения при моделировании функции sin(x)cos(2x)

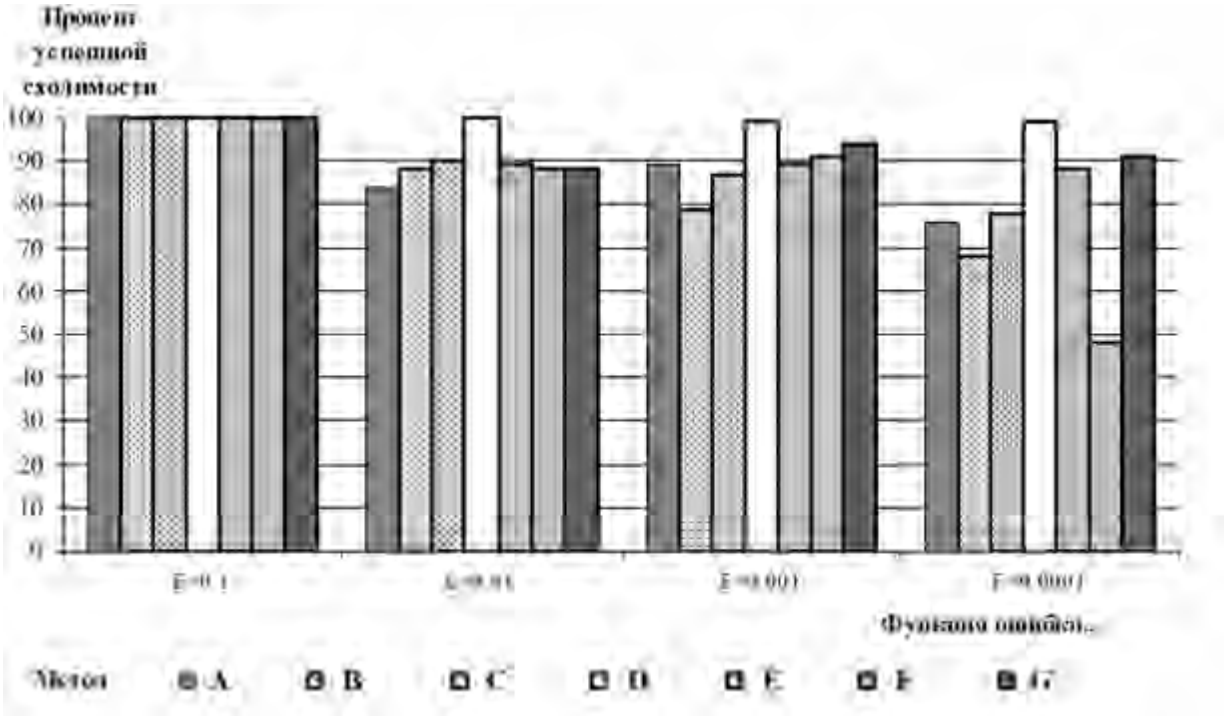


Рисунок 3.5 – Оценка глобальной надежности алгоритмов обучения при моделировании функции sin(x)

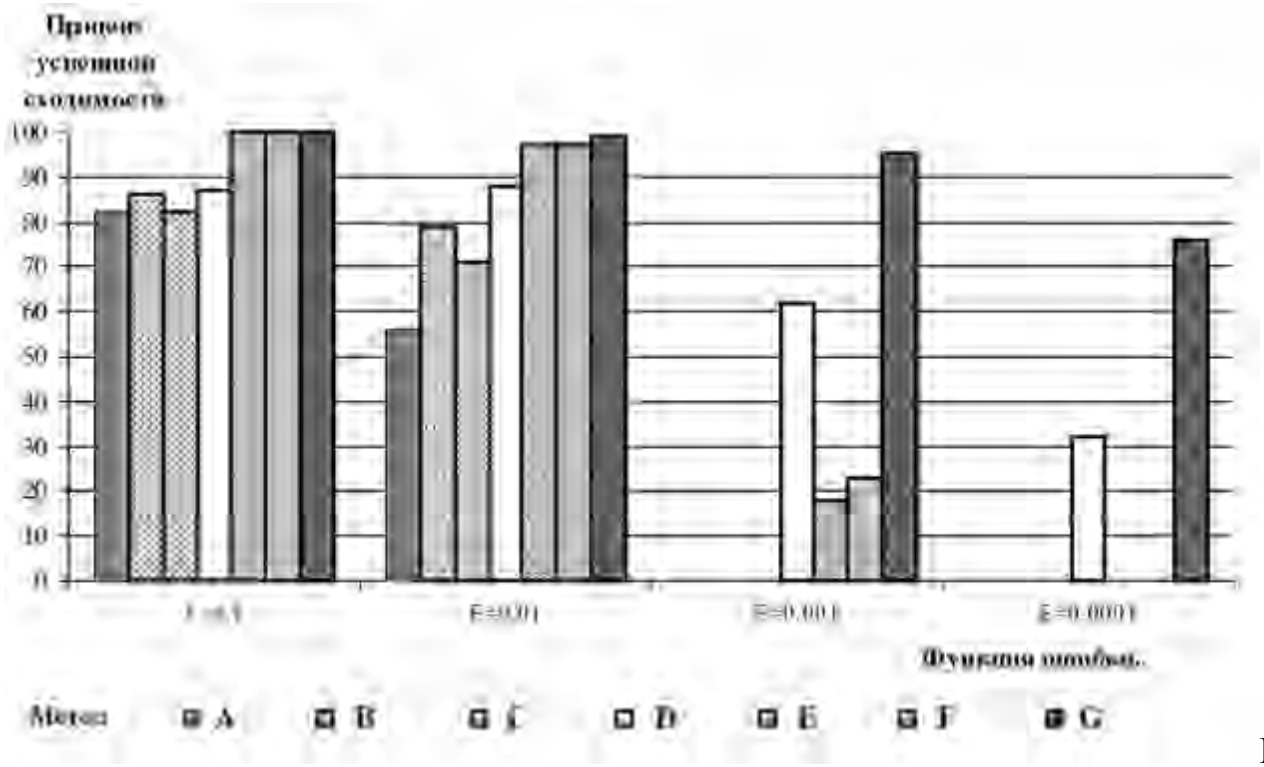


Рисунок 3.6 – Оценка глобальной надежности алгоритмов обучения при моделировании функции tan(x)

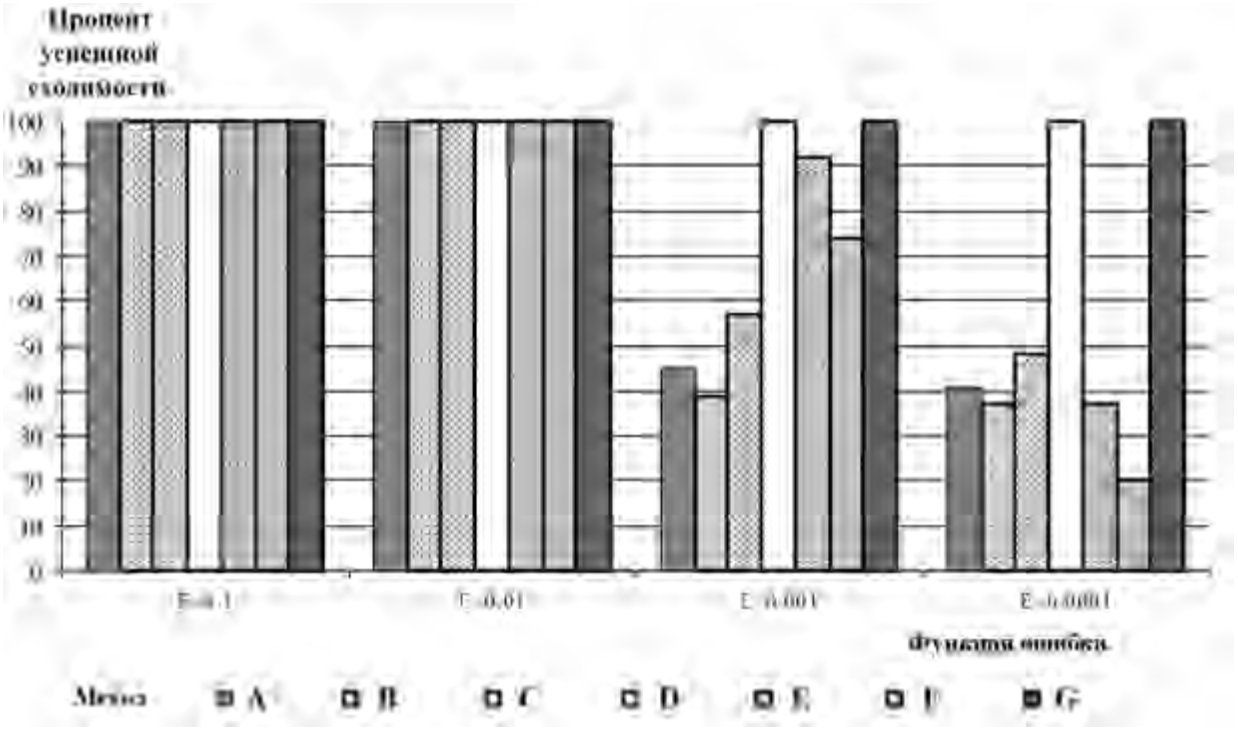


Рисунок 3.7 – Оценка глобальной надежности алгоритмов обучения при моделировании функции sin(x)cos(2x)

РАЗДЕЛ 4
РОБАСТНОЕ ОБУЧЕНИЕ ИНС ПРЯМОГО РАСПРОСТРАНЕНИЯ

4.1 Некоторые подходы, используемые при робастном обучении

Метод наименьших квадратов не является устойчивым, поскольку целевая функция может расти до бесконечности и выбросы могут стать доминирующими измерениями, которые фактически проверяют реальную модель. В качестве альтернативы, для обеспечения робастности целевую функцию модифицируют таким образом, чтобы ограничить влияние наибольших измерений. Основным следствием этого является, как правило, более низкая скорость сходимости алгоритмов оптимизации. Это объясняется тем, что очень сложно различать в первый раз выбросы и полезные измерения. В связи с этим, некоторые выбросы могут быть отфильтрованы, что приводит к уменьшению скорости сходимости. В наиболее трудном случае небольшие, но смещенные измерения перемещают минимум целевой функции.

4.1.1 Робастный медианный метод наименьших квадратов.

Робастный подход Least Median of Squares (LMedS) [221, 222] минимизирует следующую целевую функцию:

$$C(x) = \text{median}\left(r_1^2(x), r_2^2(x), ..., r_n^2(x)\right), \tag{4.1}$$

где n – количество измерений.

Метод медианы рассматривается как робастная оценка, так как он не учитывает 50% наибольших измерений. Тем не менее, он имеет два основных недостатка:

1) Если целевая функция не дифференцируема. Методы, основанные на минимизации ее градиента очень трудно реализовать.

2) Скорость сходимости алгоритмов минимизации может быть очень низкой, если остатки распределены таким образом, что у медианы очень слабый градиент.

Так в рассматриваемом в [221] теоретическом случае минимум правильно определялся даже при наличии 20% или 40% выбросов в измерениях. Данный метод имел точку «пробоя» при 50% выбросов (максимально возможную) и крайне медленную сходимость. Увеличение скорости сходимости можно достичь, во-первых, путем использования случайных выборок подмножеств измерений, и во-вторых, применением укороченного метода наименьших квадратов.

4.1.2 Укороченный метод наименьших квадратов.

Для улучшения скорости сходимости МНК в [221] предложен метод, известный как Least Trimmed Squares (LTS). Он заключается в минимизации суммы квадратов первых q измерений

$$C(x) = \sum_{k=1}^{q} r_k^2(x). \qquad (4.2)$$

В общем случае выбирается $q = n/2$, однако наличие априорной информации о количестве выбросов позволяет оптимизировать этот выбор. Целевая функция (4.2) аналогична методу LMedS, но градиент, как правило, выше. Хотя вклад q первых измерений делает целевую функцию гладкой, тем не менее, она остается недифференцируемой. Практическое исследование показало, что данный метод позволяет правильно находить минимум целевой функции при наличии 20% выбросов. Если процент выбросов возрастает до 40%, то поиск истинного минимума все еще остается возможным, несмотря на то, что в целевой функции появляются локальные минимумы. Возникновение этой проблемы зависит от распределения выбросов, что делает ее анализ весьма затруднительным. Следует отметить, что она

присуща и другим робастным методам (в том числе и LMedS). В общем случае, априори невозможно определить, какой робастный метод позволит избежать образования новых локальных минимумов целевой функции.

4.1.3 М-обучение.

Если информация о принадлежности помехи ξ некоторому определенному классу распределений известна, то путем минимизации оптимального критерия, представляющего собой взятый с обратным знаком логарифм функции распределения помехи, может быть получена оценка максимального правдоподобия (*M*-оценка). Если же такой информации нет, то для оценивания искомого вектора параметров θ следует применить какой-либо неквадратичный критерий, обеспечивающий робастность получаемой оценки [223-224].

В [152-153, 155] рассмотрены некоторые типы классов распределений, встречающиеся при решении практических задач: P_1 - класс невырожденных распределений, P_2 – класс распределений с ограниченной дисперсией, P_5 – класс финитных распределений (помеха ограничена по абсолютной величине, а какие-либо сведения о плотности ее распределения отсутствуют), P_3, P_4 и P_6 - классы приближенно нормальных, приближенно равномерных и приближенно финитных распределений соответственно, описываемых моделью Тьюки-Хьюбера [151, 225-226]

$$p(x) = (1-\varepsilon)p_0(x) + \varepsilon q(x), \tag{4.3}$$

где $p_0(x)$ – плотность соответствующего основного распределения; $q(x)$ – плотность засоряющего (произвольного) распределения; $\varepsilon \in [0,1]$ – параметр, характеризующий степень засорения основного распределения.

Даже если основное $p_0(x)$ и засоряющее $q(x)$ распределения являются гауссовскими с нулевыми математическими ожиданиями и дисперсиями σ_1^2

и σ_2^2, $\sigma_1^2 << \sigma_2^2$ соответственно, оценки, получаемые при выборе $\rho(e) = \frac{1}{2}e^2$, и являющиеся оптимальными для гауссовского распределения, будут неустойчивыми, так как $\rho'\big(e(i,\theta)\big) = e(i,\theta)$ растет линейно с увеличением $e(i,\theta)$.

Для всех этих классов были найдены наименее благоприятные, т.е. минимизирующие фишеровскую информацию, распределения. Так, минимум фишеровской информации для класса P_1 дает распределение Лапласа $p^*(\xi) = L(0, s_\xi)$, для класса P_2 – Гаусса $p^*(\xi) = N(0, \sigma^2)$, для классов финитных распределений наименее благоприятной плотностью является

$$p^*(\xi) = \begin{cases} \frac{1}{l}\cos^2\frac{\pi\xi}{2l} & \text{при } |\xi| \le l; \\ 0 & \text{при } |\xi| > l. \end{cases}$$

Для ε-засоренных вероятностных распределений (4.3) плотность распределения p^*, дающего минимум фишеровской информации, содержит некоторую центральную область $p = (1-\varepsilon)p_0(\xi)$ и хвосты с экспоненциально убывающей плотностью $p_0(\xi) = ce^{-\lambda|x|}$.

Использование этих распределений позволило получить нелинейные оценки огрубленного или робастного метода максимального правдоподобия, которые являются работоспособными практически для любых распределений помех. В этом случае алгоритм обучения является комбинированным: при $|e(k)| \le 3\sigma_1^2$ обучение происходит по РМНК, при $|e(k)| > 3\sigma_1^2$ - по рекуррентному методу наименьших модулей (РМНМ).

С другой стороны, выбор функции потерь, отличной от квадратичной, позволяет обеспечить робастность оценок, т.е. их работоспособность практически для всех распределений помех.

4.2 Выбор критерия робастного обучения

4.2.1 Оптимальные процедуры обучения.

Робастные оценки определяются из условия

$$\boldsymbol{w}(k) = \arg\min I_k(\boldsymbol{w}),\ \ I_k(\boldsymbol{w}) = \sum_{i=1}^{k} F\left[e\left(\boldsymbol{w}, \boldsymbol{x}(k)\right)\right], F(e) = -\ln\ p(\xi)\big|_{\xi=e} \qquad (4.4)$$

и являются оптимальными в минимаксном смысле на соответствующих классах распределений.

Оптимальная функция потерь, т.е. функция, минимизирующая асимптотическую матрицу ковариации ошибок оценивания, равна логарифму плотности распределения помехи при $\xi = e(\boldsymbol{x}, \boldsymbol{w})$, взятой с обратным знаком

$$F_0[e(\boldsymbol{x}, \boldsymbol{w})] = -\ln\ p_0(\xi)\big|_{\xi=e(\boldsymbol{x},\boldsymbol{w})},$$

т.е. логарифмической функции неправдоподобия.

Таким образом, оптимальной функцией потерь для класса P_1 будет модульная, для класса P_2 – квадратичная, для класса P_5

$$F_0[e(\boldsymbol{x}, \boldsymbol{w})] = -\ln\cos\left(\frac{\pi e}{c}\right).$$

Им соответствуют кривые III, I и II на рис.4.1-а).

Для ε-засоренных вероятностных распределений F_0 является нелинейной на некотором интервале, определяемом параметрами помех, и линейной вне этого интервала.

В силу того, что модульный критерий позволяет получить оценку, менее чувствительную к хвостам распределения помехи, чем МНК-оценка,

представляют интерес такие разновидности модульного критерия, как функционал А.Форсайта (рис. 4.1-б)

$$F[e(k)] = |e(k)|^{\lambda}, \tag{4.5}$$

и функционал вида (рис.4.1-в)

$$F[e(k)] = -arctg|e(k)|^{\lambda}, \tag{4.6}$$

где $0 < \lambda < 2$.

Кривые, соответствующие функционалам (4.5) и (4.6) с $\lambda = 1$ обозначены на рис.4.1 римскими цифрами III и V. Кроме того, на том же рисунке показана кривая IV, соответствующая функционалу Д.Эндрюса

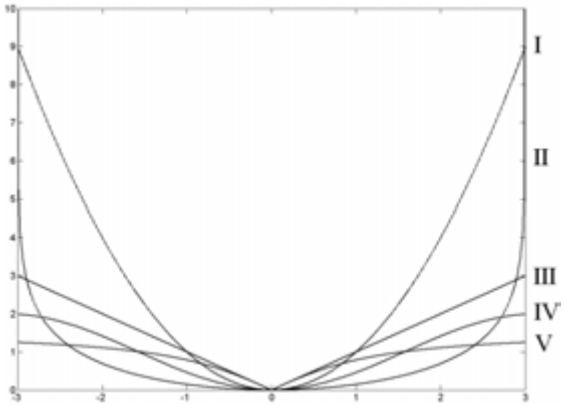
$$F[e(k)] = \begin{cases} 1 - \cos\left(\frac{e}{c}\right), |e| \le \pi c; \\ 0, |e| > \pi c, \end{cases} \tag{4.7}$$

и занимающая некоторое промежуточное положение между III и IV.

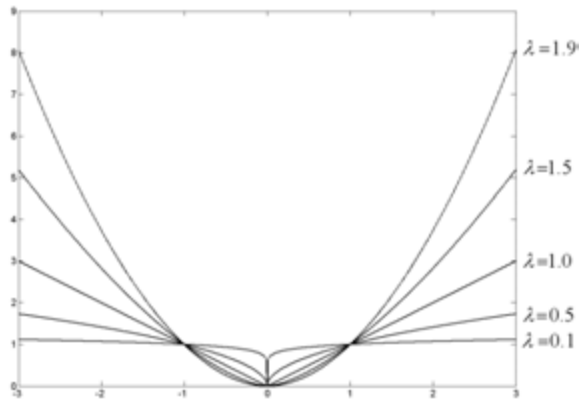
Как следует из (4.7), при построении оценок Эндрюса наблюдениям, ошибки которых велики по абсолютной величине, присваиваются нулевые веса.

4.2.2 Традиционные критерии М-обучения.

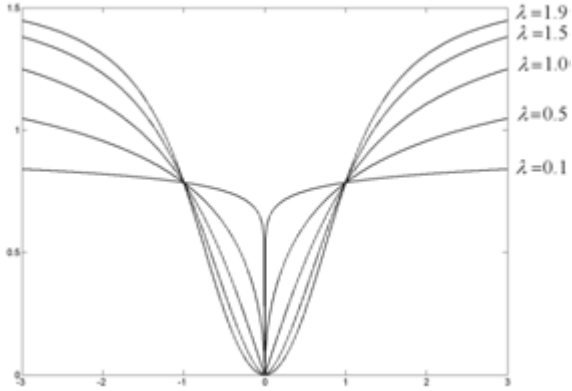
Существует достаточно большое количество функционалов, обеспечивающих получение робастных М-оценок (см., например [56, 151-153, 157, 226-232]), однако наиболее распространенными являются комбинированные функционалы, предложенные Хьюбером [151] и Хемпелем [227] и состоящие из квадратичного, обеспечивающего оптимальность оценок для гауссовского распределения, и модульного, позволяющего получить более робастную к распределениям с тяжелыми «хвостами»



а)



б)



в)

Рисунок 4.1 – Кривые, соответствующие функционалам (4.5)-(4.7)

(выбросами) оценку. Данные функционалы и их функции влияния ψ, имеющие соответственно вид

$$\rho_1(e)=\begin{cases}\dfrac{e^2}{2}, & |e|\le c;\\ c|e|-\dfrac{c^2}{2}, & c<|e|;\end{cases}\qquad \psi_1(e)=\begin{cases}e, & |e|\le c;\\ csign(e), & c<|e|;\end{cases}$$

$$\psi_1'(e)=\begin{cases}1, & |e|\le c;\\ 0, & c<|e|;\end{cases}\qquad \omega_1(e)=\begin{cases}1, & |e|\le c;\\ \dfrac{c}{e}, & c<|e|;\end{cases} \tag{4.8}$$

$$\rho_2(e)=\begin{cases}\dfrac{e^2}{2},\ 0\le|e|<b;\\ b|e|-\dfrac{b^2}{2},\ b\le|e|<c;\\ \dfrac{b}{2}(c+d)-\dfrac{b}{2}\left(b-\dfrac{(|e|-d)^2}{c-d}\right),\ c\le|e|<d;\\ \dfrac{b}{2}(c+d)-\dfrac{b^2}{2},\ d\le|e|;\end{cases}\qquad \psi_2(e)=\begin{cases}e,\ 0\le|e|<b;\\ bsign(e),\ b\le|e|<c;\\ \dfrac{b(|e|-d)}{c-d}sign(e),\ c\le|e|<d;\\ 0,\ d\le|e|;\end{cases}$$

$$\psi_2'(e)=\begin{cases}1,\ 0\le|e|<b;\\ 0,\ b\le|e|<c;\\ \dfrac{b}{c-d},\ c\le|e|<d;\\ 0,\ d\le|e|;\end{cases}\qquad \omega_2(e)=\begin{cases}1,\ 0\le|e|<b;\\ \dfrac{b}{|e|},\ b\le|e|<c;\\ \dfrac{b(|e|-d)}{(c-d)|e|},\ c\le|e|<d;\\ 0,\ d\le|e|;\end{cases}\qquad (4.9)$$

приведены на рис. 4.2.

Обычно М-оценки описываются путем задания функции влияния ψ, а не функции ρ.

Функция ψ, предложенная Хьюбером, является монотонной, а ψ-функция Хемпела – немонотонной. Как отмечается в [233], при наличии в распределении тяжелых хвостов, применение немонотонных ψ- функций позволяет улучшить результаты оценивания.

Эффективность применения этих функционалов зависит от того, насколько удачно выбраны входящие в них константы *a*, *b*, *c* и *d*, которые и определяют степень помехоустойчивости. В указанных выше работах рекомендуется выбирать значения a из интервала $[\sigma,2\sigma]$, где σ - стандартное отклонение наблюдения $\boldsymbol{x}$, а значения *b*, *c* и *d* задавать равными 1.5, 3.5 и 8 соответственно.

Модификацией хуберовский оценок являются оценки Мэллоуза [234]. В них наряду со взвешиванием остатков ε производится взвешивание факторов, что позволяет уменьшить влияние точек, резко выделяющихся в

пространстве независимых переменных. Для нахождения оценок Мэллоуза необходимо решить $\rho+1$ уравнений

$$\psi(\varepsilon)=\begin{cases}\sum_{i=1}^{n}\psi(\varepsilon[\mathrm{i}]);\\ \sum_{i=1}^{n}x_m[i]\psi^x(x[\mathrm{i}])\psi(\varepsilon[\mathrm{i}]),\ \mathrm{m}=0,1,..,\rho,\end{cases} \tag{4.10}$$

где ψ определяется по Хьюберу, а

$$\psi^x(x[\mathrm{i}])=\prod_{j=1}^{\rho}\psi^{(j)}(x[\mathrm{i}]). \tag{4.11}$$

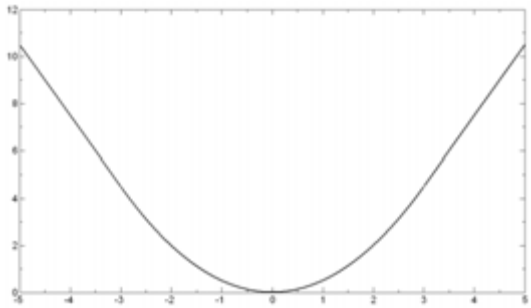
Упоминавшиеся выше оценки Форсайта близки к хьюберовским, однако, не имеют столь убедительного теоретического обоснования. Эмпирически показано, что приемлемым является α = 1.5. При α = 2 оценки Форсайта совпадают с оценками МНК, а при α = 1 получаем метод наименьших абсолютных отклонений (модулей), минимизирующий функционал

$$Q=\sum_{i=1}^{n}|\varepsilon[\mathrm{i}]|. \tag{4.12}$$

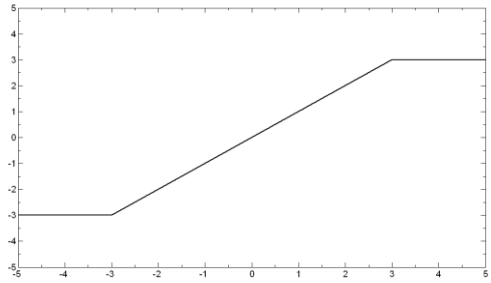
Для получения оценок Меррила-Швеппа используется функционал

$$\rho(\varepsilon)=\begin{cases}\gamma_1\varepsilon^2, |\varepsilon|\le k\\ \gamma_2\sqrt{\varepsilon}, |\varepsilon|>k\end{cases},$$

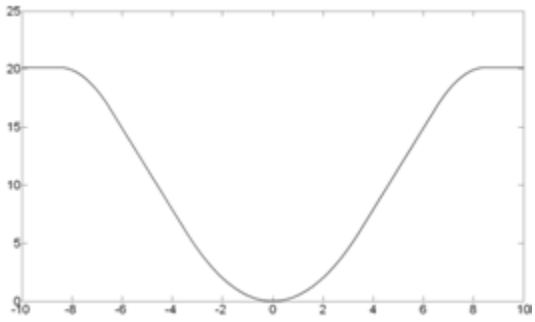
где $\gamma_1>0, \gamma_2>0, 0\le k$.



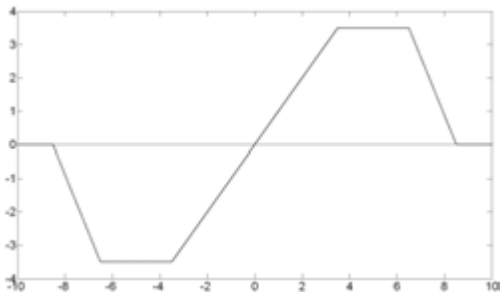
а) $\rho_1(e)$



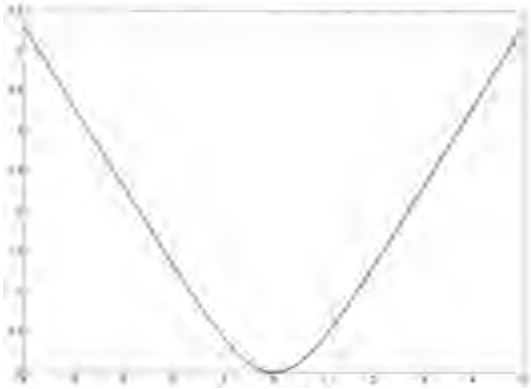
б) $\psi_1(e)$



в) $\rho_2(e)$

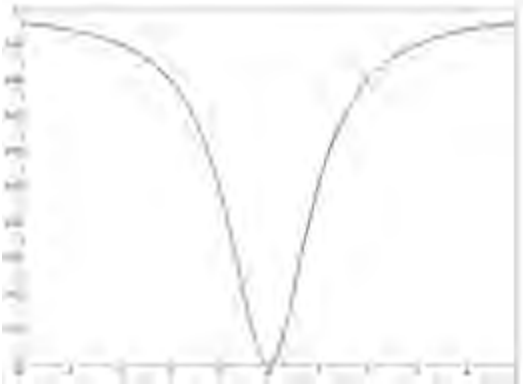


г) $\psi_2(e)$



д) $\rho_3(e)$

е) $\psi_3(e)$



ж) $\rho_4(e)$

з) $\psi_4(e)$

Рисунок 4.2 – Функционалы и их функции влияния

Отметим, что для обеспечения робастных свойств получаемых оценок достаточно эффективным является применение комбинированного функционала обучения [235]

$$F[e(k)] = \lambda M\left\{e^2(k)\right\} + (1-\lambda)M\left\{\left|e(k)\right|\right\}, \tag{4.13}$$

где $\lambda \in [0,1]$.

Кроме функций (4.8), (4.9), достаточно эффективными являются, например, следующие, также приведенные на рис.4.2 [157]:

$$\rho_3(e) = c\ln\cosh\left(\frac{e}{c}\right); \psi_3(e) = \tanh\left(\frac{e}{c}\right);$$

$$\psi_3'(e(k)) = \frac{1}{c^2}\left(1-\tanh^2\left(\frac{e(k)}{c}\right)\right);\ \ \omega_3(e(k)) = \frac{1}{ce(k)}\tanh\left(\frac{e(k)}{c}\right); \tag{4.14}$$

$$\rho_4(e) = \frac{e^2}{c^2+e^2};\ \ \psi_4(e) = \frac{2c^2e}{\left(c^2+e^2\right)^2};$$

$$\psi_4'(e(k)) = \frac{2c^4 - 6c^2e^2(k)}{\left(c^2+e^2(k)\right)^3};\ \omega_4(e(k)) = \frac{2c^2}{\left(c^2+e^2(k)\right)^2}. \tag{4.15}$$

Классические робастные методы, минимизирующие как (4.8-4.9), (4.14-4.15), так и остальные неквадратичные функционалы, ориентированы на симметричность засорения, когда выбросы одинаково часто появляются как в области отрицательных, так и в области положительных значений.

Указанные методы позволяют эффективно бороться с помехами, описываемыми моделью (4.3), когда основное и засоряющее распределения являются гауссовскими с нулевыми математическими ожиданиями и дисперсиями σ_1^2 и σ_2^2, $\sigma_1^2 << \sigma_2^2$.

В более общей ситуации произвольного вида засорения, например, когда гауссовское засоряющее распределение имеет ненулевое математическое

ожидание или когда засоряющее распределение является несимметричным, оценки, даваемые этими методами, будут смещенными. Необходимость учета асимметрии распределений обусловливает целесообразность выбора асимметричных функционалов.

4.2.3 Выбор критерия обучения при несимметричном распределении.

Принципиально могут быть получены асимметричные функционалы (АФ) различных видов. При этом базой для них служат соответствующие традиционные симметричные функционалы робастного М-обучения. Наличие информации о виде асимметричного распределения данных и помех является основой для выбора параметров функционалов, оказывая значительное влияние на вид АФ.

Так, например, в задачах интерполяции, оценивания и имитационного моделирования, связанных с пространственным распределением признаков, часто возникает необходимость учета левой асимметрии распределения исходных данных. Обычно в таких задачах данные имеют логнормальное распределение и в этом случае оптимальной будет функционал следующего вида [236]:

$$F(e)=\begin{cases}\dfrac{1}{e\sigma\sqrt{2\pi}}\exp\left(\dfrac{\left(\ln e-\mu\right)^2}{2\sigma^2}\right), & \textit{если } e>0;\\ 0, & \textit{если } e\le 0,\end{cases} \qquad (4.16)$$

где μ - параметр масштаба (медиана); σ - параметр формы (дисперсия).

В экономике и менеджменте, теории и практике надежности и испытаний, в различных областях техники и метеорологии и т.д. широко применяются гамма распределения. Им подчинены во многих ситуациях такие величины, как общий срок службы изделия, время наработки до k-го отказа. Гамма-распределение является одним из наиболее распространенных в статистической теории радиотехники. В частности, такое распределение

имеют выходные эффекты типовых РТ систем обработки сигналов при приеме гауссовских процессов, многочастотных сигналов, излучений сложных радиоисточников и т.д. Оптимальный функционал для работы с подобными системами будет иметь вид:

$$F(e)=\begin{cases}\dfrac{1}{\sigma\Gamma(\alpha)}\left(\dfrac{e}{\sigma}\right)^{\alpha-1}\exp\left(-\dfrac{e}{\sigma}\right), \ \textit{если}\ e\geq 0;\\ 0, \ \textit{если}\ e<0,\end{cases} \tag{4.17}$$

где $\Gamma(\alpha)$ - гамма-функция Эйлера, $\sigma,\alpha>0$.

Довольно часто в реальных системах встречаются помехи, представляющие собой смесь нормальной помехи и помехи с распределением Релея.

На рис.4.3-а) приведена гистограмма помехи, описываемой моделью (4.3) с $\varepsilon=0.1$ и представляющей собой смесь нормальной помехи $N(0;0.6)$ и помехи с распределением Релея Ray (4.8), и соответствующие АФ Хемпеля (4.9) – рис.4.3-б). На рис.4.3-в) и 4.3-г) представлены асимметричные модификации функционалов (4.14)-(4.15) соответственно, которые были получены с помощью следующего выражения:

$$\rho_i(e)=\begin{cases}\rho_i(e), \ \text{если}\ -3.5\leq e\leq 1.8;\\ const, \ \text{в противном случае.}\end{cases}$$

Возможен и другой, более общий, вид асимметричности функционалов, примеры которых, построенных на основе (4.8)-(4.9), а также их первых, вторых производных и весовых функций приведены на рис.4.4. На данном рисунке вместо функции Хемпеля (4.9) показана ее модификация, полученная из (4.9) при $c = d$ (подобная ψ-функция использовалась в работе [237]). Коэффициент c задавался следующим образом:

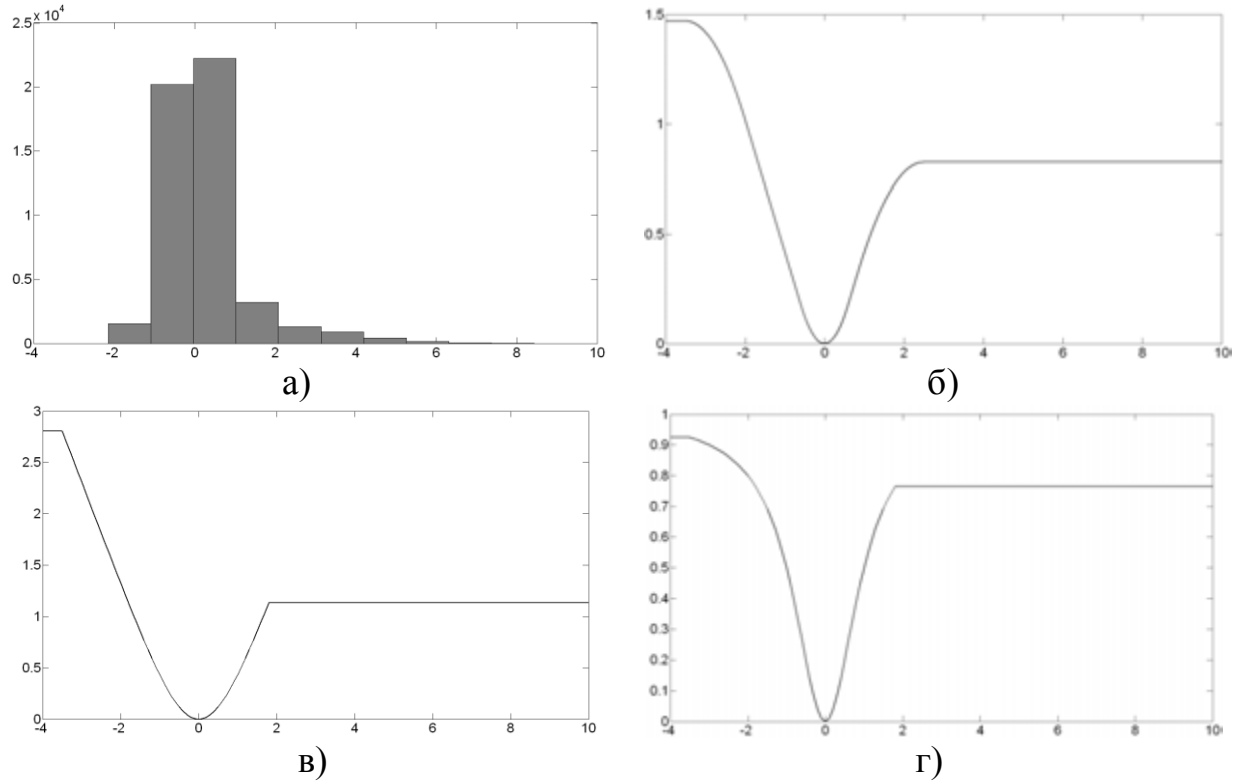


Рисунок 4.3 – Гистограмма помехи и асимметричные модификации функционалов

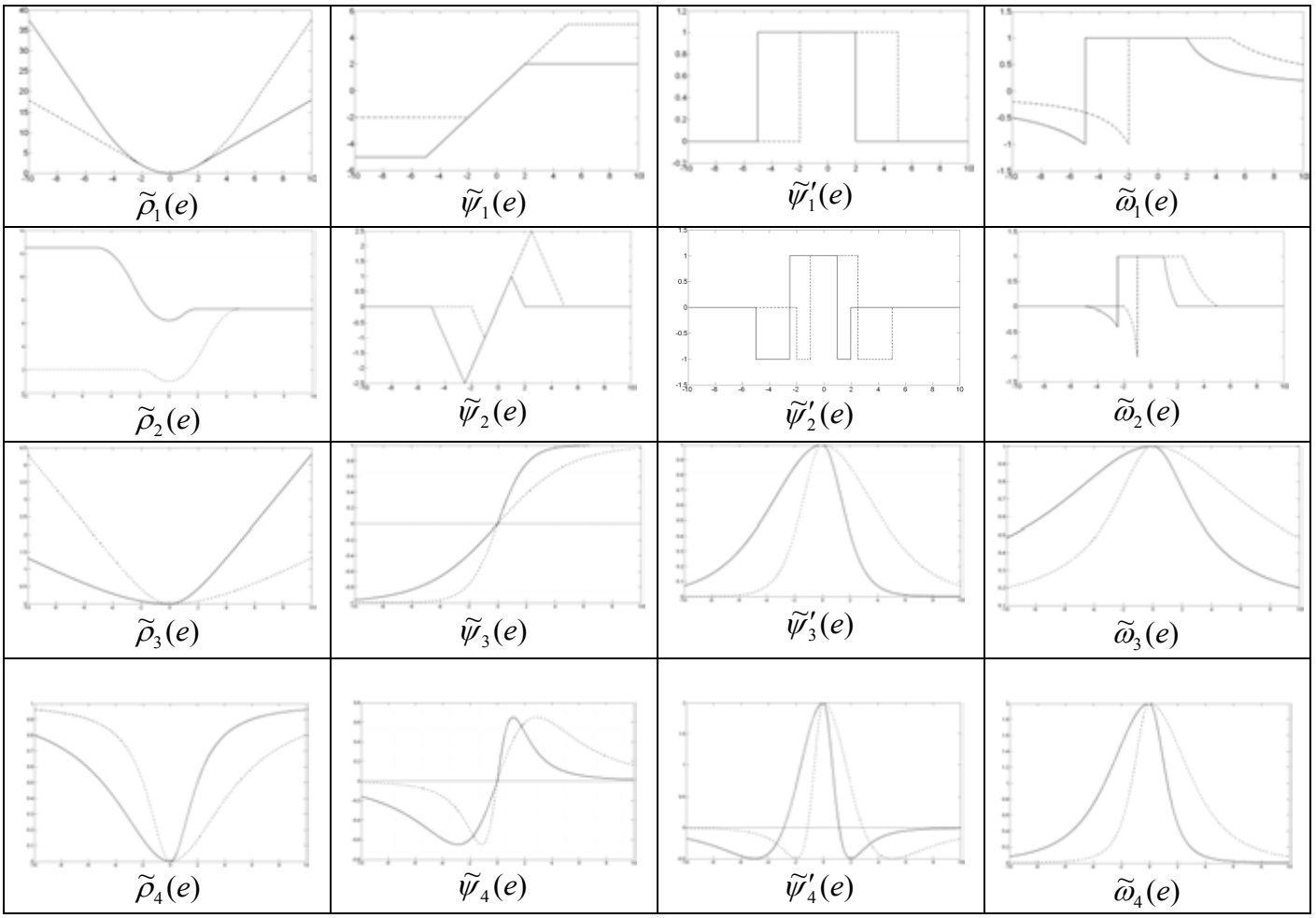


Рисунок 4.4 – Примеры асимметричности функционалов

$$c = \begin{cases} c_1, если\ \zeta \le 0; \\ c_2 \text{ в противном случае.} \end{cases}$$

На всех рисунках кривая, обозначенная пунктиром, соответствует случаю $c_1 < c_2$, сплошной линией – $c_1 > c_2$.

4.3 Процедуры робастного обучения

4.3.1 Оптимальные процедуры обучения.

Если плотность распределения помех $p_0(\xi)$ полностью не известна, а известно лишь, что она принадлежит некоторому классу $p_0(\xi) \in p_k$, то для наименее благоприятной на этом классе плотности распределения $p^*(\xi)$ можно получить соответствующий оптимальный алгоритм. Если $p_0(\xi) = p^*(\xi)$ этот алгоритм будет абсолютно оптимальным для класса p_k.

Оптимальные алгоритмы обучения сети представляют собой градиентные процедуры вида

$$\theta(k) = \theta(k-1) - H(k)\nabla F_0\left[e(k)\right], \qquad (4.18)$$

где $H(k)$ – матрица усиления.

Учитывая то, что $\nabla F_0\left[e(k)\right] = F'\left[e(k)\right]\nabla f(k)$, где

$$\nabla f(k) = \Big[1,\ \Phi_1(\mathrm{x}(k)),\ 2\Phi_1(\mathrm{x}(k))c_1\sigma_1^{-2}(\mathrm{x}(k) - \mu_1)^T,\ 2\Phi_1(\mathrm{x}(k))c_1\sigma_1^{-3}\left\|\mathrm{x}(k) - \mu_1\right\|^2,$$

$$\ldots, \Phi_{\mathrm{N}}(\boldsymbol{x}(k)), 2\Phi_{\mathrm{N}}(\boldsymbol{x}(k))c_N\sigma_N^{-2}(\boldsymbol{x}(k) - \boldsymbol{\mu}_N)^T, 2\Phi_{\mathrm{N}}(\boldsymbol{x}(k))c_N\sigma_{\mathrm{N}}^{-3}\left\|\boldsymbol{x}(k) - \boldsymbol{\mu}_N\right\|^2 \Big]^T,$$

можно записать оптимальные алгоритмы обучения следующим образом:

$$\theta(k) = \theta(k-1) + \frac{P(k-1)\nabla f(k)}{\gamma + \nabla f^T(k)P(k-1)\nabla f(k)} F'\left[e(k)\right], \qquad (4.19)$$

$$P(k) = P(k-1) - \frac{P(k-1)\nabla f(k)\nabla f^T(k)P(k-1)}{\gamma + \nabla f^T(k)P(k-1)\nabla f(k)}, \tag{4.20}$$

где

$$F'[e(k)] = \begin{cases} e(k) \text{ для } p_0(\xi) = N(0,\sigma_\xi)^2; & (4.21) \\ sign(e(k)) \text{для } p_0(\xi) = L(0,s_\xi); & \\ \dfrac{2e(k)}{s_\xi^2 + e^2(k)} \text{ для } p_0(\xi) = C(0,s_\xi); & (4.22) \end{cases}$$

$$\gamma = \begin{cases} 1 \text{ для } p_0(\xi) = N(0,\sigma_\xi)^2; & (4.23) \\ s_\xi \text{ для } p_0(\xi) = L(0,s_\xi); & \\ s_\xi^2 \text{ для } p_0(\xi) = C(0,s_\xi). & (4.24) \end{cases}$$

Как следует из приведенных соотношений, если алгоритм МНК (4.19)-(4.21), (4.23) инвариантен к параметру масштаба нормального распределения σ_ξ^2, то оценки МНМ (4.19), (4.20), (4.22), (4.24) и оценка, соответствующая распределению Коши зависят от параметра s_ξ. При неизвестной величине s_ξ в этих алгоритмах следует использовать оценку $\hat{s}$, получаемую, например, с помощью алгоритма стохастической аппроксимации.

Для ε-засоренных вероятностных распределений оптимальные алгоритмы обучения представляют собой комбинированные процедуры вида

$$\theta(k) - \theta(k-1) = \begin{cases} -H_1(k)F'[e(k)]\nabla f(k) \text{ при } |e(k)| \le \Delta; \\ -H_2(k)\nabla f(k)\operatorname{sign} e(k) \text{ при } |e(k)| > \Delta, \end{cases} \tag{4.25}$$

где $H_1(k)$, $H_2(k)$ – матрицы усиления.

4.3.2 Одношаговые робастные процедуры обучения.

В робастной идентификации и фильтрации достаточно широкое распространение получили релейные алгоритмы [202] или алгоритмы,

использующие нелинейное преобразование ошибки обучения [119]. Основу данных процедур, которые могут быть использованы для обучения линейной Адалины, составляет следующая

$$\theta(k) = \theta(k-1) + \gamma(k)x(k)\operatorname{sign} e(k). \tag{4.26}$$

Очевидно, что с целью ускорения скорости оценивания могут быть предложены следующие модификации (4.26):

$$\theta(k) = \theta(k-1) + \gamma(k)\frac{\operatorname{sign} e(k)}{\|x(k)\|^2}x(k), \tag{4.27}$$

$$\theta(k) = \theta(k-1) + \gamma(k)\frac{\operatorname{sign} e(k)}{r(k)}x(k), \tag{4.28}$$

где $r(k)$ определяется как в (3.11).

Данные процедуры минимизируют модульный критерий. При использовании комбинированного критерия (4.13) градиентная процедура минимизации имеет вид

$$\theta(k) = \theta(k-1) + \gamma(k)\left[\lambda 2e(k) + (1-\lambda)\operatorname{sign} e(k)\right]x(k). \tag{4.29}$$

Данная процедура сочетает свойства МНК со свойствами МНМ, т.к. при $\lambda = 1$ из (4.29) следует алгоритм МНК, а при $\lambda = 0$ - алгоритм МНМ, и позволяет бороться с импульсными помехами. Варьируя параметр λ, можно изменять свойства алгоритма.

Свойства процедуры (4.29) исследованы в Приложении Е.

Как показано в Приложении Е, процедура (4.29) будет сходиться при выполнении условия

$$0 < \gamma < \frac{2\lambda + (1-\lambda)\beta}{6\lambda\left(\lambda + (1-\lambda)\beta\right)\sigma_x^2}. \tag{4.30}$$

Наличие помехи $\xi(k)$ приводит к тому, что процедура (4.29) будет сходиться к области, определяемой этой помехой, и равной

$$\lim_{k\to\infty} M\left\{\left\|\theta(k-1)\right\|^2\right\} = \gamma^2 (1-\lambda)^2 N\sigma_\xi^2\sigma_x^2. \tag{4.31}$$

Из полученного выражения видно, что для обеспечения $\lim_{k\to\infty} M\left\{\left\|\tilde{\theta}(k-1)\right\|^2\right\} = 0$ параметр γ должен выбираться переменным и с ростом *n* стремиться к нулю, т.е. удовлетворять условиям Дворецкого [184].

Рассмотренные в предыдущем разделе градиентные одношаговые процедуры обучения могут использоваться и для робастного обучения, если в качестве минимизируемого выбрать какой-либо неквадратичный функционал.

Обобщением процедур (4.26)-(4.28), которые могут быть использованы для обучения нелинейной Адалины, являются следующие:

$$\theta(k) = \theta(k-1) + \gamma(k)\nabla f(k)\operatorname{sign} e(k). \tag{4.32}$$

$$\theta(k) = \theta(k-1) + \gamma(k)\frac{\operatorname{sign} e(k)}{\left\|\nabla f(k)\right\|^2}\nabla f(k), \tag{4.33}$$

$$\theta(k) = \theta(k-1) + \gamma(k)\frac{\operatorname{sign} e(k)}{r(k)}\nabla f(k). \tag{4.34}$$

Процедура (4.29) также может быть модифицирована по аналогии с вышеизложенным, например,

$$\theta(k)=\theta(k-1)+\gamma(k)\left[\lambda 2e(k)-(1-\lambda)\operatorname{sign}e(k)\right]z(k), \qquad (4.35)$$

где $z(k)=\nabla f(k)\left\|\nabla f(k)\right\|^{-2}$ - для нормализованной процедуры;

$z(k)=\nabla f(k)\left(\left\|\nabla f(k)\right\|^{2}+\delta\right)^{-1}$ - для регуляризованной процедуры;

$z(k)=\nabla f(k)r^{-1}(k)$ - для модификации, основанной на стохастической аппроксимации.

Кроме того, если в качестве $z(k)$ взять вектор $\operatorname{sign}\nabla f(k)$, то можно получить и другие модификации.

Следует, однако, отметить, что при использовании всех этих модификаций возникает проблема выбора оптимальных значений параметров $\gamma(k)$, δ, λ. Трудность решения этой проблемы существенно зависит от наличия информации о статистических свойствах полезных сигналов и помех.

4.3.3 Многошаговые робастные процедуры обучения.

Рассмотренные выше процедуры обучения (4.19)-(4.20) будут работоспособными после $k \geq N+1$ тактов. Если же $k < N+1$, обучение сети может осуществляться с помощью проекционных алгоритмов [238]. Следует также отметить, что применение этих процедур целесообразно и в случае, когда исследуемый объем является нестационарным, и параметры его изменяются с течением времени.

s-шаговые проекционные процедуры обучения с $s < N$ могут быть получены по аналогии с процедурами, рассмотренными в подразделе 3.3.3. Данные процедуры могут быть описаны соотношениями:

$$\theta(k)=\theta(k-1)+\gamma(k)\nabla f_s(k)\left[\nabla f_s^{T}(k)\,\nabla f_s(k)\right]^{-1}F_s'[e(k)], \qquad (4.36)$$

где $F_s'[e(k)]=(F'[e(k)],0,0,...,0)^{T}$;

$\nabla f_s(k) = (\nabla f(k), \nabla f(k-1), ..., \nabla f(k-s-1))$ – матрицы $(N+1)$xs; $\gamma(k)$ – параметр, удовлетворяющий условиям Дворецкого.

Более удобная в вычислительном отношении процедура, использующая рекуррентное вычисление матрицы $[\nabla f_s^T(k)\, \nabla f_s(k)]^{-1}$ вместо её непосредственного обращения может быть получена по аналогии с рассмотренным в разделе 3 подходом, т.е. с учетом правил формирования матриц $\nabla f_s(k-1) = (\nabla f_{s-1}(k-1) \vdots \nabla f(k-s))$, $\nabla f_s(k) = (\nabla f(k) \vdots \nabla f_{s-1}(k-1))$.

В этом случае можно записать

$$\theta(k) = \theta(k-1) + \gamma(k) \frac{R_{s-1}(k-1)}{\nabla f^T(k)\, R_{s-1}(k-1)\, \nabla f(k)}\, F_s'[e(k)], \qquad (4.37)$$

где $R_{s-1}(k-1) = I - \nabla f_{s-1}(k-1)\, [f_{s-1}^T(k-1)\, \nabla f_{s-1}(k-1)]^{-1}\, \nabla f_{s-1}^T(k-1) =$ $= I - f_{s-1}^+(k-1)\, f_{s-1}^T(k-1)$. Здесь $f_{s-1}^+(k-1)$ – матрица, псевдообратная к $f_{s-1}(k-1)$.

Матрица $R_s(k)$ также вычисляется рекуррентно. Если получение соотношений для вычисления этой матрицы при поступлении новой информации достаточно тривиально, то для реализации процедуры сброса следует воспользоваться теоремой Гревиля и следующими соотношениями между матрицами $\nabla f_{s+1}^+(k)$ и $\nabla f_s^+(k)$ [239]:

$$(\nabla f_s^+(k) \vdots 0) = \nabla f_{s+1}^+(k)(I - g\, K^T(k)),$$

где для рассматриваемого случая $s < N$

$$K(k) = \frac{(I - \nabla f_{s+1}(k)\, \nabla f_{s+1}^+(k))\, \mathrm{g}}{\mathrm{g}^T (I - \nabla f_{s+1}(k)\, \nabla f_{s+1}^+(k))\, \mathrm{g}},$$

$g = (0, 0, ..., 1)^T$ - вектор sx1.

Проводя несложные вычисления, получим, что при поступлении новой информации на *k*-ом такте матрица $R_s(k)$ пересчитывается по формуле

$$R_s(k) = R_{s-1}(k-1) + \frac{R_{s-1}(k-1)\,\nabla f(k)\,\nabla f^T(k)\,R_{s-1}(k-1)}{\nabla f^T(k)\,R_{s-1}(k-1)\,\nabla f(k)}, \qquad (4.38)$$

а при сбросе устаревшей информации об (*k*–*s*–1)-м такте – по формуле

$$R_s(k+1) = R_{s+1}(k+1) + \frac{\nabla \hat{f}(k-s+1)\,\nabla \hat{f}^T(k)}{\left\|\nabla \hat{f}(k-s+1)\right\|^2}, \qquad (4.39)$$

где $\nabla \hat{f}(k-s+1) = U_{s+1}(k+1)\,\nabla f(k-s+1)$; $R_0 = I$;

$U_{s+1}(k+1) = (\nabla f_{s+1}^T(k+1)\,\nabla f_{s+1}(k+1))^+$.

Входящая в (4.38), (4.39) матрица $U_s(k)$ в свою очередь при поступлении новой информации и сбросе устаревшей вычисляется соответственно по формулам:

$$\begin{aligned} U_{s+1}(k+1) = U_s(k) &- \frac{(U_s(k)+R_s(k))\,\nabla f(k)\,\nabla f^T(k)\,U_s(k)}{\nabla f^T(k+1)\,U_s(k)\,\nabla f(k+1)} - \\ &+ \frac{R_s(k)\,\nabla f(k+1)\,\nabla f^T(k+1)\,R_s(k)}{\nabla f^T(k+1)\,U_s(k)\,\nabla f(k+1)} \times \\ &\times \frac{R_s(k)\,\nabla f(k+1)\,\nabla f^T(k+1)\,U_s(k)\,\nabla f(k)\;\nabla f^T(k+1)\,R_s(k)}{\left\|R_s(k)\,\nabla f(k+1)\right\|^2} \end{aligned} \qquad (4.40)$$

$$\begin{aligned} U_s(k+1) &= U_{s+1}(k+1) - \\ &- \frac{U_{s+1}(k+1)\,\nabla \hat{f}(k-s+1)\,\nabla \hat{f}(k-s+1) + \nabla \hat{f}(k-s+1)\nabla \hat{f}^T(k)\,U_{s+1}(k+1)}{\left\|\nabla \hat{f}(k-s+1)\right\|^2} + \\ &+ \frac{\nabla \hat{f}(k-s+1)\,U_{s+1}(k+1)\,\nabla \hat{f}(k-s+1)}{\left\|\nabla \hat{f}(k-s+1)\right\|^4}\nabla \hat{f}(k-s+1)\nabla \hat{f}^T(k-s+1). \end{aligned} \qquad (4.41)$$

Таким образом, процедуры (4.37), (4.38), (4.40) соответствуют накоплению, а (4.37), (4.39), (4.41) – сбросу устаревшей информации.

4.4 Модификации многошаговых процедур, содержащие зону нечувствительности

4.4.1 Модификация многошаговых процедур.

По аналогии с рассмотренными выше модифицированными одношаговыми процедурами обучения используем в данном алгоритме зону нечувствительности, определяемую следующим образом:

$$\theta(k)=\theta(k-1)+\gamma\nabla f_s\,(k)\left(\nabla f_s^T(k)\nabla f_s\,(k)\right)^{-1}F_s\,(e,\Delta,k). \tag{4.42}$$

Здесь $F_s\,(e,\Delta,k)=\left(f(e,\Delta,k),f(e,\Delta,k-1),...,f(e,\Delta,k-s+1)\right)^T$ - вектор Sx1, компонентами которого являются функции от рассогласований на S тактах.

Как показано в Приложении Ж, данная модификация многошаговой процедуры будет иметь вид

$$\begin{aligned}\theta(k)&=\theta(k-1)+\gamma\frac{R_{s-1}(k-1)\nabla f(k)}{\nabla f^T(k)R_{s-1}(k-1)\nabla f(k)}f(e,\Delta,k)+\\&+\gamma\left(I-\frac{R_{s-1}(k-1)\nabla f(k)\nabla f^T(k)}{\nabla f^T(k)R_{s-1}(k-1)\nabla f(k)}\right)r_{s-1}(k-1);\end{aligned} \tag{4.43}$$

$$r_s\,(k)=r_{s-1}(k-1)-\frac{R_{s-1}(k-1)\nabla f(k)}{\nabla f^T(k)R_{s-1}(k-1)\nabla f(k)}; \tag{4.44}$$

$$\begin{aligned}&R_i\,(k-s+i)=R_{i-1}(k-s+i-1)-\\&-\frac{R_{i-1}(k-s+i-1)\nabla f(k-s+i)\nabla f^T(k-s+i)R_{i-1}(k-s+i-1)}{\nabla f^T(k-s+i)R_{i-1}(k-s+i-1)\nabla f(k-s+i)},\\&i=1,2,...,s-1.\end{aligned} \tag{4.45}$$

4.4.2 Сходимость модифицированных многошаговых процедур.

Так как алгоритм (4.40) был получен в предположении стационарности исследуемого объекта [240], а затем исследованы его свойства при оценивании изменяющихся параметров [241], то рассмотрим сначала некоторые вопросы сходимости модифицированного многошагового алгоритма (4.43)-(4.45) для стационарного случая, а затем - для нестационарного.

Так как в случае стационарного объекта изменения параметров не происходит, т.е. $\theta^*(k)=\theta^*=const\left(\theta^*(k)-\theta^*(k-1)=0\right)$, то вычитая из обеих частей (4.43) θ^* запишем алгоритм относительно ошибок обучения

$$\begin{aligned}\theta(k)=\theta(k-1)+\gamma\frac{R_{s-1}(k-1)\nabla f(k)}{\nabla f^T(k)R_{s-1}(k-1)\nabla f(k)}f(k)+\\ +\gamma\left(I-\frac{R_{s-1}(k-1)\nabla f(k)\nabla f^T(k)}{\nabla f^T(k)R_{s-1}(k-1)\nabla f(k)}\right)r_{s-1}(k-1).\end{aligned}\tag{4.46}$$

Умножая полученное выражение слева на $\theta^T(k)$ и пропуская несложные преобразования, получим

$$\begin{aligned}&\left\|\theta(k)\right\|^2=\left\|\theta(k-1)\right\|^2+\\ &+2\gamma\frac{R_{s-1}(k-1)\nabla f(k)}{\nabla f^T(k)R_{s-1}(k-1)\nabla f(k)}\times\\ &\times\left(\theta^T(k-1)R_{s-1}(k-1)\nabla f(k)-\gamma\nabla f^T(k)R_{s-1}(k-1)r_{s-1}(k-1)\right)f(e,\Delta,k)+\\ &+\frac{\gamma^2}{\nabla f^T(k)R_{s-1}(k-1)\nabla f(k)}\left(2\nabla f^T(k)r_{s-1}(k-1)-1\right)f^2(e,\Delta,k)+\\ &+2\gamma\theta^T(k-1)\left(I-\frac{R_{s-1}(k-1)\nabla f(k)\nabla f^T(k)}{\nabla f^T(k)R_{s-1}(k-1)\nabla f(k)}\right)r_{s-1}(k-1).\end{aligned}\tag{4.47}$$

Здесь учтено, что

$$\nabla f^T(k)R_{s-1}^T(k-1)R_{s-1}(k-1)\nabla f(k) = \nabla f^T(k)R_{s-1}(k-1)\nabla f(k).$$

Выбирая в качестве функции Ляпунова величину $\Delta V(k) = \|\theta(k-1)\|^2 - \|\theta(k)\|^2$, получаем, что с учётом $\gamma > 0$ сходимость алгоритма (4.43)-(4.45) будет иметь место при выполнении условия

$$A\nabla f^2(e,\Delta,k) + Bf\ (e,\Delta,k) + \mathrm{C} \geq 0, \tag{4.48}$$

где

$$A = \frac{\gamma}{\nabla f^T(k)R_{s-1}(k-1)\nabla f(k)}\left(1 - 2\nabla f^T(k)r_{s-1}(k-1)\right); \tag{4.49}$$

$$B = \frac{2}{\nabla f^T(k)R_{s-1}(k-1)\nabla f(k)}\left(\gamma\nabla f^T(k)R_{s-1}(k-1)r_{s-1}(k-1) - \theta^T(k-1)R_{s-1}(k-1)\nabla f(k)\right); \tag{4.50}$$

$$\begin{aligned} C = {} & 2\theta^T(k-1)\left(\frac{R_{s-1}(k-1)\nabla f(k)\nabla f^T(k)}{\nabla f^T(k)R_{s-1}(k-1)\nabla f(k)} - I\right)r_{s-1}(k-1) - \\ & - \left\|\left(I - \frac{R_{s-1}(k-1)\nabla f(k)\nabla f^T(k)}{\nabla f^T(k)R_{s-1}(k-1)\nabla f(k)}\right)r_{s-1}(k-1)\right\|^2. \end{aligned} \tag{4.51}$$

Решение неравенства (4.48) даст значения $f(e,\Delta,k)$, которые ввиду их громоздкости здесь не приводятся, и при которых данный алгоритм будет сходиться. Нетрудно видеть, что эти значения $f(e,\Delta,k)$ будут зависеть как от известных величин $\left(\gamma, \nabla f(k), R_{s-1}(k-1), r_{s-1}(k-1)\right)$, так и от неизвестного вектора ошибок $\theta(k-1)$. Хотя скалярное произведение $\theta^T(k-1)\nabla f(k)$ легко вычисляется и представляет собой величину $\xi(k)$, скалярные произведения типа $\theta^T(k-1)R_{s-1}(k-1)\nabla f(k)$ и $\theta^T(k-1)r_{s-1}(k-1)$ вычислить невозможно. Кроме того, получающееся значение нелинейной функции $f(e,\Delta,k)$ на *k*-ом такте будет зависеть от значений этой функции на (*k*-1)-м, (*k*-2)-м,..., (*k*-s+1)-

м тактах, входящих в вектор $r_{s-1}(k-1)$. Таким образом, получающееся в результате решения неравенства (4.48) значение $f(e,\Delta,k)$ имеет скорее теоретический, чем фактический интерес.

В Приложении З качестве примера исследована сходимость процедуры:

$$\theta(k)=\theta(k-1)+\gamma\frac{R_{s-1}(k-1)\nabla f(k)}{\nabla f^T(k)R_{s-1}(k-1)\nabla f(k)}\left(y(k)-\theta^T(k-1)\nabla f(k)\right). \qquad (4.52)$$

$$R_i\,(k-s+i)=R_{i-1}(k-s+i-1)-$$
$$\frac{R_{i-1}(k-s+i-1)\nabla f(k-s+i)\nabla f^T(k-s+i)R_{i-1}(k-s+i-1)}{\nabla f^T(k-s+i)R_{i-1}(k-s+i-1)\nabla f(k-s+i)}, \qquad (4.53)$$
$$i=1,2,...,s-1,$$

являющейся частным случаем (4.43)-(4.45).

Соответствующая модификация этого алгоритма с использованием зоны нечувствительности будет выглядеть следующим образом:

$$\theta(k)=\theta(k-1)+\frac{R_{s-1}(k-1)\nabla f(k)}{\nabla f^T(k)R_{s-1}(k-1)\nabla f(k)}f\left(e(k),\Delta,\mathrm{k}\right), \qquad (4.54)$$

где $R_{s-1}(k-1)$ вычисляется в соответствии с (4.53).

Анализируя полученное соотношение (Ж.7), можно сделать вывод, что решение неравенства (4.48) не представляется возможным, так как в него помимо упоминавшихся выше неизвестных величин вошли дополнительно подлежащее определению или оцениванию потактное приращение вектора параметров $\Delta\theta^*(k)$, а также различные скалярные произведения, содержащие $\Delta\theta^*(k)$. Однако полученные зависимости позволяют сделать вывод о том, что для монотонной сходимости процедур обучения необходимо на каждом такте выбирать (или корректировать) используемую в них величину зоны

нечувствительности в зависимости от степени нестационарности исследуемого объекта. Следовательно, целесообразно строить многошаговые процедуры, использующие адаптивную настройку зоны нечувствительности аналогично тому, как это было сделано в разделе 3.

4.5 Робастные процедуры Гаусса-Ньютона и Левенберга-Марквардта

4.5.1 Рекуррентные процедуры Гаусса-Ньютона и Левенберга-Марквардта.

Робастные процедуры Гаусса-Ньютона и Левенберга-Марквардта вытекают из

$$\hat{\theta}_K(i+1) = \hat{\theta}_K(i) - \gamma_K(i)\left[Q_K(i)\right]^{-1} \nabla F_K\left(\hat{\theta}(i)\right)$$

при выборе соответственно $\gamma_K(i) = 1$, $Q_K(i) = \nabla^2 F_K\left(\hat{\theta}(i)\right)$ и $Q_K(i) = \nabla^2 F_K\left(\hat{\theta}(i)\right) + \delta I$. Здесь $\nabla^2 F_K\left(\hat{\theta}(i)\right) = \frac{1}{k}\sum_{i=1}^{K} \psi'\left(e\left(i,\theta\right)\right)\nabla\hat{f}(i)\nabla\hat{f}^T(i)$ - матрица Гессе; $\delta > 0$ - параметр регуляризации.

Следует отметить, что в настоящее время существует достаточно много таких функций $\rho(e)$, обеспечивающих получение робастных оценок, однако учитывая, что в процедурах обучения используется $\rho''(e(k))$, целесообразно выбирать функции $\rho(e(k))$, которые имеют отличные от нуля вторые производные. В качестве таких функций могут быть взяты, например,

$$\rho[e(k)] = \ln\cosh(e(k)), \tag{4.55}$$

$$\rho[e(k)] = \sqrt{1 + e^2(k)} - 1, \tag{4.56}$$

$$\rho[e(k)] = c^2\left(\frac{|e(k)|}{c} - \ln\left(1 + \frac{|e(k)|}{c}\right)\right), \qquad (4.57)$$

графики которых приведены на рис.4.5.

Необходимо отметить, что при использовании функционалов типа (4.57) возникает проблема выбора (оценивания) параметра c (на рис.4.5 приведен вид функционала при $c = 5$).

В этом случае при обучении в режиме on-line на каждом такте поступления новой информации об обучающей паре $\{x_k, y_k\}$ осуществляется коррекция оценки в соответствии с формулами

$$\hat{\theta}(k) = \hat{\theta}(k-1) + \frac{P(k-1)\nabla\hat{f}(k)\psi(e(k))}{1 + \psi'\left(e(k)\right)\nabla^T\hat{f}(k)P(k-1)\nabla\hat{f}(k)}; \qquad (4.58)$$

$$P(k) = P(k-1) - \frac{P(k-1)\nabla\hat{f}(k)\nabla^T\hat{f}(k)P\left(k-1\right)}{1 + \psi'\left(e(k)\right)\nabla^T\hat{f}(k)P(k-1)\nabla\hat{f}(k)}\psi'\left(e(k)\right). \qquad (4.59)$$

Робастная рекуррентная процедура Левенберга-Марквардта может быть получена, если воспользоваться аппроксимацией. После несложных преобразований в окончательном виде рекуррентную робастную процедуру Левенберга-Марквардта, минимизирующую произвольный критерий М-обучения, можно записать следующим образом:

$$\hat{\theta}(k+1) = \hat{\theta}(k) + P(k)\nabla\hat{f}(k)\psi\left(e(k)\right); \qquad (4.60)$$

$$P(k) = P(k-1) - P(k-1)\nabla\hat{f}^*(k)K^{-1}(k)\nabla^T\hat{f}^*(k)P(k-1)\psi'(k), \qquad (4.61)$$

где

$$K(k) = \psi'(k)\nabla^T\hat{f}^*(k)P(k-1)\nabla\hat{f}^*(k) + \Lambda^*(k);$$

$$\Lambda^{*-1}(k)=\begin{pmatrix}1 & 0\\ 0 & \delta N\end{pmatrix};$$

$$\nabla\hat{f}^{*}(k)=\begin{pmatrix}\nabla^{T}\hat{f}^{*}(k)\\ 0...0 \quad 1...0\end{pmatrix}^{T}.$$

$$\uparrow$$

$$\text{позиция } = (k \bmod N+1)$$

Достаточно часто в алгоритмах (4.91-4.58)-(4.92-4.59) и (4.93-4.60)-(4.94-4.61) вместо $\psi'(e(k))$ используют весовые функции $\omega(e(k))$. Однако следует отметить, что как $\psi'(e(k))$, так и $\omega(e(k))$ некоторых функционалов (см. рис. 4.5) при определенных значениях аргументов могут быть отрицательными, что приводит к неустойчивости алгоритмов обучения. Поэтому в этих алгоритмах целесообразно использовать вместо величин $\psi'(e(k))$ и $\omega(e(k))$ их модули $\left|\psi'(e(k))\right|$ и $\left|\omega(e(k))\right|$.

4.5.2 Робастные процедуры Гаусса-Ньютона и Левенберга-Марквардта с зоной нечувствительности (с ограниченной точностью).

Зона нечувствительности, которая определяет степень допустимых ошибок, может быть определена следующим образом:

$$\psi_1'(e^{*}(k),\delta)=\begin{cases}\psi'\left(e^{*}(k)\right) \text{ для } \left|e^{*}(k)\right|>\delta;\\ 0 \text{ для } \left|e^{*}(k)\right|\le\delta;\end{cases} \tag{4.62}$$

и

$$\psi_2'(e^{*}(k),\delta)=\begin{cases}\psi'\left(e^{*}(k)\right)-\delta \text{ для } e^{*}(k)>\delta;\\ 0 \text{ для } \left|e^{*}(k)\right|\le\delta;\\ \psi'\left(e^{*}(k)\right)+\delta \text{ для } e^{*}(k)<-\delta.\end{cases} \tag{4.63}$$

Вид функций (4.62) и (4.63) для функционалов (4.55)-(4.57) представлен на рис. 4.6.

В этом случае робастная процедура Гаусса-Ньютона принимает вид

$$\hat{\theta}(k)=\hat{\theta}(k-1)+\frac{\alpha P(k-1)\nabla\hat{f}(k)\psi(e^*(k),\delta)}{1+\psi'\left(e^*(k),\delta\right)\nabla^T\hat{f}(k)P(k-1)\nabla\hat{f}(k)}; \tag{4.64}$$

$$P(k)=P(k-1)-\frac{\alpha P(k-1)\nabla\hat{f}(k)\nabla^T\hat{f}(k)P\left(k-1\right)}{1+\psi'\left(e^*(k),\delta\right)\nabla^T\hat{f}(k)P(k-1)\nabla\hat{f}(k)}\psi'\left(e^*(k),\delta\right); \tag{4.65}$$

где

$$\psi(e^*(k),\delta)=\begin{cases}\psi\left(e^*(k)\right)-\delta \text{ для } e^*(k)>\delta;\\ 0 \text{ для } \left|e^*(k)\right|\le\delta;\\ \psi\left(e^*(k)\right)+\delta \text{ для } e^*(k)<-\delta.\end{cases}$$

$$\alpha=\begin{cases}1 \text{ если } \left|e^*(k)\right|>\delta;\\ 0 \text{ в противном случае.}\end{cases}$$

Из рисунка видно, что у функционала (4.57) существуют области, где $\psi'<0$. Это может привести к нестабильности оценок $\hat{\theta}$. Таким образом, в процедуре (4.64), (4.65) вместо $\psi'\left(e^*(k),\delta\right)$ может быть использована весовая функция $\omega\left(e^*(k),\delta\right)$, которая, как видно из рисунка, всегда больше нуля. В этом случае процедура (4.64), (4.65) принимает вид

$$\hat{\theta}(k)=\hat{\theta}(k-1)+\frac{\alpha P(k-1)\nabla\hat{f}(k)\psi(e^*(k),\delta)}{1+\omega\left(e^*(k),\delta\right)\nabla^T\hat{f}(k)P(k-1)\nabla\hat{f}(k)} \tag{4.66}$$

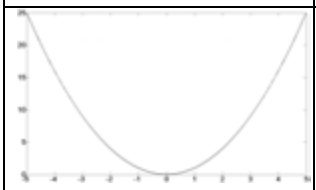
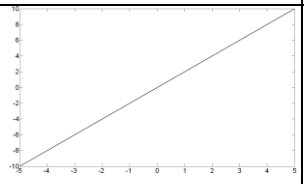
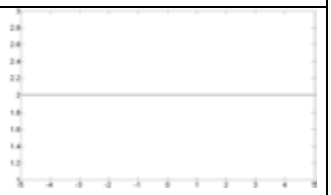
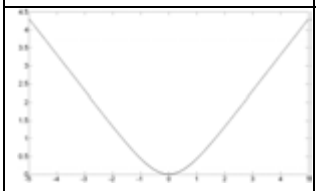
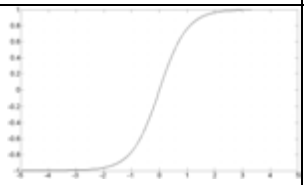
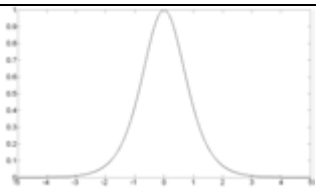
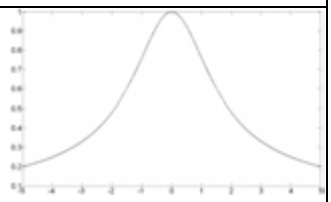
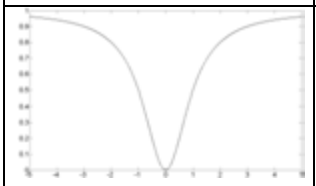
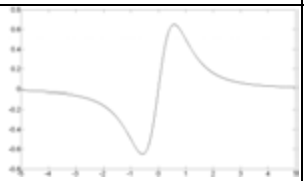
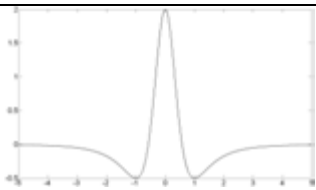
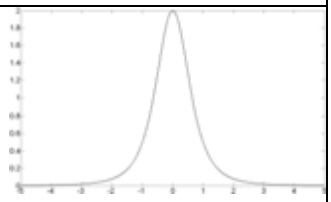
Функционал $\rho(e(k))$	$\psi(e(k))$	$\psi'(e(k))$	$\omega(e(k))$
 (4.55)			
 (4.56)			
 (4.57)			

Рисунок 4.5 – Графики функционалов (4.55)-(4.57), их производные и весовые функции

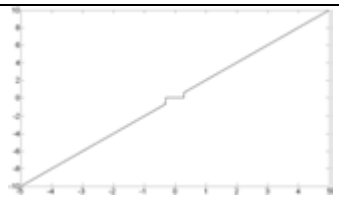
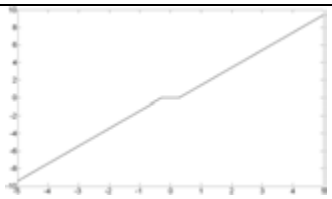
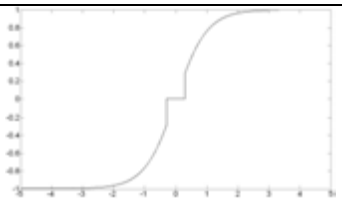
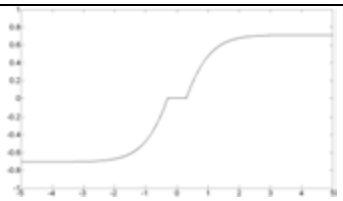
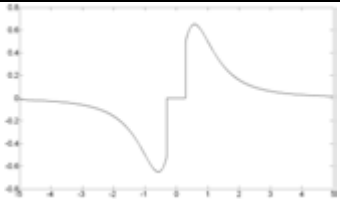
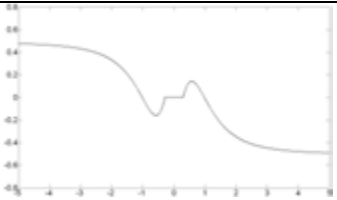
$\rho(e(k))$	$\psi_1(e^*(k),\delta)$	$\psi_2(e^*(k),\delta)$
(4.55)		
(4.56)		
(4.57)		

Рисунок 4.6 – Вид функций (4.62) и (4.63) для функционалов (4.55)-(4.57)

$$P(k)=P(k-1)-\frac{\alpha P(k-1)\nabla\hat{f}(k)\nabla^T\hat{f}(k)P\left(k-1\right)}{1+\omega\left(e^*(k),\delta\right)\nabla^T\hat{f}(k)P(k-1)\nabla\hat{f}(k)}\omega\left(e^*(k),\delta\right). \qquad (4.67)$$

4.6 Оценивание параметров функционалов

Как следует из соотношений (4.9), (4.14)-(4.15), параметры, входящие в выражение как для симметричных так и для асимметричных функционалов, влияют на их вид и зависят от статистических свойств помехи $\xi(k)$. При этом необходимо иметь ввиду следующее. В случае нормального распределения, оптимальные оценки, получаемые с помощью метода наименьших квадратов, инвариантны к параметру масштаба σ_ξ^2. При М-оценивании, вследствие неоднородности функции ρ, получаемые оценки не сохраняют свойства инвариантности. Для того чтобы свойство инвариантности масштаба выполнялось, в функционале (4.3) вместо ошибки $e\left(i,\theta\right)$ следует брать $\tilde{e}\left(i,\theta\right)=\left(e\left(i,\theta\right)-m\right)/S$, где S - помехоустойчивая оценка параметра масштаба или мера рассеяния остаточных разностей (в случае нормального распределения S является оценкой σ_ξ); m - математическое ожидание засоряющей помехи $q(x)$ в модели (4.3), которое в общем случае отлично от нуля.

При нулевом математическом ожидании помехи оценка параметра масштаба S используются в качестве константы c, входящей в рассмотренные выше функционалы и функции влияния.

4.6.1 Оценивание параметра масштаба в режиме off-line.

Если обучение сети происходит в режиме off-line, то в качестве оценки S следует взять какую-либо MAD-оценку (**M**edian of **A**bsolute **D**eviations, медиана абсолютных отклонений) [242], которая, как показано в [227], является наиболее помехоустойчивой.

В ранних работах по робастному оцениванию в качестве устойчивых оценок параметра масштаба рассматривались медианные.

Так в [233] отмечается, что в качестве оценки S можно использовать медиану модуля

$$S = med\left\{|e|\right\}. \tag{4.68}$$

Более точная, MAD-оценка (Median of Absolute Deviations)

$$S = \frac{med\left\{|e|\right\}}{0.6745}. \tag{4.69}$$

рассматривалась в работах [227-228, 232]. Кроме того, в [227] доказано, что MAD-оценка является наиболее помехоустойчивой оценкой дисперсии.

В [243] использовалась медианная оценка вида

$$S = \frac{med\left\{|e - med(e)|\right\}}{0.6745}. \tag{4.70}$$

Несколько иной подход оценивания S изучался в работах [233, 244-245]. Здесь рассматривался предложенный Хьюбером подход получения оценки параметра S путем решения относительно S уравнения

$$\frac{1}{n-2p-1}\sum_{i=p+1}^{n}\psi^2\left[\frac{e_i}{S_i}\right] = \beta, \tag{4.71}$$

где константа β выбирается так, что S является состоятельной оценкой σ_ε в случае $\xi \in N(0,\sigma_\varepsilon^2)$, т.е. $\beta = E_\Phi\left\{\psi^2(e)\right\}$, где Φ - стандартное нормальное распределение:

$$\beta = \frac{1}{\sqrt{2\pi}} \int \left[\Phi(x)\right]^2 e^{-\frac{x^2}{2}} dx \,. \qquad (4.72)$$

Значения константы β для различных функций влияния $\psi'(e)$ приведены в [245].

При решении практических задач для оценивания S обычно используют некоторое ограниченное число измерений. Например, в работе [246] рассматривается последовательность ошибок $e(i)$, $i = 1, 2, ..., n$, из которых k последних ($p < k < n$) используются для получения оценки параметра масштаба.

Предложенная в этой работе оценка имеет вид

$$S_k = \frac{d_k}{\Phi^{-1}\left[0.5(1 + k/n)\right]}, \qquad (4.73)$$

где d_k - половина ширины окна, включающего k последних ошибок; $\Phi^{-1}[\bullet]$ - аргумент общей (интегральной) функции распределения.

Оптимальное значение k соответствует минимуму дисперсии нормализованной ошибки ε_k^2 для определения которой используются только разности q_k точек:

$$\varepsilon_k^2 = \frac{1}{q_k - p} \sum_{i=1}^{q_k} \left(\frac{e_k(i)}{S_k} \right)^2 = \frac{\hat{\sigma}_k^2}{S_k^2} \,. \qquad (4.74)$$

4.6.2 Оценивание параметров помех в режиме on-line.

При решении задач в режиме on-line естественным является рекуррентное оценивание параметров помехи.

Помехоустойчивое оценивание параметра положения на основе алгоритма стохастической аппроксимации Роббинса-Монро для случая,

когда множество распределений помехи принадлежит классу асимметричных ε-засоренных нормальных распределений (4.3), рассматривалось в работах [247, 248], а для распределений произвольного вида – в работе [249]. Такой подход является весьма привлекательным в вычислительном аспекте, так как оценки определяются с помощью простой рекуррентной формулы и не используют в явном виде данных прошлых наблюдений.

4.6.3 Аппроксимация асимметрично распределенной помехи.

При использовании модели засорения (4.3) необходимо также оценить величины S_1^2 и S_2^2 и учесть эти оценки в алгоритме обучения. Если σ_1^2 и σ_2^2 не изменяются во времени, такое оценивание также может быть осуществлено методом стохастической аппроксимации:

$$S_1^2(k) = \begin{cases} S_1^2(k-1) + \dfrac{1}{l_1(k)}\left(\tilde{e}^2(k) - S_1^2(k-1)\right), \textit{если } \left|\tilde{e}(k)\right| \le 3S_1\ (k- \\ S_1^2(k-1)\ \textit{в противном случае}, \end{cases}$$

$$S_2^2(k) = \begin{cases} S_2^2(k-1) + \dfrac{1}{l_2(k)}\left(\tilde{e}^2(k) - S_2^2(k-1)\right), \textit{если } \left|\tilde{e}(k)\right| > 3S_1\ (k- \\ S_2^2(k-1)\ \textit{в противном случае}; \end{cases} \qquad (4.75)$$

где

$$l_1(k) = k - l_2(k);$$

$$l_2(k) = \begin{cases} 0, \textit{если } \left|\tilde{e}(k)\right| \le 3S_1\ (k-1), \\ l_2(\text{k-1}) + 1\ \textit{в противном случае}. \end{cases} \qquad (4.76)$$

Общая дисперсия помехи, вычисляемая по формуле

$$S^2(k) = \begin{cases} S_1^2(k), \textit{если } \left|\tilde{e}(k)\right| \le 3S_1\ (k-1); \\ S_2^2(k)\ \textit{в противном случае}, \end{cases} \qquad (4.77)$$

может быть использована как параметр взвешивания квадратичного функционала [250], т.е. $\rho\left(\tilde{e}\left(k\right),S\right)=0.5e^2(k)S^{-2}(k)$. В этом случае оценка, получаемая путем его минимизации, будет более устойчивой.

Так как для квадратичной функции $\rho'\left(e\left(k,\theta,\sigma^2\right)\right)=e(k,\theta)\sigma^{-2}$ и $\rho''\left(e\left(k,\theta,\sigma^2\right)\right)=\sigma^{-2}$, алгоритм обучения Гаусса-Ньютона имеет вид

$$\hat{\theta}(k)=\hat{\theta}(k-1)+\frac{\alpha P(k-1)\nabla\hat{f}(k)e(k,\theta)}{\sigma^2(k)+\nabla^T\hat{f}(k)P(k-1)\nabla\hat{f}(k)}, \tag{4.78}$$

$$P(k)=P(k-1)-\frac{\alpha P(k-1)\nabla\hat{f}(k)\nabla^T\hat{f}(k)P\left(k-1\right)}{\sigma^2(k)+\nabla^T\hat{f}(k)P(k-1)\nabla\hat{f}(k)}, \tag{4.79}$$

$$\alpha=\begin{cases}1, если\ \left|e(k)\right|>\delta;\\ 0, если\ \left|e(k)\right|\le\delta.\end{cases}$$

где $\sigma^2(k)$ настраивается в соответствии с (4.77).

Если распределение помехи асимметрично, то, как отмечалось выше, ненулевое математическое ожидание обусловливает смещение оценок, которое может быть устранено лишь при наличии информации о виде распределения. Если же такой информации нет, можно использовать некоторую аппроксимацию распределения, например, моделью (4.3) с $m_1\neq 0$ и $m_2\neq 0$. При этом следует иметь ввиду, что основной целью аппроксимации скорее является не адекватное отображение свойств помехи, а компенсация возможного смещения оценок.

В этом случае оценивание данного параметра можно осуществить с помощью следующего алгоритма стохастической аппроксимации:

$$\hat{m}_1(k)=\begin{cases}\hat{m}_1(k-1), если\ \left|e(k)\right|>3S_1\ (k-1);\\ \hat{m}_1(k-1)+\dfrac{1}{k^{\lambda}}\left(y(k)-\hat{m}_1(k-1)\right)\ в\ противном\ случае,\end{cases}$$

$$\hat{m}_2(k)=\begin{cases}\hat{m}_2(k-1), \textit{если } |e(k)|\le 3S_1(k-1);\\ \hat{m}_2(k-1)+\dfrac{1}{k^{\lambda}}\left(y(k)-\hat{m}_2(k-1)\right) \textit{ в противном случае},\end{cases} \tag{4.80}$$

где $\lambda\in(0,1]$ - некоторый коэффициент.

4.7 Имитационное моделирование

Эксперимент 1. Решалась задача аппроксимации сильно зашумленной функции

$$y=0.725\sin\left(\frac{16x_1+8x_2}{\left(3+4x_1^2+4x_2^2\right)}\right)+0.2x_1+0.2x_2+\xi(k), \tag{4.81}$$

где $\xi(k)=(1-\varepsilon)q_1(k)+\varepsilon q_2(k)$ - засоренная помеха ($q_1(k)$, $q_2(k)$ – нормально распределенные помехи с дисперсиями σ_1^2 и σ_2^2 соответственно); с использованием МП и РБС. На 500-ом шаге обучения происходило изменение параметров помехи с $\sigma_2=6$ на $\sigma_2=12$. Результаты аппроксимации приведены на рис.4.7. На рис.4.7-а), б) приведены графики изменения ошибок обучения сетей МП и РБС соответственно, а на рис.4.7-в) показан график изменения количества нейронов шаблонного слоя сети РБС в ходе обучения.

Эксперимент 2. Рассматривалась задача аппроксимации при наличии помех с различными распределениями в выходных сигналах. Аппроксимируемая функция описывалась следующим уравнением

$$y=0.5+\frac{\sin^2\sqrt{x_1^2+x_2^2}-0.5}{1+0.001(x_1^2+x_2^2)}+\xi(k), \tag{4.82}$$

где x_1, x_2 – стационарные случайные последовательности с равномерным законом распределения в интервале [-10, 10], генерируемые датчиком случайных чисел; ξ – помеха измерений.

Популяция состояла из 128 особей (РБС), максимально допустимое количество нейронов в каждой сети было ограничено 100. На рис. 4.8-а) показана поверхность, описываемая уравнением (4.82), а на рис.4.8-б) приведена поверхность, восстановленная с помощью ЭРБС, использующей асимметричную модификацию фитнесс-функции Хьюбера. На рис. 4.9-а) приведены графики изменения значения различных фитнесс функций при решении данной задачи. Так, сплошной линией показан график изменения фитнесс-функции Хьюбера (4.8), линей со звездочками - фитнесс-функции Хемпеля (4.9), а с кружками – квадратичной фитнесс-функции, пунктирной же линей об означено целевое значение. Как видно из рисунка, использование квадратичной фитнесс-функции является нецелесообразным при наличии помех измерений. График изменения количества нейронов сети-победителя в популяции показан на рис. 4.9-б).

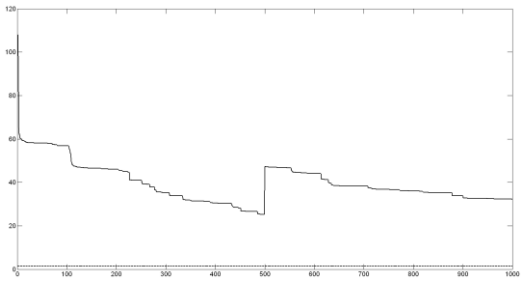
Эксперимент 3. Решалась задача аппроксимации зашумленной нелинейной функции

$$y(k) = 0.725\sin\left(\frac{16x_1 + 8x_2}{3 + 4x_1^2 + 4x_2^2}\right) + 0.2x_1 + 0.2x_2 + \xi(k), \qquad (4.83)$$

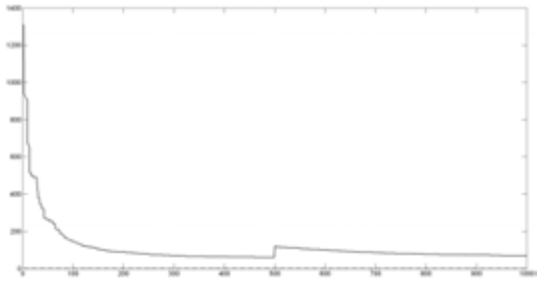
где $\mathbf{x} = [x_1, x_2]^T$ – входной сигнал, компоненты которого представляли собой стационарные случайные последовательности с равномерным законом распределения в интервале [-1, 1], генерируемые датчиком случайных чисел;

$\xi(k) = (1-\varepsilon)q_1(k) + \varepsilon q_2(k)$ - засоренная помеха ($q_1(k)$, $q_2(k)$ - нормально распределенные помехи с дисперсиями σ_1^2 и σ_2^2 соответственно).

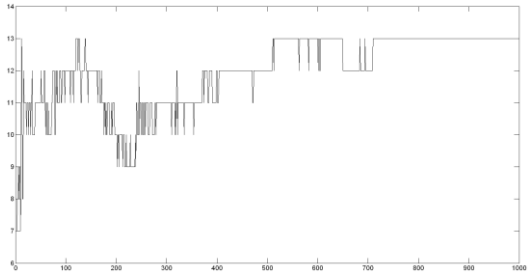
При исследовании данной функции использовалось 50000 обучающих пар. Результаты аппроксимации функции (4.83) с использованием различных



а)

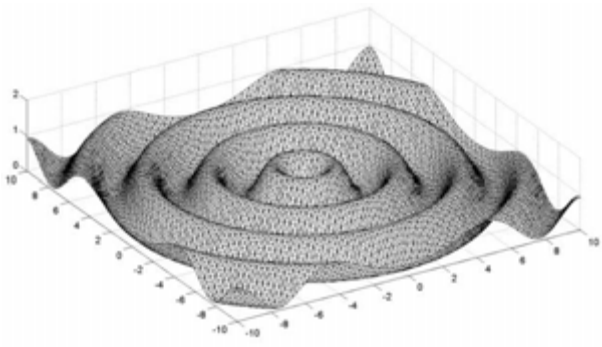


б)

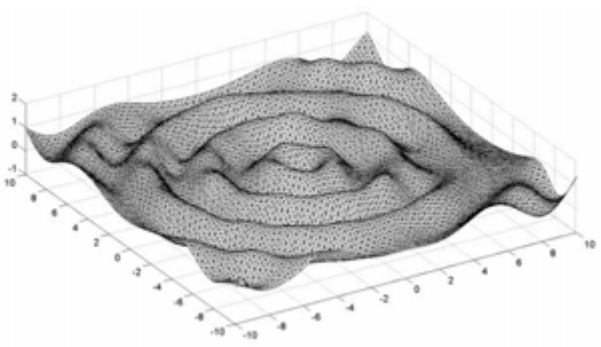


в)

Рисунок 4.7 – Результаты аппроксимации функции (4.81) с помощью МП и РБС

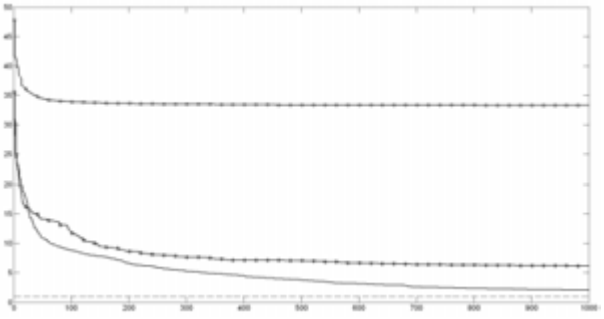


а)

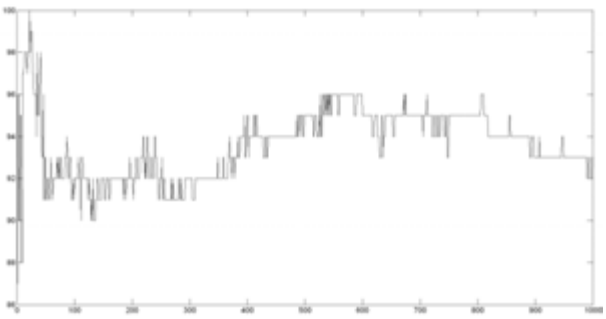


б)

Рисунок 4.8 – Поверхность, описываемая уравнением (4.82) – а) и ее аппроксимация с помощью ЭРБС – б)



а)



б)

Рисунок 4.9 – Графики изменения значения фитнесс функций – а) и количества нейронов сети-победителя – б)

значений ε, σ_1^2 и σ_2^2 приведены в табл.4.1. Здесь представлены значения среднеквадратичной ошибки, вычисленной после обучения сети для 2500 эталонных значений по формуле

$$\varsigma = \frac{\sqrt{\sum_{i=1}^{2500} \left(y^*(i) - \hat{y}(i) \right)^2}}{2500}, \qquad (4.84)$$

где y^* – эталонное значение выходного сигнала при отсутствии помех измерений; $\hat{y}$ – реальный выходной сигнал сетей.

В табл. 4.2 представлены результаты оценивания σ_1 и σ_2 и количества выбросов в соответствии с (4.77). Использование зон нечувствительности привело к сокращению времени обучения примерно на 16%.

Графики уточнения оценок σ_1 и σ_2 на каждом шаге обучения сети приведены на рис. 4.10.

Как видно из результатов моделирования, алгоритм (4.75)-(4.77) позволяет получить достаточно точные оценки σ_1^2 и σ_2^2 (при условии $\sigma_1^2 << \sigma_2^2$), которые используются при нормировании функции потерь, что обеспечивает высокую точность аппроксимации сильно зашумленных нелинейных функций.

Эксперимент 4. Проводился сравнительный анализ результатов аппроксимации функции (4.83) радиально-базисной сетью с 15 нейронами в скрытом слое, обученной с применением алгоритмов (4.64)-(4.65) и (4.66)-(4.67) и использованием производных $\psi_i'(e)$ и весовых функций $\omega_i(e)$ $i = \overline{1,4}$ на 50000 обучающих парах.

В алгоритмах обучения были использованы функционалы (4.8)-(4.9), (4.14)-(4.15). Результаты экспериментов приведены в табл.4.3. Здесь представлены значения среднеквадратичной ошибки, вычисленной после обучения сети для 2500 эталонных значений по формуле (4.84).

Значения (4.84) для квадратичного критерия обучения оказались равными: для алгоритма Гаусса-Ньютона 33.78, для алгоритма Левенберга-Марквардта 28.97 (в обоих случаях в алгоритмах обучения использовалась весовая функция ω).

В табл.4.4 приведены результаты аппроксимации функции (4.83) радиально-базисной сетью, обученной с помощью алгоритма Гаусса-Ньютона, минимизирующего взвешенный квадратичный функционал $\rho\left(\tilde{e}(k), S\right)$ и АФ $\rho_4\left(\tilde{e}(k), S\right)$. Здесь показаны также заданные параметры σ_1, σ_2, m_2, реальное количество выбросов L и полученные в процессе обучения их оценки S_1, S_2, $\hat{m}_2$ и $\hat{L}$.

При этом числа без скобок соответствуют тем же условиям засорения помехи, что и выше, а числа в скобках – случаю, когда нормально распределенная помеха засорена помехой, имеющей распределение Релея *Ray*(1.6).

Наконец, в табл.4.5 отражены результаты аппроксимации функции (4.83) той же РБС, обученной с помощью алгоритма Гаусса-Ньютона, использующего ψ-функцию Хьюбера (4.8), при наличии помехи с $\varepsilon = 0.1$, $q_1(k) \sim N\left(m_1, \sigma_1^2\right)$ и $q_2(k) \sim N\left(m_2, \sigma_2^2\right)$, $m_1, m_2 \neq 0$.

Как следует из результатов моделирования, использование обычного квадратичного функционала в алгоритмах обучения РБС при наличии негауссовских помех измерений, является нецелесообразным, в то же время применение асимметричных модификаций неквадратичных функционалов приводит к лучшим результатам.

Эксперимент 5. Решалась задача аппроксимации функции (4.83), где $\xi(k)$ - случайная помеха, распределенная по закону Релея с дисперсией $\sigma = 1.6$ ($m = 2.0053$). Параметры нейронной сети выбирались такие же, как и в предыдущем эксперименте. Зашумленная поверхность, ее сечения и результаты аппроксимации представлены на рис.4.11. Линией 1 обозначено сечение зашумленной поверхности, 4 – сечение эталонной поверхности, 2 –

сечение поверхности, восстановленной без учета математического ожидания помехи, 3 – с учетом оценки математического ожидания помехи, полученной с помощью алгоритма (4.80) с $\lambda = 0.4$. На последнем шаге обучения сети оценка математического ожидания помехи была равна $\hat{m} = 2.0078$.

Так как в реальных системах информация в виде распределения помехи зачастую неизвестна, в данном эксперименте использовалась аппроксимация распределения Релея *Ray*(1.6) моделью Тьюки-Хьюбера (4.3), в которой и основное, и засоряющее распределения являются нормальными с m_1, $m_2 \neq 0$. В результате обучения РБС были получены следующие параметры засоренной помехи: $S_1 = 0.9943$, $S_2 = 0.7529$, $\hat{m}_1 = 2.0424$, $\hat{m}_2 = 5.3030$, которые и использовались для коррекции выходных сигналов сети. При этом средняя ошибка (4.61) аппроксимации функции (4.60) составила $\varsigma = 2.7704$. Соответствующие сечения показаны на рис.4.11 линией с крестиками.

Эксперимент 6. Проводилась аппроксимация функции, описываемой следующим уравнением [251]:

$$y(k) = x_1 \sin\left(4\pi x_1\right) - x_2 \sin\left(4\pi x_2 + \pi\right) + 1 + \xi(k). \qquad (4.85)$$

В данном эксперименте, как и в эксперименте 5, помеха $\xi(k)$ представляла собой смесь двух нормально распределенных помех $q_1(k) \sim N(0;0.6)$ и $q_2(k) \sim N(1;6)$ при $\varepsilon =$ 0.1. Гистограмма данной помехи представлена на рис.4.12-а)

Так как аппроксимируемая поверхность является достаточно сложной, количество нейронов в скрытом слое сети было увеличено до 150, а количество обучающих пар – до 150000.

На каждом шаге процесса обучения осуществлялась оценка параметров помехи с использованием алгоритмов (4.78)-(4.80). На последнем шаге обучения были получены следующие оценки: $S_1 = 0.6022$; $S_2 = 6.3530$; $\hat{m}_1 = 0.0231$ и $\hat{m}_2 = 1.2539$, которые использовались для коррекции

результатов. На рис. 4.12–б) показана эталонная поверхность, описываемая уравнением (4.85), а на рис. 4.12 – в) – восстановленная с помощью нейронной сети.

Как видно из результатов моделирования, асимметрия распределения помехи приводит к смещению оценок. Устранить данный нежелательный эффект можно путем оценивания математического ожидания помехи и последующей коррекции предъявляемых сети желаемых выходных сигналов на величину этой оценки.

Выводы по разделу 4

1. Методы обучения ИНС, основанные на минимизации квадратичных функционалов, не являются устойчивыми, если в измерениях присутствуют выбросы или помехи имеют распределения отличные от гауссовских. В качестве альтернативы, для обеспечения робастности целевую функцию модифицируют таким образом, чтобы ограничить влияние наибольших измерений. Основным следствием этого является, как правило, более низкая скорость сходимости алгоритмов оптимизации.

2. Если информация о принадлежности помехи некоторому определенному классу распределений известна, то путем минимизации оптимального критерия, представляющего собой взятый с обратным знаком логарифм функции распределения помехи, может быть получена оценка максимального правдоподобия (*M*-оценка). Если же такой информации нет, то для оценивания искомого вектора параметров следует применить какой-либо неквадратичный критерий, обеспечивающий робастность получаемой оценки.

3. Предложена новая процедура М-обучения нейросетей, в которой для выбора структуры сети применен робастный информационный критерий, для оценки параметров – робастный алгоритм Гаусса-Ньютона. Использованные

в данной процедуре вместо вторых производных функционалов значений весовых функций позволяет повысить ее устойчивость.

4. Классические робастные методы, ориентированы на симметричность засорения, когда выбросы одинаково часто появляются как в области отрицательных, так и в области положительных значений.

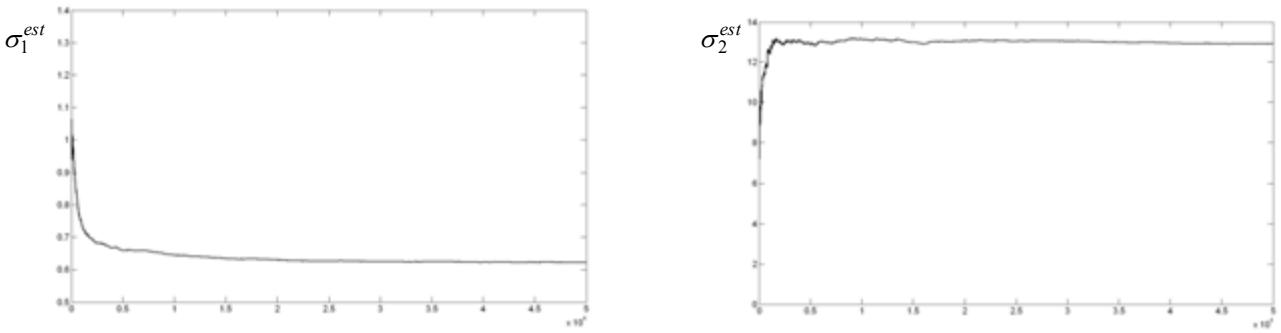
В более общей ситуации произвольного вида засорения, например, когда гауссовское засоряющее распределение имеет ненулевое математическое ожидание или когда засоряющее распределение является несимметричным, оценки, даваемые этими методами, будут смещенными.

5. Разработаны новые робастные методы обучения ИНС, дающие возможность эффективно обрабатывать информацию при наличии помех с асиметричными распределениями. Данные методы используют дополнительные процедуры оценивания параметров функционалов и помех, что позволяет осуществлять коррекцию получаемых оценок и устранять их смещение.

6. Разработаны новые процедуры коррекции параметров используемых при обучении функционалов и оценивания параметров помехи, описываемой моделью Тьюки-Хьюбера, что позволяет при отсутствии априорной информации о статистических свойствах помех адаптивно корректировать параметры ИНС.

7. Получил дальнейшее развитие эволюционный метод устранения влияния помех при определении структур и параметров нейросетевых моделей за счет использования оценок параметров модели помехи Тьюки-Хьюбера в процедуре М-обучения. Такой подход позволяет улучшить фильтрующие свойства используемых процедур, упростить структуру хромосомы и сократить тем самым процесс получения модели ИНС.

8. Результаты имитационного моделирования разработанных моделей и процедур обучения подтвердили их работоспособность и эффективность.



а) б)

Рисунок 4.10 – Результаты оценки $\sigma_1 = 0.6$ и $\sigma_2 = 12$ при $\varepsilon = 0.2$

Таблица 4.1 – Результаты аппроксимации функции (4.83)

Заданные параметры				$F[e(k)] = \left(\frac{e^2(k)}{2c} \right)$		
ε	σ_1^{ref}	σ_2^{ref}	Реальное кол-во выбросов	ς без зоны	ς с зоной 1-го типа, равной σ_1^{est}	ς с зоной 2-го типа, равной σ_1^{est}
0.0	0	0	0	0.6286	-	-
0.1	0.6	3	5061	1.5252	2.7339	2.6047
		6	5008	1.6415	2.4909	2.4697
		12	4991	1.9389	1.9634	1.9491
0.2	0.6	3	10013	1.6497	2.1061	2.0698
		6	10020	2.0402	2.1209	2.0813
		12	10111	1.9863	2.2117	2.1887

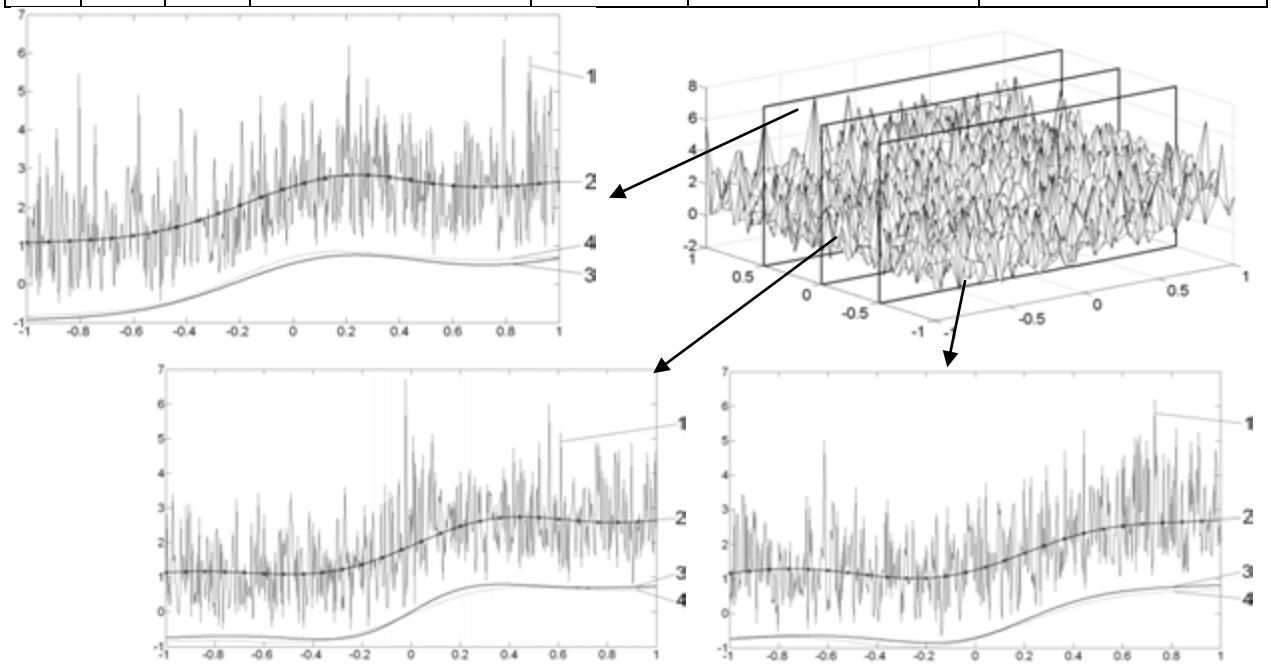
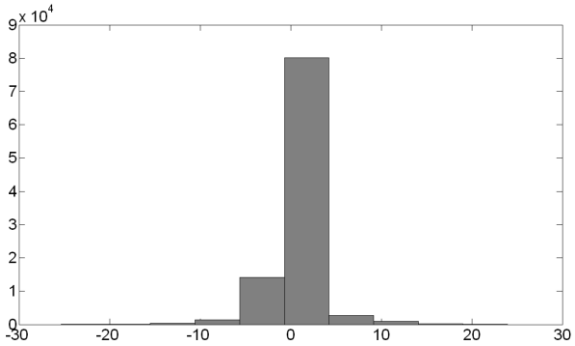
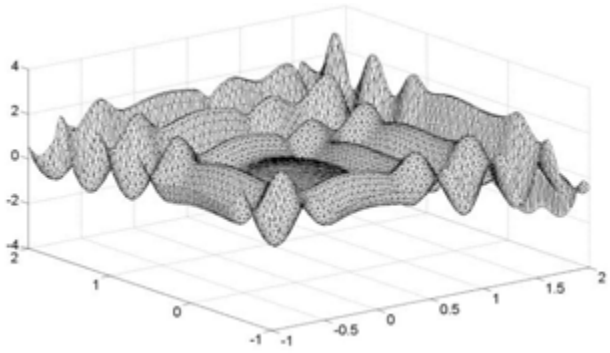


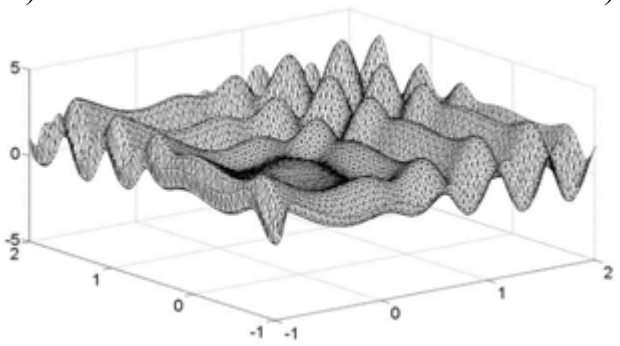
Рисунок 4.11 – Результаты аппроксимации функции (4.83)



а)



б)



в)

Рисунок 4.12 – Результаты аппроксимации функции (4.85)

Таблица 4.2 – Результаты оценивания σ_1 и σ_2 и количества выбросов в соответствии с (4.77)

Заданные параметры				Полученные оценки		
ε	σ_1^{ref}	σ_2^{ref}	Реальное кол-во выбросов	σ_1^{est}	σ_2^{est}	Оценочное кол-во выбросов N
0.0	0	0	0	-	-	-
0.1	0.6	3	5061	0.6369	4.0902	4758
		6	5008	0.6166	6.7468	4984
		12	4991	0.6073	12.5611	4969
0.2	0.6	3	10013	0.7351	4.3658	9957
		6	10020	0.6151	6.8815	9897
		12	10111	0.6220	12.8381	10005

Таблица 4.3 – Значения среднеквадратичной ошибки

Алгоритм		Значение ς для разных функционалов							
		$\rho_1(e)$	$\widetilde{\rho}_1(e)$	$\rho_2(e)$	$\widetilde{\rho}_2(e)$	$\rho_3(e)$	$\widetilde{\rho}_3(e)$	$\rho_4(e)$	$\widetilde{\rho}_4(e)$
Левенберга-Марквардта	$\psi'(e)$	3.1925	1.8514	4.5334	2.5431	4.9573	2.3699	2.3408	2.1393
	$\omega(e)$	2.5111	1.7123	3.3428	2.3968	5.1923	2.3864	3.0885	1.8163
Гаусса-Ньютона	$\psi'(e)$	2.0644	1.7309	3.9032	2.9075	5.5626	2.8054	2.2850	2.1639
	$\omega(e)$	2.2883	1.8676	4.2834	2.6939	5.6984	2.8663	3.6532	1.9101

Таблица 4.4 – Результаты аппроксимации функции (4.83)

Заданные параметры					Полученные оценки				Значение ς для разных функционалов	
ε	σ_1^{ref}	σ_2^{ref}	m_2	L	S_1	S_2	$\hat{m}_2$	$\hat{L}$	ρ	$\widetilde{\rho}_4$
0.1	0.6	1.5	1	4882	0.6399 (0.6384)	2.3315 (1.1537)	2.4626 (3.2320)	2896 (4944)	3.3005 (2.6217)	1.6474 (1.6392)
		3	5	5168	0.6344 (0.5994)	3.3840 (2.3243)	5.5819 (8.6127)	3704 (5085)	1.9878 (1.8434)	2.7888 (2.2881)
		6	6	4919	0.6115 (0.5989)	6.9127 (4.6167)	6.2708 (13.2043)	4062 (5076)	1.8960 (2.0123)	1.7629 (2.1436)
		12	9	4932	0.6086 (0.6005)	12.3464 (8.6636)	9.7411 (23.6266)	4510 (4908)	1.9973 (2.8050)	2.0761 (2.8718)
0.2	0.6	1.5	1	9945	0.7176 (1.0383)	2.0894 (1.7050)	2.8378 (4.2248)	5502 (2992)	6.8584 (17.2742)	3.0246 (5.8312)
		3	5	10356	0.6803 (0.5974)	3.3120 (2.2409)	5.7387 (8.7293)	7196 (9960)	3.0984 (2.1305)	3.2308 (3.4510)
		6	6	9992	0.6591 (0.5975)	6.8509 (4.1979)	6.7023 (13.3749)	7837 (9941)	1.7228 (2.2576)	2.4336 (4.9453)
		12	9	10020	0.6233 (0.5936)	12.1381 (8.6200)	9.9726 (23.7478)	9011 (9861)	1.5640 (2.1448)	2.8418 (3.3868)

Таблица 4.5 – Результаты аппроксимации функции (4.83)

Заданные параметры					Полученные оценки				Значение ς
ε	σ_1	σ_2	m_1	m_2	S_1	S_2	$\hat{m}_1$	$\hat{m}_2$	$\widetilde{\rho}_1$
0.1	0.6	1.5	0.25	1	0.4642	0.8829	0.1934	3.3337	2.4687
		3	0.5	5	0.6132	3.6363	0.4993	6.6085	1.4707
		6	0.75	6	0.6072	7.1325	0.7746	7.1782	2.0410
		12	1	9	0.6069	12.8221	0.9953	9.7078	1.9737
0.2	0.6	1.5	0.25	1	0.7256	2.3261	0.1151	2.5300	2.0921
		3	0.5	5	0.6708	3.3137	0.7734	7.1449	2.5811
		6	0.75	6	0.6294	7.0955	0.5095	7.3940	1.8348
		12	1	9	0.6157	13.1472	1.0533	9.7203	2.3019

РАЗДЕЛ 5
МНОГОКРИТЕРИАЛЬНАЯ ОПТИМИЗАЦИЯ ЭВОЛЮЦИОНИРУЮЩИХ НЕЙРОСЕТЕВЫХ МОДЕЛЕЙ

Использование ИНС требует решения задач структурной и параметрической оптимизации, соответствующие выбору оптимальной топологии сети и ее обучению (настройке параметров - в МП это весовые параметры и коэффициенты α в (2.1), в РБС – весовые параметры, центры базисных функций и их дисперсии).

Традиционные методы определения структуры сети заключаются либо в последовательном ее усложнении путем ввода новых нейронов и новых связей между ними, либо в последовательном ее упрощении, начиная с некоторой достаточно сложной топологии.

В отличие от задачи определения структуры, являющейся дискретной оптимизационной (комбинаторной), поиск оптимальных параметров осуществляется в непрерывном пространстве с помощью классических методов оптимизации.

Для обучения сетей прямого распространения с учителем применяются, как правило, алгоритмы, оптимизирующие некоторую целевую функцию. Однако традиционно во внимание принимается лишь одна цель в качестве стоимостной функции либо несколько целей объединяются в одну скалярную функцию. Это главным образом обусловлено тем, что большинство обычных алгоритмов обучения являются одноцелевыми, т.е. могут работать только со скалярными стоимостными функциями. Обычно в таких алгоритмах осуществляется поиск минимума функционала на некоторой выборке обучающих данных

$$F(e) = \frac{1}{M}\sum_{i=1}^{M}\rho\left(e\left(i,\boldsymbol{\theta}\right)\right), \tag{5.1}$$

где $\rho\left(e\left(i,\theta\right)\right)$ - некоторая функция потерь, зависящая от вида закона распределения присутствующей в измерениях помехи ξ; $e(i) = y(i) - \hat{y}(i)$ - ошибка аппроксимации; $y(i)$ и $\hat{y}(i)$ соответственно реальный и желаемый выходы ИНС; M – количество пар обучения в выборке.

Минимизация ошибки обучения только на обучающих данных может привести к оверфитингу (переобучению) - явлению, когда получаемый алгоритм обучения слишком хорошо работает на обучающей выборке и плохо – на тестовой. Вследствие этого, нейросетевая модель будет обладать плохими обобщающими свойствами. Для предотвращения эффекта переобучения модели необходимо контролировать ее сложность, что может быть осуществлено путем перехода от функционала (1.54) к функционалу вида

$$f = F(e) + \lambda\Omega, \tag{5.2}$$

где F - функция ошибки, например (5.1); Ω - мера сложности модели, например, количество свободных параметров модели; $\lambda > 0$ - некоторый свободно выбираемый параметр. Получаемый при этом алгоритм обучения способен оптимизировать две цели, хотя функция цели (5.2) является скалярной.

Однако использованию скаляризованных целевых функций для многоцелевой оптимизации присущи два основных недостатка, во-первых, нетривиальной является задача определения оптимального параметра λ, во-вторых, при этом может быть получено только одно решение, которое в ряде случаев может быть неэффективным. Это особенно важно при наличии

нескольких конфликтующих целей, приводящих к тому, что не представляется возможным получение единого оптимального решения, которое оптимизировало бы все цели одновременно. Так, например, уменьшение ошибки аппроксимации часто приводит к увеличению сложности модели.

Более мощным по сравнению с обучением на основе скалярной стоимостной функции является многоцелевое обучение на основе подхода Парето, когда минимизируется векторная целевая функция, что обеспечивает получение некоторого количества Парето-оптимальных решений. Так, скаляризованная двуцелевая проблема обучения (5.2) может быть сформулирована как многокритериальная оптимизация на основе Парето следующим образом:

$$\min\{f_1, f_2\}; \tag{5.3}$$

$$f_1 = F(e); \tag{5.4}$$

$$f_2 = \Omega. \tag{5.5}$$

Наиболее часто в качестве f_1 выбирается квадратичный функционал, а в качестве f_2, служащего для оценивания сложности нейросетевой модели, сумма квадратов весов:

$$\Omega = \sum_{i=1}^{M} w_i^2 \tag{5.6}$$

или сумма их абсолютных значений

$$\Omega = \sum_{i=1}^{M} |w_i|, \tag{5.7}$$

известные как регуляризатор Гаусса и Лапласа соответственно. Здесь $w_i, i = 1,...,M$ – веса нейросетевой модели; M – общее количество нейронов сети (для РБС) либо общее количество связей (для МП).

Сравнивая скаляризованное многокритериальное обучение на основе (5.2) и многокритериальное обучение по Парето (5.3), можно видеть, что с одной стороны при обучении по Парето нет проблемы выбора значения параметра λ перед обучением, а с другой - после обучения возникает необходимость выбора одного или нескольких вариантов решения из полученного набора Парето-оптимальных решений в соответствии с предпочтениями пользователя и определении таким образом наиболее приемлемого решения для рассматриваемой задачи.

Дополнительным преимуществом обучения на основе подхода Парето является то, что мульти-объективизация (разбитие сложной задачи с одной целевой функцией на несколько подзадач и последующим поиском для каждой подзадачи решения и выбор оптимального решения) может помочь алгоритму обучения избегать застревания в локальных экстремумах, повышая тем самым точность модели обучения.

Так как получаемая нейросетевая модель, с одной стороны, должна быть достаточно простой и удобной для использования ее в прикладных задачах, а с другой – наиболее полно отражать свойства исследуемого объекта, ее качество определяется некоторым набором критериев, т.е. задача построения нейромодели является многокритериальной.

5.1 Задача многокритериальной оптимизации по Парето

Задача многокритериальной оптимизации (МО) заключается в нахождении такого вектора решений, удовлетворяющего определенным ограничениям, который давал бы приемлемые значения для всех целевых

функций [252-255]. Следовательно, существует множество целевых функций (вектор целей), которые оптимизируются (минимизируются или максимизируются) одновременно. Так как цели зачастую вступают в противоречие друг с другом таким образом, что улучшение одной из них приводит к ухудшению другой, не существует единого оптимального решения, наилучшего относительно всех целевых функций. Вместо этого, есть множество оптимальных решений задачи многоцелевой оптимизации, известное как Парето оптимальные решения или фронт Парето. Понятие фронта Парето в области значений целевых функций в задаче МО означает набор таких решений, которые являясь недоминирующими по отношению друг к другу, в то же время доминируют над всеми остальными решениями в пространстве поиска. Таким образом, невозможно найти единое решение, которое бы превосходило все другие по отношению ко всем целям, т.е. переход между решениями, принадлежащими фронту Парето, не может привести к улучшению всех целей одновременно.

Так как в реальных задачах оптимизации обычно неизвестно, где находится глобальный фронт Парето, получаемые с помощью алгоритма недоминирующие решения не обязательно являются Парето-оптимальным. Однако полученные путем многокритериальной оптимизации недоминирующие решения часто ошибочно называют Парето-оптимальными.

Математически задача многокритериальной оптимизации по Парето может быть сформулирована следующим образом.

Требуется найти такой вектор решений $\boldsymbol{x}^* \in \Re^n$, который бы оптимизировал вектор целевых функций

$$F(\boldsymbol{x}) = [f_1(\boldsymbol{x}), f_2(\boldsymbol{x}), ..., f_k(\boldsymbol{x})]^T \tag{5.8}$$

при наличии *m* ограничений в виде неравенств

$$g_i(\boldsymbol{x}) \le 0, i = \overline{1,m}, \tag{5.9}$$

и p ограничений в виде равенств

$$h_j(\boldsymbol{x}) = 0, j = \overline{1,p}, \tag{5.10}$$

где $F(\boldsymbol{x}) \in \Re^k$ - вектор целевых функций, каждая из которых должна быть оптимизирована (обычно полагают, что все целевые функции должны быть минимизированы).

При многокритериальной минимизации на основе подхода Парето используется, как отмечалось выше, понятие доминирования.

Вектор $\boldsymbol{u} = [u_1, u_2, ..., u_n]^T \in \Re^k$ доминирует над вектором $\boldsymbol{v} = [v_1, v_2, ..., v_n]^T \in \Re^k$ (обозначается $\boldsymbol{u} \prec \boldsymbol{v}$) тогда и только тогда, когда $\forall i \in \{1,2,...,k\}, u_i \le v_i \wedge \exists j \in \{1,2,...,k\} : u_j < v_j$. Другими словами, существует как минимум одна компонента вектора $\boldsymbol{u}(u_j)$, которая меньше, чем v_j, в то время как остальные компоненты вектора $\boldsymbol{u}$ больше либо равны соответствующим компонентам вектора v.

Точка $\boldsymbol{x}^* \in \Omega$ (Ω некоторая область пространства $\Re^n$, удовлетворяющая условиям (5.9)-(5.10)) является *Парето-оптимальной* по отношению ко всем $\boldsymbol{x} \in \Omega$ тогда и только тогда, когда $F(\boldsymbol{x}^*) \prec F(\boldsymbol{x})$, т.е. решение $\boldsymbol{x}^*$ является Парето-оптимальным, если не может быть найдено никакое другое решение, которое бы доминировало над $\boldsymbol{x}^*$.

Для данной задачи МО:

множеством Парето P^* называется набор векторов $\boldsymbol{x} \in \Omega$, для которого не существует такого вектора $\boldsymbol{x}' \in \Omega$, для которого бы выполнялось условие $F(\boldsymbol{x}') \prec F(\boldsymbol{x}) \forall \boldsymbol{x} \in \mathrm{P}^*$;

фронтом Парето PF^* называется такой набор векторов значений целевых

функций $F(\mathbf{x}) \in \Re^k$, который получен с помощью векторов из множества Парето, т.е.

$$PF^* = \{F(\mathbf{x}) = \left(f_1(\mathbf{x}), f_2(\mathbf{x}), ..., f_k(\mathbf{x})\right) : \mathbf{x} \in \mathrm{P}^*\}.$$

Определения множества и фронта Парето иллюстрирует рисунок 5.1. [256]. Поскольку множество PF на этом рисунке является выпуклым, фронт Парето PF^* в данном случае представляет собой дугу AB, на которой точка A соответствует $f_1(\mathbf{x})$, а точка B – $f_2(\mathbf{x})$. Среди точек $F(\mathbf{x}_1)$, $F(\mathbf{x}_2)$ лежащих на фронте Парето, нет доминируемых, поскольку если $f_1(x_1) < f_1(x_2)$, то обязательно $f_2(x_1) > f_2(x_2)$.

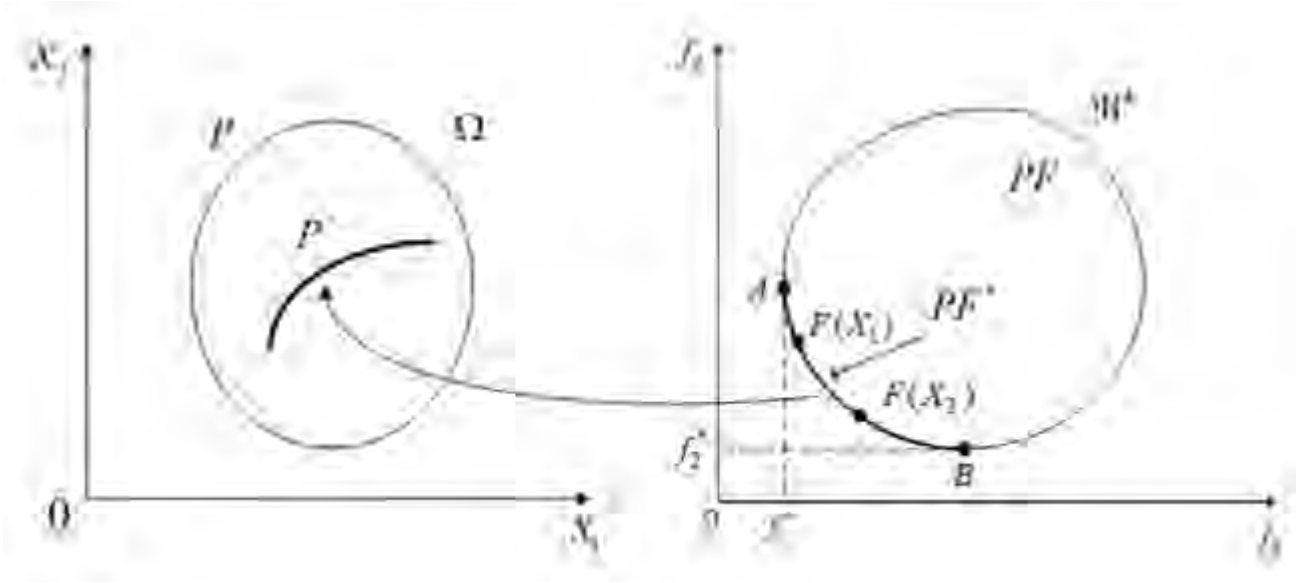


Рисунок 5.1 – Определения множества и фронта Парето

Задача приближенного построения множества Парето (а, тем самым, и фронта Парето) в МО-задаче называется *задачей Парето-аппроксимации.* Суть большинства популяционных методов Парето-аппроксимации состоит в итерационном уточнении множеств точек в *архивных множествах* A_F и A_X,

содержащих соответственно не доминируемые точки F_j^A и соответствующих ему точек X_j^A; $j = \overline{1,|A|}$, где |A| - мощность множеств A_F и A_X.

Если при этом на k-ой итерации появляется новая точка F_i , доминирующая некоторые точки из архива A_F, то все доминируемые точки, а также соответствующие точки из архива A_X удаляются. При удовлетворении некоторого критерия останова текущее содержимое архивов A_F и A_X полагаем искомой аппроксимацией фронта Парето PF^* и множества Парето P^* соответственно.

В популяционных методах Парето-аппроксимации новые точки для архивов A_F и A_X предоставляет популяция агентов s_i, текущие координаты которых в пространстве поиска Ω равны X_i, а в пространстве $\Re^k$ - $F_i = F(X_i)$; $i = \overline{1,|S|}$.

В популяционных оптимизационных алгоритмах миграция агентов в пространстве поиска подчинена задаче минимизации значений фитнесс-функции. Основной проблемой построения популяционных методов Парето-аппроксимации является построение фитнесс-функции, обеспечивающей перемещение агентов популяции s_i в направлении множества Парето P^*, а соответствующих точек F_i - в направлении фронта Парето PF^*.

В силу, как правило, меньшей размерности критериального пространства $\Re^k$ по сравнению с размерностью пространства поиска Ω, направление и величину шага перемещения агентов обычно отыскивают в терминах критериального пространства, а не пространства параметров. Важно также то, что относительно множества Парето не существует априорной информации, кроме того, что это множество точек, не связанных между собой отношением предпочтения $\prec$. В то же время по отношению к фронту Парето априорной информации значительно больше.

В общем случае выделяют четыре класса задач Парето-оптимизации, соответствующих следующим четырем типам фронта Парето: выпуклые задачи; вогнутые задачи; выпукло-вогнутые задачи; разрывные задачи (рисунок 5.2).

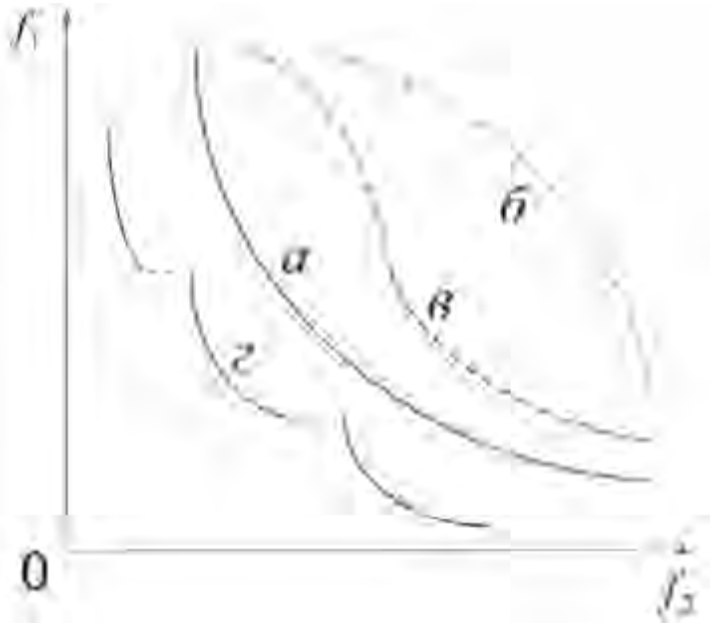


Рисунок 5.2 – Типы фронтов Парето: а) выпуклый; б) вогнутый; в) выпукло-вогнутый фронт; г) разрывный

Одной из проблем популяционных методов Парето-аппроксимации является формирование начальных состояний архивов A_F и A_X. С этой целью может быть использован сеточный метод, суть которого состоит в следующем. Покрываем область P некоторой сеткой с узлами $X_i, i = \overline{1,n}$. В каждом из этих узлов вычисляем значение вектор-функции F_i, среди векторов F_i выбираем недоминируемые векторы $\{F_{ij}^A\}$ и находим соответствующее им множество точек $\{X_{ij}^A\}$, где $F_{ij}^A = F(X_{ij}^A)$, $i_j = \overline{1,|A|}$.

Множества $\{X_{ij}^{A}\}$, $\{F_{ij}^{A}\}$ представляют собой искомую начальную дискретную аппроксимацию множества Парето и фронта Парето МО-задачи соответственно.

Известным вариантом сеточного метода является метод исследования пространства параметров [257], особенностью которого является использование специальных сеток, построенных на основе так называемых $ЛП_T$ последовательностей, обеспечивающих большую репрезентативность Парето-аппроксимации.

Для нахождения фронта Парето в задачах МО также широко используются генетические алгоритмы (ГА), так как их свойства наиболее подходят для таких типов задач [258-262]. Это обусловлено главным образом их параллельным или популяционным подходом к поиску решений, что позволяет устранить большинство трудностей и недостатков классических методов решения задач МО.

В настоящее время наиболее часто применяются следующие методы приближенного построения множества Парето на основе ГА:

- VEGA (Vector Evaluated Genetic Algorithm) [263];
- FFGA (Fonseca and Fleming's Multiobjective Genetic Algorithm) [264];
- NPGA (Niched Pareto Genetic Algorithm) [265];
- SPEA (Strength Pareto Evolutionary Algorithm) [259];
- NSGA-II (Non-dominated Sorting Genetic Algorithm-II) [266].

5.2 Парето-эволюционирующие ИНС прямого распространения

Следует отметить, что ГА для решения задачи МО во многом схож с алгоритмом, основанном на искусственных иммунных системах. Поэтому приведенный на рисунке 5.3 обобщенный алгоритм включает эти два

алгоритма, выделенные пунктирными линиями, и содержит следующие основные шаги [267-268]:

1. Создание начальной популяции.

1.1. Инициализация хромосомы каждой особи.

1.2. Оценивание начальной популяции.

2. Этап эволюции - построение нового поколения.

2.1. Отбор (селекция) кандидатов на скрещивание (клонирование).

2.2 Скрещивание (клонирование).

2.3. Мутация.

2.4. Оценивание новой популяции (полученных клонов).

3. Построение фронта Парето на основе выбранных критериев обучения.

4. Выбор единственного решения из полученного набора оптимальных решений с помощью некоторого информационного критерия после достижения критерия останова.

Рассмотрим некоторые шаги алгоритма более подробно.

Следует отметить, что на этапе оценки популяции с помощью алгоритмов МО и использовании в них в качестве целей различных функций приспособленности (фитнес-функций), возможна фильтрация зашумленных сигналов, что обеспечивает робастность получаемых оценок.

Следующий шаг работы обобщенного алгоритма – построение фронта Парето – может быть записан следующим образом.

1. Сгенерировать начальную обучающую выборку, состоящую из векторов входных переменных $\boldsymbol{x} \in \Omega$. Вычислить векторы значений целевых функций $F(\boldsymbol{x})$ для всех $\boldsymbol{x}$.

2. На основе обучающей выборки $\boldsymbol{x} \in \Omega$ и соответствующих значений $F(\boldsymbol{x})$ построить модели всех целевых функций $f_1(\boldsymbol{x}), f_2(\boldsymbol{x}), ..., f_k(\boldsymbol{x})$.

3. На основе полученных моделей $f_1'(\boldsymbol{x}), f_2'(\boldsymbol{x}),..., f_k'(\boldsymbol{x})$ с помощью выбранного алгоритма определить Парето-оптимальное множество PF^* решений задачи (5.8)-(5.10).

4. В точках полученного множества решений PF^* вычислить точные значения функций $f_1(\boldsymbol{x}), f_2(\boldsymbol{x}),..., f_k(\boldsymbol{x})$. Если критерий останова (получена требуемая точность моделей и построен фронт Парето либо осуществлено максимально допустимое число итераций) не выполняется, то все полученные значения добавляются в обучающую выборку и осуществляется возврат к шагу 2, на котором уточняются модели целевых функций.

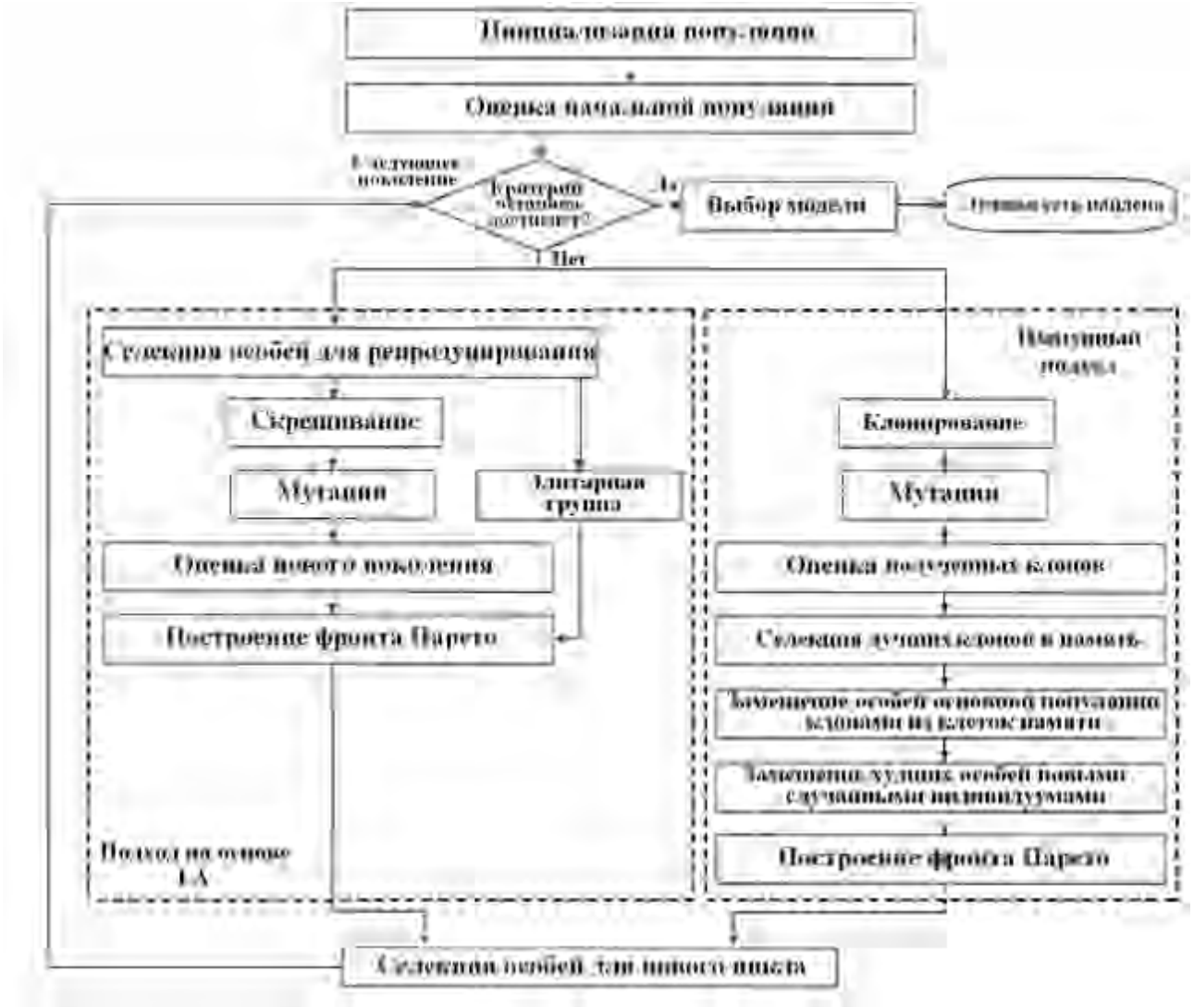


Рисунок 5.3 – Обобщенный алгоритм решения задачи МО

Важным этапом данного алгоритма является получение моделей целевых функций. Наличие данных моделей обусловлено тем, что в реальных системах получение значений целевых функций является весьма трудоемкой задачей. При решении же задачи МО подобные вычисления необходимо проводить многократно. В связи с этим, точные и быстродействующие модели целевых функций в значительной мере ускоряют задачу получения Парето-оптимальных решений. Получить такие модели можно с помощью радиально базисных сетей (РБС), которые, как уже неоднократно отмечалось, являются универсальными аппроксиматорами и при этом обладают простой структурой и отличаются высокой скоростью обучения.

Для нахождения модели нейросети, которая наилучшим образом балансирует ошибку модели и ее сложности с целью предотвращения чрезмерного недо- или переобучения. Наиболее целесообразным представляется использование информации об ошибке обучения, сложности и параметрах модели, уже полученной при построении самого фронта Парето, т.е. использование некоторого информационного критерия [70, 73, 85].

Следует отметить, что абсолютные значения таких критериев смысла не имеют – они указывают только на относительный порядок сравниваемых моделей. Считается, что минимум информационного критерия соответствует лучшей (оптимальной) сложности модели и чем меньше значение, тем лучше модель.

Кроме того, использование информационных критериев обеспечивает ограничение ненужного увеличения размера модели, приводящего к ее вырождению [157, 269].

Таким образом, выбор оптимального решения из фронта Парето является наиболее важным этапом всей процедуры аппроксимации (идентификации) нелинейной функции с помощью ЭРБС. Так как фронтом Парето обычно

представлен широкий набор возможных оптимальных решений, окончательный выбор модели должен быть достаточно точным и робастным.

5.3 Коэволюционные ГА

ГА являются быстрым, эффективным методом оптимизации, который способен решать широкий спектр задач. Однако, основным недостатком ГА является постоянное стремление к популяции, содержащей один локальный оптимум. Постоянное уменьшение генетического разнообразия популяции снижает способность ГА к поиску глобального оптимума и/или адаптации к изменяющимся параметрам целевых функций. Недавние успехи в области ГА, обусловленные тем фактом, что в природе коэволюция между видами весьма широко распространена, показывают, что введение экологических моделей и использование коэволюционных архитектур позволяет существенно замедлить или полностью устранить вероятную потерю разнообразия.

В природе конкуренция и сотрудничество играют решающую роль выживания для различных видов в условиях ограниченных ресурсов. Сотрудничество позволяет выжить слабым видам в среде, в которой различные виды конкурируют за ограниченные ресурсы. Различные виды конкурируют и сотрудничают с другими видами с целью обмена ресурсами. Различные виды имеют различные уровни приспособленности в соответствии с их генами, разнообразием популяции, а также в зависимости от условий окружающей среды.

Кооэволюционные ГА (КГА) заимствуют у природы механизмы кооперации и конкуренции с целью поддержания разнообразности популяций и, как следствие, получение лучших решений. Обычно в КГА ограничиваются использованием двух подпопуляций, но какого-либо ограничения на число подпопуляций не существует. КГА разделяют на два основных класса:

кооперативных и конкурирующих ГА [270], каждый из которых в свою очередь разбивается на ряд подклассов. Конкурирующая коэволюция состоит из следующих подклассов: 1) Конкуренция (виды соперничают друг с другом и успех одного вида является неудачей для другого); 2) аменсализм (неудача одного вида не оказывает влияния на другие виды, участвующие в коэволюции).

В кооперативной же эволюции выделяют: 1) мутуализм (виды сотрудничают с выгодой для себя); 2) комменсализм (выигрывает только один вид, а на остальные виды этот процесс не оказывает влияния); 3) Хищничество (паразитизм) (только хищники (паразиты) выигрывают, в то время как остальным видам причиняется вред).

Рассмотрим более подробно каждый из подклассов коэволюции.

5.3.1 Конкуренция.

Конкуренция является формой совместной эволюции, которая имеет обратную связь в рамках своего симбиоза. Идея конкуренции заключается в том, что существует некоторый ограниченный ресурс, имеющий решающее значение для выживания особей (конкурентов), который они должны иметь. Особи конкурируют за ограниченный ресурс и лишь одна или несколько из них его получают. Для решения практических задач обычно принимают, что конкуренты ведут себя как отдельные популяции с собственными функциями приспособленности. При этом если приспособленность одной популяции уменьшается, то это приводит к увеличению приспособленности другой популяции, и ее фитнес увеличивается. Таким образом, отношения между популяциями при конкуренции являются симметричными. Схема, отображающая симбиоз при конкуренции, показана на рисунке 5.4-а).

5.3.2 Аменсализм.

При аменсализме происходит симбиоз между двумя особями: хозяином и аменсалом. При этом хозяин пагубно влияет на аменсала и совершенно от него не зависит. Следует отметить, что предложенный Льюисом в [271] аменсализм почти никогда не упоминается в классификации отношений животных. Это обусловлено тем, что такие отношения считаются неестественными.

Примером аменсализма может быть бобер, влияющий на рыбу за счет строительства плотины. Построив плотину, бобер производит большое негативное воздействие на окружающую среду рыбы. При этом рыба вообще не влияет на бобра. Таким образом, бобер является хозяином, а рыба - аменсалом.

При аменсализме изменения в фитнесе хозяина производят к изменениям фитнеса аменсала. Если приспособленность аменсала увеличивается, то у аменсала, соответственно, фитнес будет уменьшаться. Если же приспособленность хозяина уменьшается, то фитнес аменсала увеличивается.

Схема симбиоза при аменсализме показана на рис. 5.4-б).

5.3.3 Мутуализм.

Мутуализм является формой совместной эволюции, которая содержит обратную связь в симбиозе. Мутуализм это отношения между двумя особями, которые называются симбионтами. Оба симбионта стремятся извлечь выгоду от другого. Следует отметить, что точного описания того, что такое мутуалистический симбиоз не существует в настоящее время [271, 272] в связи с многообразием противоречивых его форм.

Существует большое количество примеров мутуализма в природе. Вариации пространственных и временных зависимостей и генетических связей огромны и интересны.

При мутуализме, если фитнес одного из симбионтов увеличивается по каким-либо причинам, то это будет приводить к росту фитнеса и второго

симбионта. При уменьшении же фитнеса одного из симбионтов будет наблюдаться снижение приспособленности и у другого.

Схема симбиоза при мутуализме показана на рисунке 5.4-в).

5.3.4 Комменсализм.

При комменсализме также происходит симбиоз между двумя особями - хозяином и комменсалом. Комменсал при этом извлекает выгоду и возлагает на хозяина регуляцию своих отношений с внешней средой, но не вступает с ним в тесные взаимоотношения. Классическим примером комменсализма являются отношения между акулой и рыбой прилипало.

Поскольку хозяин не зависит от комменсала, то фитнес хозяина не изменяется непосредственно в результате изменения пригодности комменсала (т.е. отсутствует обратная связь). Однако комменсал получает выгоду от хозяина, поэтому изменение в его пригодности будет влиять и на комменсала. Если фитнес хозяина возрастает, то можно сделать вывод, что комменсал сможет извлечь большую выгоду от хозяина, тем самым увеличивая собственную приспособленность. Если же фитнес хозяина уменьшается по каким-либо причинам, то комменсал извлекает меньшую выгоду и тем самым уменьшается его пригодность. Схема симбиоза при комменсализме имеет вид, показанный на рисунке 5.4-г).

5.3.5 Хищничество.

Хищничество, как и другие обсуждаемые в данном разделе формы коэволюции, охватывает широкий спектр межпопуляционных отношений. Эта форма коэволюции также содержит в себе обратную связь в отношениях между популяциями. В данной форме коэволюции присутствуют хищник и его жертвы. Хищник, получает выгоду от добычи, а добыча в свою очередь пагубно влияет на хищника. Другим общим отношением в хищничестве является

отношение паразит-хозяин (паразит берет на себя роль хищника, а хозяин берет на себя роль жертвы). Пример отношений лисицы и зайца показывает характер хищничества.

Если фитнес хищника увеличивается, то приспособленность жертвы будет уменьшаться. В случае же, когда фитнес хищника уменьшается, то в результате этого приспособленность жертвы будет увеличиваться.

Если же фитнес жертвы увеличивается, то ожидаемая приспособленность хищника будет уменьшаться. В случае снижения фитнеса жертвы, фитнес хищника будет увеличиваться.

Динамика изменения численности популяций хищников и жертв описывается с помощью модели Лотка-Вольтерра

$$\frac{dx}{dt} = \alpha x - \beta xy;$$
$$\frac{dy}{dt} = -\gamma y + \delta xy,$$

где x – численность популяции жертв, а y – хищников.

При этом предполагается, что:

1. У популяции жертв всегда достаточно пищи.

2. Хищники питаются только добычей из популяции жертв.

3. Скорость изменения численности популяций пропорциональна их размерам.

4. Окружающая среда является статической.

Таким образом, изменение численности популяции жертв $(\frac{dx}{dt})$ обусловлено:

- увеличением в связи с рождением новых особей (пропорционально численности популяции, αx) и

- уменьшением, вызванным хищничеством (которое пропорционально количеству встреч хищник-жертва, βxy).

Изменение же численности популяции хищников ($\frac{dy}{dt}$) обусловлено:

- уменьшением из-за естественной смерти (пропорционально численности популяции, γy) и

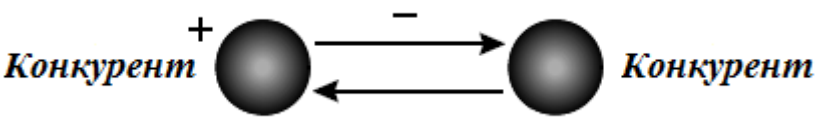
- увеличением, допускаемым в пищевой цепочке (пропорционально количеству встреч хищник-жертва, δxy).

Схема симбиоза при хищничестве показана на рисунке 5.4-д).

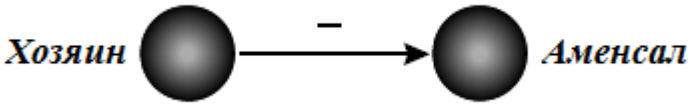
Из всех рассмотренных типов коэволюции при проектировании ЭИНС наибольшее распространение получили конкуренция и хищничество.

5.4 Коэволюционирующие ИНС

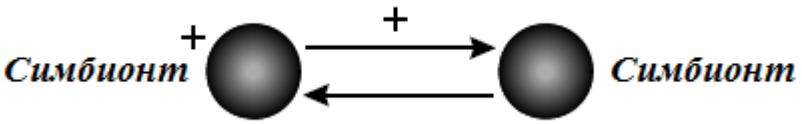
Как уже отмечалось, использование ИНС требует решения двух проблем: определения оптимальной архитектуры сети и оптимальных значений ее параметров. Следует отметить, что в последнее время указанные задачи все чаще стремятся решать одновременно с помощью коэволюционирующих адаптивных систем, т.е. систем, состоящих из разнообразных эволюционирующих групп особей, которые действуют совместно для выполнения сложных вычислений или выработки совместного эффективного поведения. Подобные системы широко распространены в природе, в социальных системах (экономических и политических) и в искусственных адаптивных системах (нейронных, иммунных и др.).



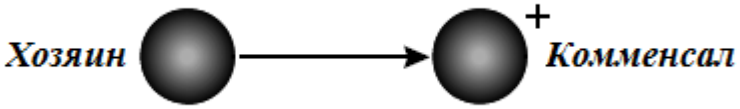
а)



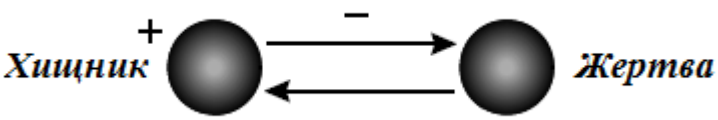
б)



в)



г)



д)

Рисунок 5.4 – Схемы коэволюции

Отличительными особенностями коэволюционнирующих систем является то, что, во-первых, популяции могут иметь разный размер; во-вторых, эволюция в разных популяциях может идти на основе различных алгоритмов; в-третьих, альтернативные решения из разных популяций могут быть использованы для решения задач, отличающихся физической размерностью [273].

Типичные ИНС характеризуются, как правило, кооперативным поведением, благодаря суммированию в выходном слое сигналов, поступающих со скрытых слоев нейронов. В самоорганизующихся же ИНС (например, в сети Кохонена) реализуется конкурентное поведение с помощью механизма разрешения конфликтов, обеспечивающего выбор лишь одного победителя в любой ситуации («winner takes all»).

Важно отметить, что оба типа систем (на основе сотрудничества и конкуренции) могут нуждаться в коэволюции. Даже в конкурентной системе нескольким элементам может потребоваться совместное существование и коэволюция друг с другом для формирования полного набора действий. Необходимо, тем не менее, отметить, что структуры, возникающие в кооперативных системах, могут существенно отличаться от тех, которые развиваются в конкурентных системах [274].

Кооперативная коэволюция [275] предполагает разбиение основной задачи на несколько меньших подзадач, для решения которых использует стандартную эволюционную оптимизацию с целью постепенного решения основной проблемы. При этом подзадачи рассматриваются как особи или модули и представляются в виде суб-популяций. Каждая суб-популяция эволюционирует отдельно и их сотрудничество осуществляется только при оценке фитнеса для соответствующих особей в каждой суб-популяции.

Одной из основных причин использования кооперативной коэволюции в ИНС является то, что она обеспечивает более разнообразные решения по сравнению с другими ЭА, основанными на одной популяции [276]. При этом,

однако, возникает серьезная проблема разложения задачи на подзадачи, обусловленная необходимостью объединения в группы взаимодействующих компонент с целью минимизации взаимодействия между суб-популяциями [277]. В ИНС разложение задачи на подзадачи осуществляются на уровне синапсов [278] и на уровне нейронов [277]. При разложении на уровне нейронов число суб-компонент равно общему числу скрытых и выходных нейронов [279]. Субкомпоненты при таком подходе образуют суб-популяции. Пригодность каждой особи для конкретной популяции оценивается совместно с недоминирующими особями из других суб-популяций [280]. Эти методы разложения задачи показали свои сильные и слабые стороны при решении различных типов задач с помощью нейронных сетевых архитектур.

Ниже предлагается эволюционный алгоритм определения архитектуры нейронных сетей прямого распространения и их обучения, основанный на стратегиях кооперации и конкуренции. Алгоритм обучения реализует среду, способствующую сотрудничеству и конкуренции популяций, в которых каждая особь представляет собой ИНС прямого распространения, а вся совокупность популяций несет ответственность за окончательное решение поставленной задачи.

5.4.1 Многокритериальный метод обучения ИНС на основе кооперативной коэволюции.

Большое количество многокритериальных задач оптимизации (оптимизация функций больших масштабов, обучение глубоких нейронных сетей, обучение рекуррентных нейронных сетей) привело к появлению множества их оптимальных решений, известных как Парето-оптимальные [252, 266-267] . Среди решений, лежащих в Парето-оптимальном фронте, называемых недоминирующими, ни одно не является лучше остальных с учетом всех целевых функций [281]. Любое недоминирующее решение в рамках Парето-

оптимального набора можно улучшить по какой-либо целевой компоненте лишь путем ухудшения, по крайней мере одной из остальных его целевых составляющих [282].

Базовая модель коэволюции на основе кооперации показана на рисунке 5.5. Хотя на данном рисунке показаны три популяции, фактическое их число в экосистеме может быть иным. При таком подходе каждый вид эволюционирует в своей популяции и адаптируется к окружающей среде путем многократного применения ЭА. На рисунке показан этап оценки пригодности ЭА с точки зрения каждого из этих трех видов. Для реализаций такой модели может быть использована как последовательная схема оценки популяций, так и параллельная. Оценка приспособленности особей одного вида проводится с участием представителей и других видов. Существует большое количество способов выбора представителей популяций для сотрудничества. В некоторых случаях достаточно просто выбрать текущую лучшую особь от каждого вида в качестве представителя всей популяции. В других случаях такой подход может быть слишком «жадным» и альтернативные стратегии являются более предпочтительными. Так, например, особь от каждого вида может быть выбрана случайным образом или может быть применен более естественный подход, при котором представители выбираются недетерминированно на основе их пригодности.

Рассмотрим более подробно многокритериальный коэволюционный метод обучения нейронных сетей на основе кооперации. При таком подходе каждая нейронная сеть обучается с учетом нескольких целей (например, точность и сложность сети). Решения, которые являются приемлемыми для обеих целей, переносятся и сохраняются. При кооперативной коэволюции лучшие особи используются как представители своей суб-популяции, а особи, оставшиеся внутри суб-популяции, ранжируются в соответствии со значением их фитнес-функций.

Кооперативная оценка особи в суб-популяции осуществляется путем объединения данной особи со случайными недоминирущими особями в остальных суб-популяциях. Затем особь оценивается для каждой заданной цели. После оценивания всех особей в пределах суб-популяций, оценивается вся суб-популяция и определяются ее недоминирующие особи (по аналогии с NSGA-II [279]). Во время первой итерации, когда недоминирующие особи неизвестны, особи оцениваются путем объединения со случайными особями из других суб-популяций. Цикл завершается, когда все суб-популяции эволюционировали. После этого происходит их переоценка с целью проведения оценки новых особей, возникщих в процессе эволюции.

Следует отметить, что для МП и РБС подходы к разбиению сети на суб-популяции несколько отличаются. Детально процесс разбиения глобальной задачи многокритериального обучения МП на подзадачи, решаемые в отдельных суб-популяциях и оценки их пригодности, поясняется на рис.5.6. На данном рисунке представлен общий вид всех суб-популяций в рамках коэволюции на основе кооперации [270].

В начале работы алгоритма каждая особь в суб-популяции стремится минимизировать каждую из целей фитнес-функции. Для оценки пригодности особи она объединяется со случайными недоминирующими особями из остальных суб-популяций. Затем все особи объединяются для формирования общего решения. Выбранные особи из суб-популяций объединяются, формируя общую хромосому (генотип) и отображаются в нейронную сеть (фенотип). Полученная ИНС затем оценивается с учетом различных целей. После этого значение фитнес-функции полученной результирующей сети участвует в вычислении фитнес-функций тех особей (представителей суб-популяций), которые принимали участие в формировании данной сети. Затем весь процесс начинается снова для следующей особи.

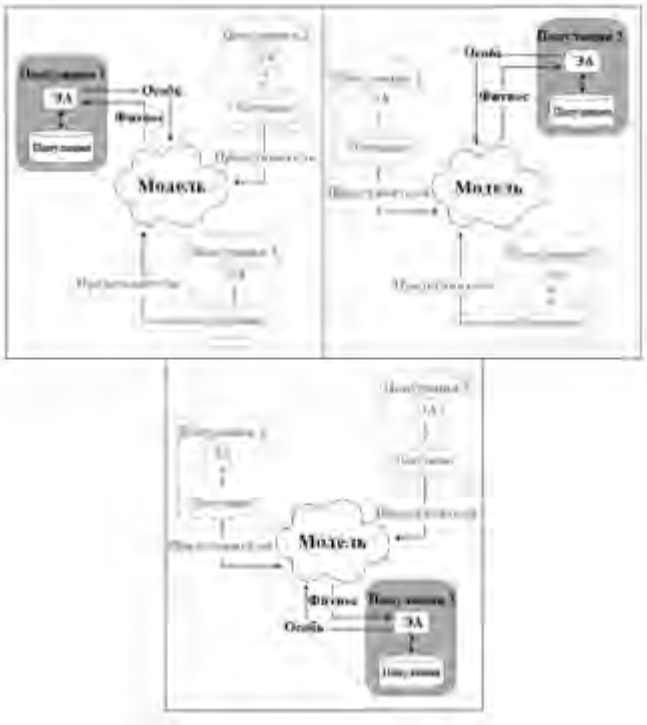


Рисунок 5.5 – Коэволюция на основе кооперации

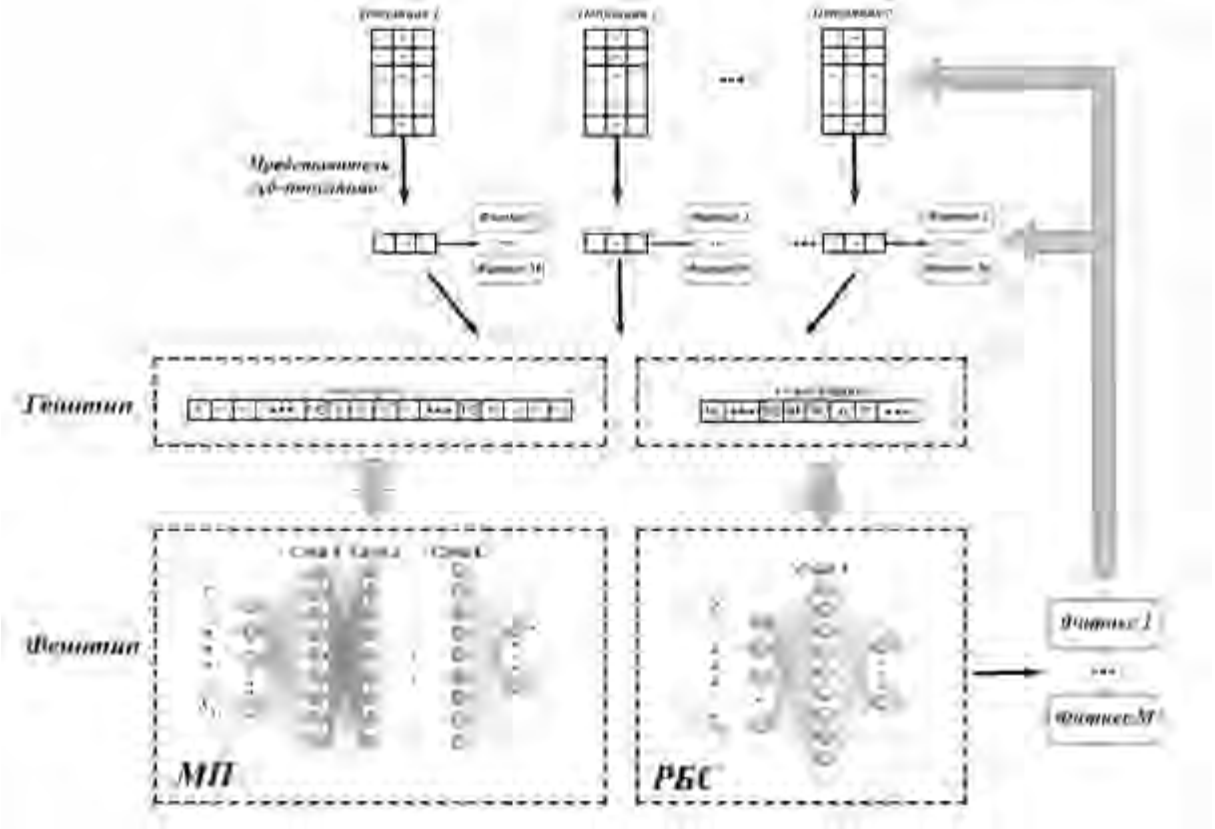


Рисунок 5.6 – Процесс разбиения глобальной задачи многокритериального обучения ИНС на подзадачи

Следует отметить, что для всех типов сетей во всех суб-популяциях используются общие эволюционные процедуры (инициализация популяции, оценка популяции, селекция, скрещивание, мутации), а различия заключаются лишь в способе кодирования структуры и параметров той или иной ИНС в виде хромосомы (см.рис.2.3) [118, 145, 224]. Схема работы такого эволюционного алгоритма настройки ИНС приведена на рис.2.2.

В работе [283] предложен несколько иной поход к применению механизма коэволюции к настройке ИНС по получивший название COVNET. COVNET является кооперативной коэволюционирующей моделью, то есть, несколько видов совместно коэволюционируют. Каждый вид при этом является подсетью и представляет собой частичное решение проблемы. Комбинация нескольких особей из разных видов составляет ИНС, которая должна быть применена к конкретной проблеме. Популяция подсетей, называемых нодулями, состоит из нескольких суб-популяций, которые эволюционируют независимо друг от друга. Каждая из таких суб-популяций представляет собой отдельный вид, а ключевым элементом предложенной модели является комбинация особей из этих различных суб-популяций, коэволюционирующих вместе.

Нодуль является подсетью, образованной с помощью набора нейронов с свободными связями между ними, соединений этих нейронов с нейронами входного и выходного слоев и не имеет соединений с каким-любо нейроном, принадлежащем другому нодулю. Пример нодуля показан на рис. 5.7. Входные и выходные слои для всех нодулей являются общими, они являются входными и выходными слоями результирующей сети. Следует отметить, что генотип нодуля имеет взаимно-однозначное отображение в фенотип. ИНС при этом можно определить как совокупность нодулей, т.е. ИНС является сочетанием конечного числа нодулей, а выходом сети является сумма выходов всех входящих в нее нодулей. На практике для упрощения алгоритма обычно

полагают, что все сети популяции должны иметь одинаковое фиксированное на протяжении всего процесса эволюции количество нодулей.

Процесс эволюции субкомпонент должен решать четыре основные задачи: разложение основной задачи на подзадачи, обеспечение взаимозависимости

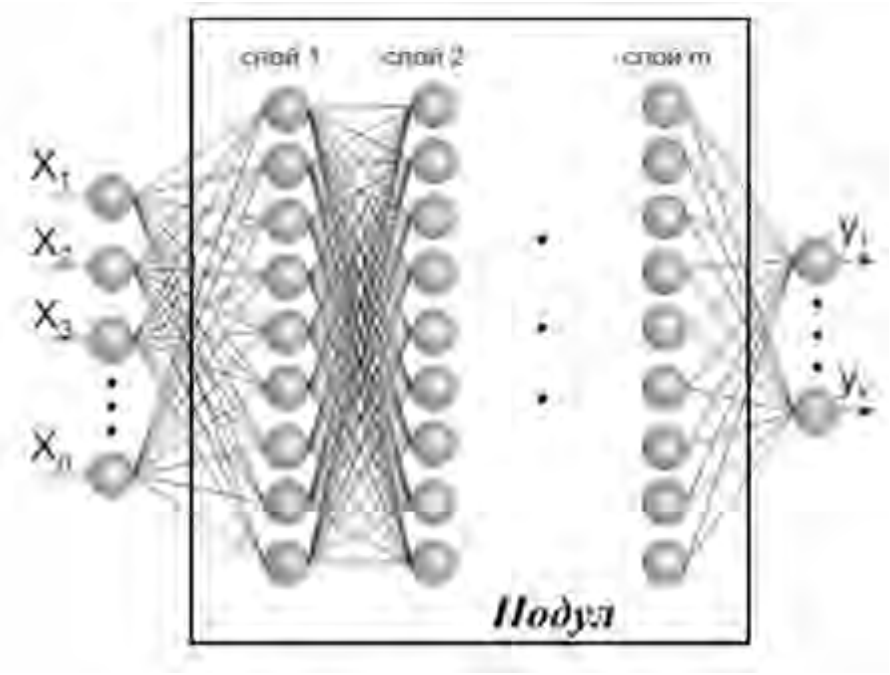


Рисунок 5.7 – Модель нодула

между субкомпонентами, оценку суб-популяций (присвоение коэффициентов доверия) и обеспечение разнообразия популяций.

Кооперативная коэволюция позволяет решить данные задачи естественным образом. Так как каждая популяция будет производить особи определенного вида, которые должны сотрудничать с целью получения вознаграждения высокими значениями пригодности, то нет необходимости в каких-либо априорных знаниях для разложения проблемы вручную. Взаимозависимость среди суб-компонент обусловлена тем, что фитнес каждой особи зависит от того, насколько хорошо она сотрудничает с членами других видов. Оценка

популяций производится в соответствии с моделями, рассмотренными ранее (см. Раздел 1).

Разнообразие поддерживается в процессе эволюции благодаря тому, что каждый вид развивается без обмена генетическим материалом с другими видами. Это являетсяважным аспектом данной коэволюционной модели. Обмен генетическим материалом между двумя различными видами (субпопуляциями), как правило, производит нежизнеспособное потомство. Кроме того, смешение генетического материала может привести к уменьшению разнообразия популяций.

Алгоритм настройки ИНС с помощью подхода COVNET показан на рис.5.8.

5.4.2 Многокритериальный метод обучения ИНС на основе конкурентной коэволюции.

Рассмотрим алгоритм обучения ИНС на основе конкурентной коэволюции [284]. В нем можно выделить следующие основные шаги.

1. Инициализация. Создается *N* популяций, состоящих из *m* особей (ИНС) каждая. Каждая популяция $P_i = \{H_1, H_2, .., H_m\}$ инициализируется случайным образом. Каждая особь в популяции при этом получает свое уникальное описание, закодированное в хромосоме $H_j = \{h_{1j}, h_{2j}, .. h_{Lj}\}$, которая состоит из *L* генов, где $h_{ij} \in [w_{\min} w_{\max}]$ - значение *i*-го гена *j*-ой хромосомы ($w_{\min}$ – минимальное, и $w_{\max}$ – максимальное допустимые значения соответственно). Следует отметить, что длина хромосомы зависит от размерности аппроксимируемой функции и максимально допустимого количества нейронов.

2. Первый шаг эволюции. Все *N* популяций обучаются на полном наборе входных сигналов.

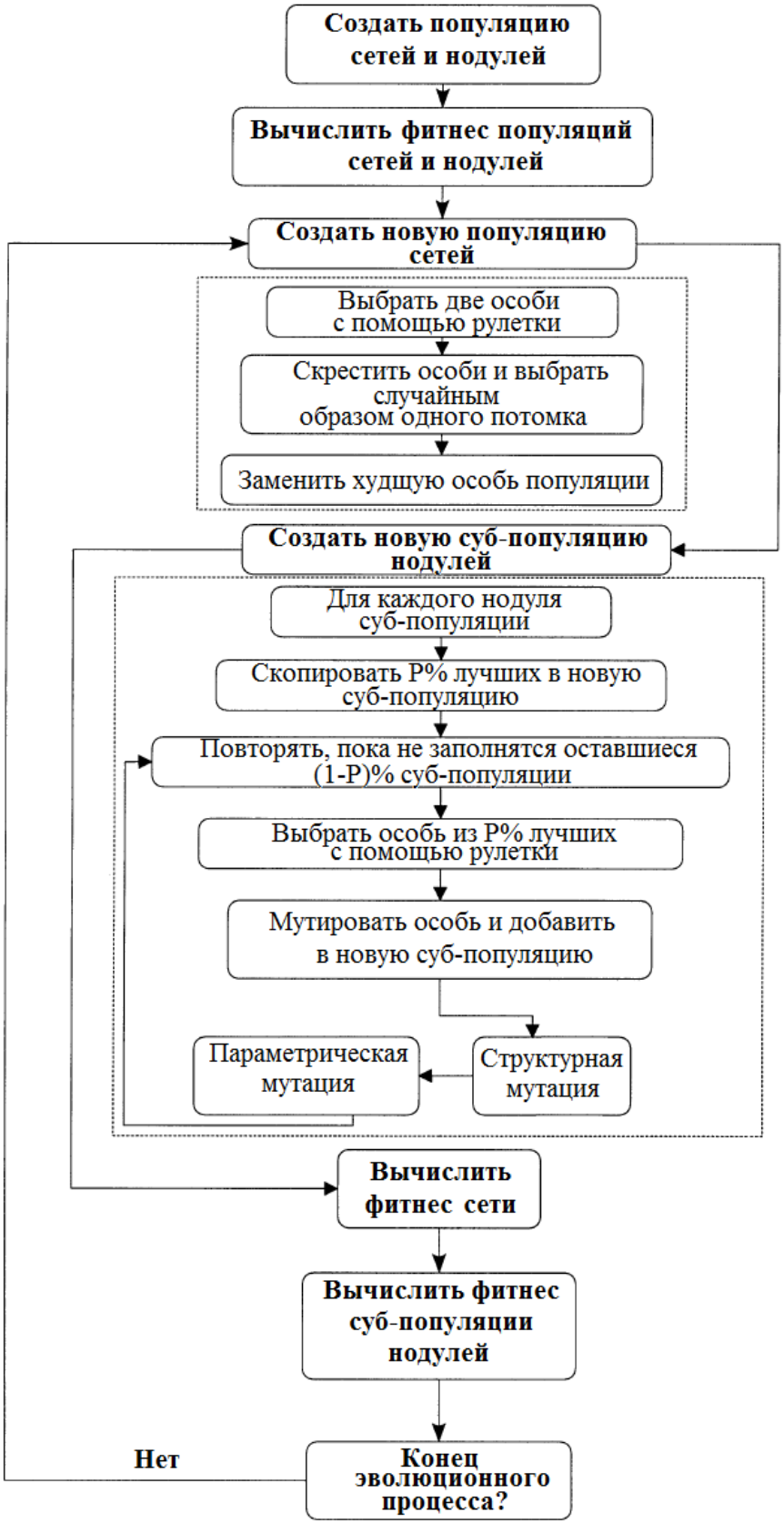


Рисунок 5.8 – Алгоритм настройки ИНС с помощью подхода COVNET

3. Ранжирование. Все $N{\cdot}m$ особей ранжируются (от 0 до $N{\cdot}m-1$) в зависимости от их фитнеса F_i, i =0,…, $N{\cdot}m-1$, вычисленного во время обучения. Ранжирование популяций выполняется на основе фитнеса первых M особей после индивидуального ранжирования при $M \geq m$. M будет отлично от m только тогда, когда $F_m = F_{m-1}$, в этом случае при ранжировании необходимо учесть все особи, имеющие одинаковый фитнес $F_{m-1} = F_m = \ldots = F_{M-1}$.

4. Присвоение коэффициентов доверия. Каждой из M особей присваивается коэффициент, вычисляемый следующим образом: $\Gamma_i = e^{(M-l)/M}$, i =0,…, $m-1$. Если k особей имеют одинаковый фитнес F_j , F_{j+1},…, F_{j+k-1} с $0 \leq j,\ j + k - 1 \leq M - 1$, им будет присвоен одинаковый коэффициент доверия

$$\Gamma = \frac{1}{k}\left(\sum_{l=0}^{k-1} e^{(M-l)/M}\right).$$

Каждой из N популяций присваивается коэффициент доверия, эквивалентный сумме коэффициентов доверия, присвоенных ее особям. Следует отметить, что если популяция не имеет своих представителей среди M лучших особей, ей присваивается коэффициент доверия равный 0.

5. Проверка критерия исключения популяции. Если ранг популяции не изменялся в течение предыдущих D эпох, то популяция исключается на 6 шаге, в противном случае осуществляется переход к шагу 2. Для ускорения процесса обучения D уменьшается на 1 каждые P итераций (параметры D и P задаются на этапе инициализации алгоритма). Возможен также подход, при котором через каждые D шагов исключается популяция с наименьшим рейтингом независимо от того, менялся он на последних итерациях или нет.

6. Исключение популяции. Если выполняется критерий исключения популяции, то геномы особей исключаемой популяции распределяются между

оставшимися $N-1$ популяциями. После уменьшения N на единицу, алгоритм продолжает работу с шага 2, если $N > 1$, до тех пор, пока не останется лишь одна популяция, состоящая из $N \cdot m$ особей. Оставшаяся популяция обучается до выполнения условия останова.

5.4.3 Многокритериальный метод обучения ИНС на основе комбинированного коэволюционного подхода.

Предложенный в работе [270] алгоритм CCC-GA (Competitive-cooperative coevolutionary – GA) объединяет в себе два рассмотренных выше коэволюционных метода. При этом, фаза кооперации на основе конкуренции начинается с N популяций, содержащих по m особей каждая, каждая популяция эволюционирует с помощью ГА с перекрывающимися популяциями и заканчивается с 1 популяцией, содержащей $N \cdot m$. Таким образом, менее приспособленные популяции постепенно поглощаются более приспособленными. При этом для каждой популяции применяется ГА с различными параметрами мутаций и вероятностями скрещивания. Алгоритм CCC-GA можно записать следующим образом:

1) Фаза конкуренции (см. п. 5.4.2).
    - 1.1) Инициализация
    - 1.2) Первый шаг эволюции.
    - 1.3) Ранжирование.
    - 1.4) Присвоение коэффициентов доверия.
    - 1.5) Проверка критерия исключения популяции.
    - 1.6) Исключение популяции.
2) Фаза сотрудничества.
    - 2.1) Интеграция победителя. Победивший в конкуренции вид кооперативно коэволюционирует с лучшими представителями предыдущих выигравших видов, пока СКО не уменьшается менее

чем на δ в I последовательных итерациях. В конце итерационного процесса геном с самой высокой пригодностью используется для замещения худшей особи популяции.

2.2) Переобучение. После того, как новая сеть добавляется в популяцию, предпринимается попытка переобучить сразу все виды, созданные до этого момента. Во время переобучения, каждый вид вносит свой вклад в общее решение. Виды переобучаются в случайном порядке, кроме новейших видов, которые всегда переобучаются последними. Переобучение осуществляется с использованием того же критерия останова, как в пункте 2.1. Если вид не может найти более подходящее решение в процессе переобучения, то предыдущее будет сохранено.

5.4.4 Робастность коэволюционирующих систем.

При разработке коэволюционирующих систем следует различать два типа их робастности: локальную и глобальную. Обычно ограничиваются исследованием проблем глобальной робастности разрабатываемой системы, опуская влияние локальных возмущений, так как после получения окончательного решения все промежуточные популяции теряют свое значение. При разработке коэволюционирующей системы обучения ИНС разработчики обычно заинтересованы не в поиске лишь некоторой оптимальной популяции, а в поиске популяции, не только хорошо решающей поставленную задачу, но и устойчивой к отклонениям в поведении отдельных ее членов. Эта форма устойчивости является важной, поскольку на практике возможности членов популяции, и даже состав самой популяции, часто изменяются с течением времени. Таким образом, например, в рамках кооперативной коэволюции, робастность системы будет определяться возможностью входящих в нее суб-популяций осуществлять поиск таких решений, которые смогут эффективно

сотрудничать с широким диапазоном решений, получаемых в других суб-популяциях. Более детально такой подход рассматривается в работе [285].

5.5 Особенности решения задачи многокритериального обучения при использовании РБС

В общем виде алгоритм построения эволюционирующей РБС может быть записан следующим образом:

1) Инициализация популяции РБС
2) Настройка параметров РБС с помощью ГА
3) Оценка пригодности РБС
4) Применение генетических операторов (селекция, скрещивание, мутация)
5) Замещение худших особей популяции
6) Проверка критерия останова. Если условие останова не удовлетворяется, переход к п. 3, иначе – к п.7.
7) Применение тонкой настройки (например, с помощью алгоритма Левенберга-Марквардта.

При многокритериальном обучении сети РБС происходит два основных процесса, аналогичных обучению МП: нейроны (БФ) внутри суб-популяций конкурируют за право принимать участие в формировании результирующей нейронной сети, и отобранные нейроны осуществляют стратегию сотрудничества при генерации конечного решения.

Для оценки роли каждого нейрона (БФ) в кооперативно-конкурентной среде требуется некоторый механизм их оценки. Для этого в работе [286] предлагается определять следующие три параметра: вклад - a_i, ошибку - e_i и перекрытие - o_i.

Вклад a_i для i-ой РБС φ_i ($i = 1...m$), определяется с учетом ее веса w_i, и количества образцов обучающей выборки, находящихся внутри ее рецептивного поля, определяемого радиусом σ_i . РБС с небольшим значением веса и небольшим количеством входных сигналов внутри этого рецептивного поля будет соответственно вносить несущественный вклад в общий результат сети:

$$a_i = \begin{cases} |w_i|, \text{ если } \sigma_i > q; \\ |w_i|(\sigma_i / q), \text{ в противном случае,} \end{cases}$$

где q – среднее значение всех σ_i минус их стандартное отклонение.

Измерение параметра ошибки e_i каждой БФ (нейрона) основано на подсчете всех ошибок при обработке обучающих пар внутри ее рецептивного поля:

$$e_i = \frac{\sum_{j=1}^{N} \left(y_j^* - y_j \right)^2}{N},$$

где y_j^* и y_j - соответственно желаемое и реальное значения выходного сигнала сети; N – количество обучающих пар внутри рецептивного поля i-ой БФ (нейрона).

Степень перекрытия i-ой БФ φ_i остальными БФ характеризуется параметром o_i, вычисляемым следующим образом:

$$o_i = \sum_{j=1}^{M} o_{ij};$$

$$o_{ij} = \begin{cases} \left(1 - \dfrac{\|\phi_i - \phi_j\|}{d_i}\right), \text{если} \|\phi_i - \phi_j\| < d_i; \\ 0, \text{в противном случае,} \end{cases}$$

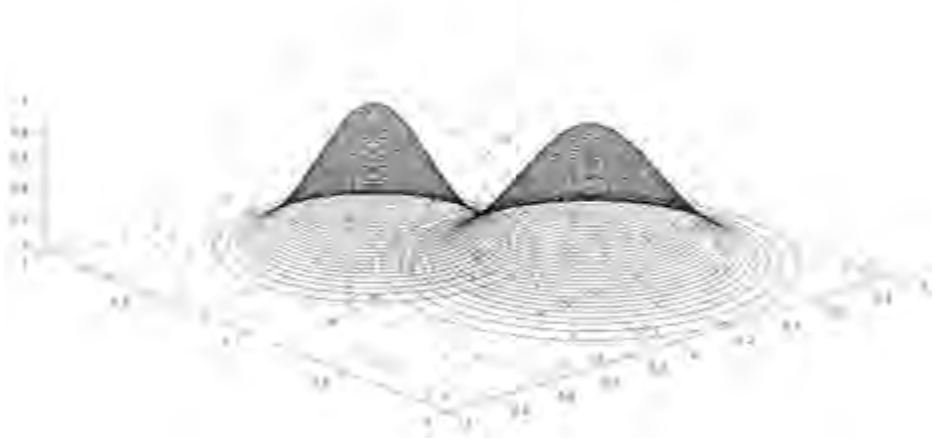
где o_{ij} определяет степень перекрытия i-ой БФ φ_i с j-ой φ_j.

На рис.5.9 показан пример перекрывающихся (а) и неперекрывающихся (б) БФ (крестиками обозначены поступающие на сеть входные сигналы). Следует отметить, что стремление минимизировать степень перекрытия БФ может, с одной стороны, приводить к повышению точности работы сети, но с другой – к увеличению количества нейронов скрытого слоя.

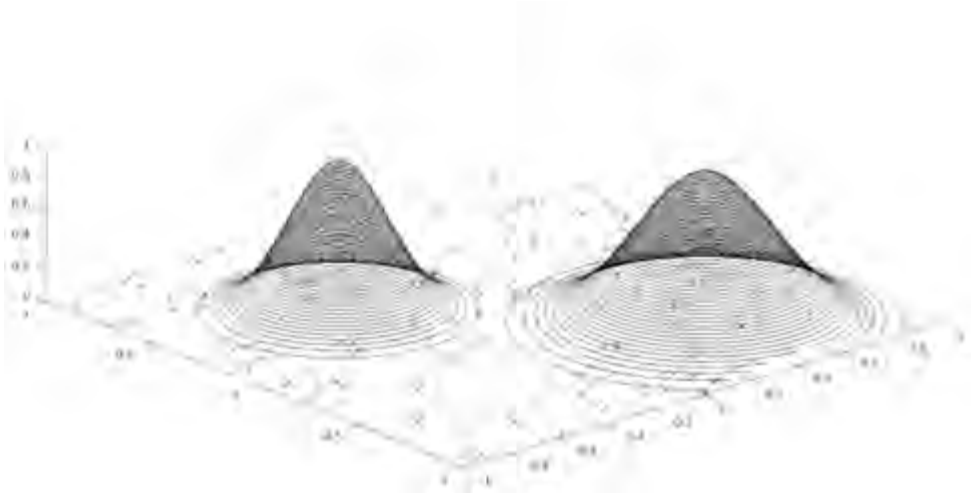
5.6 Многокритериальное обучение ИНС с использованием алгоритмов кластеризации

Существует также несколько иной подход к многокритериальному обучению ИНС. Он основан на разбиении пространства входных сигналов на кластеры на основе некоторого алгоритма кластеризации и многокритериальном обучении суб-популяций ИНС в каждом полученном кластере. Особи-победители в каждой суб-популяции избираются для формирования конечного решения.

Т.Кохоненом был предложен ряд алгоритмов решения задачи кластеризации, наиболее эффективным из которых является оптимальный алгоритм векторного квантования OLVQ [287-288], в соответствии с которым адаптация вектора весов сети происходит по правилу



а)



б)

Рисунок 5.9 – Пример перекрывающихся (а) и неперекрывающихся (б) БФ

$$\mathbf{w}_c(k+1)=\begin{cases}\mathbf{w}_c(k)+\alpha_c(k)[\mathbf{x}(k)-\mathbf{w}_c(k)], & \text{если } \mathbf{w}_c \text{ и } \mathbf{x} \text{ принадлежат} \\ & \textit{одному классу;} \\ \mathbf{w}_c(k)-\alpha_c(k)[\mathbf{x}(k)-\mathbf{w}_c(k)] & \textit{в противном случае;}\end{cases} \qquad (5.11)$$

$$\mathbf{w}_j(k+1)=\mathbf{w}_j(k) \qquad \textit{для всех } j\neq c\,.$$

При этом определение оптимального $\alpha_c^{onm}(k)$ происходит следующим образом.

Запишем (5.11) в виде

$$\mathbf{w}_c(k+1) = \mathbf{w}_c(k) + S(k)\alpha_c(k)\left[\mathbf{x}(k) - \mathbf{w}_c(k)\right], \qquad (5.12)$$

где

$$S(k) = \begin{cases} +1, & \textit{если } w \textit{ и } x \textit{ принадлежат одному классу;} \\ -1, & \textit{если } w \textit{ и } x \textit{ принадлежат разным классам.} \end{cases}$$

Следует отметить, что $\boldsymbol{x}(k)$ несет информацию о вновь поступившем образе, а информация о всех предыдущих входных сигналах содержится в $\mathbf{w}_c(k)$, поскольку

$$\begin{aligned} &\mathbf{w}_c(k+1) = \left[1 - S(k)\alpha_c(k)\right]\mathbf{w}_c(k) + S(k)\alpha_c(k)\mathbf{x}(k) = \\ &= \left[1 - S(k)\alpha_c(k-1)\right]\left[1 - S(k)\alpha_c(k)\right]\mathbf{w}_c(k-1) + \\ &+ S(k)\alpha_c(k)\left[1 - S(k)\alpha_c(k-1)\right]\mathbf{x}(k-1) + S(k)\alpha_c(k)\mathbf{x}(k). \end{aligned} \qquad (5.13)$$

Величины коэффициентов обучения $\alpha(k), \alpha(k-1), \ldots$ отражают влияние соответствующих входных образов на корректируемый вектор w_c. Из (5.13) видно, что влияние на $w_c(k+1)$ образа $\boldsymbol{x}(k)$ определяется весом $\alpha_c(k)$, образа $\boldsymbol{x}(k-1)$ – весом $\alpha_c(k-1)\left[1 - S(k)\alpha_c(k)\right]$ и т.д. Распределение весовых векторов $\boldsymbol{w}$ является статистически оптимальным, если при $k \to \infty$ влияние всех обучающих образов на изменение вектора w_c одинаково. Формально это можно записать так

$$\alpha_c(k) = \left[1 - S(k)\alpha_c(k)\right]\alpha_c(k-1). \qquad (5.14)$$

Так как на двух соседних тактах суммирующее влияние соответствующих входных векторов на весовой вектор одинаково и это справедливо для всех моментов времени k, то по индукции можно показать, что при $k \to \infty$ влияние всех входных векторов на $\boldsymbol{w}$ одинаково. Из уравнения (5.14) имеем:

$$\alpha_c(k) = \frac{\alpha_c(k-1)}{1 + \alpha_c(k-1)S(k)}. \qquad (5.15)$$

Хотя $\alpha_c(k)$ принципиально может возрастать вследствие того, что на некоторых тактах $S(k) = -1$, значение этого коэффициента не должно быть больше единицы.

Обычно в качестве начального значения выбирают $\alpha_c(0) = 0,3.$

Применение данного алгоритма позволяет разбить множество векторов x размерности Nx1 на конечное число классов M, $M<N$, каждому из которых присваивается свой код (назначается опорный представитель) $w_i (i = \overline{1, M}).$ Множество всех w_i образует *кодовое множество (кодовую книгу)* классификатора.

Векторное квантование осуществляется по методу "*ближайшего соседа*", причем под "ближайшим" понимается вектор, удовлетворяющий различным требованиям. Если в качестве такового выбирается вектор, находящийся от данного на минимальном евклидовом расстоянии, имеем классификатор, называемый в литературе *Voronoi-классификатор* [1].

При предъявлении подлежещего распознаванию входного образа x_i выдается представитель w_i того класса, к которому относится x_i. Границы между классами образуют медиатриссы между опорными преставителями.

В общем виде алгоритм многокритериального обучения ИНС с использованием алгоритмов кластеризации может быть записан следующим образом:

1) Инициализация популяции центров кластеров (случайное либо равномерное разбиение).
2) Инициализация популяций ИНС (одна популяция ИНС для каждого кластера).
3) Коэволюционное обучение популяций ИНС и кластеров на основе сотрудничества.
4) Если в кластере достигнут критерий останова, то выдвижение представителя популяции для формирования конечного решения.
5) Формирование конечного решения с помощью особей-победителей в каждом кластере.

5.7 Имитационное моделирование

5.7.1 Тестирование алгоритмов многокритериальной оптимизации.

Достаточно сложной задачей, требующей отдельного исследования, является задача тестирования алгоритмов многокритериальной оптимизации. Тестовые функции должны обладать рядом свойств, позволяющих с высокой точностью оценить эффективность разработанного алгоритма. В работе [289] предложена общая схема построения таких тестовых функций. Идея метода состоит в том, чтобы составить из некоторых базисных функций более сложные тестовые функции с хаотически расположенными глобальными оптимумами и несколькими случайно расположенными локальными оптимумами.

Составная тестовая функция при таком подходе вычисляется по формуле

$$F(x) = \sum_{i=1}^{n} w_i \left(f_i \left((x - o_i + o_{iold}) \lambda_i M_i \right) + b_i \right) + b, \qquad (5.16)$$

где w_i - вес каждой функции $f_i(x)$, вычисленный следующим образом:

$$w_i = \exp\left(- \frac{\sum_{k=1}^{D} \left(x_k - o_{ik} + o_{ikold} \right)^2}{2D\sigma_i^2} \right),$$

$$w_i = \begin{cases} w_i, если\ w_i = \max(w); \\ w_i (1 - (\max(w))^n), \end{cases}$$

после вычисления всех весов осуществляется их нормализация

$$w_i = \frac{w_i}{\sum_{i=1}^{n} w_i};$$

n – число базисных функций; σ_i^2 - радиус i-ой базисной функции; λ_i - параметр, используемый для масштабирования функций; o_i - определяет положение локальных и глобального минимумов; b_i - определяет какой из оптимумов будет глобальным (наименьшее значение b_i соответствует глобальному оптимуму); D - размерность x; M - ортогональная матрица поворота для каждой $f_i(x)$.

С помощью параметров o_i и b_i глобальный оптимум может быть расположен в любой заранее заданной точке.

В качестве базисных могут быть использованы, например, следующие тестовые функции:

- сферическая функция:

$$f(x)=\sum_{i=1}^{D}x_i^2, x\in[-100,100]^D; \tag{5.17}$$

- функция Растригина

$$f(x)=\sum_{i=1}^{D}\left(x_i^2-10\cos(2\pi x_i)+10\right), x\in[-5,5]^D; \tag{5.18}$$

- функция Вейерштрасса

$$\begin{aligned} f(x)=&\sum_{i=1}^{D}\left(\sum_{k=0}^{k\max}\left[a^k\cos\left(2\pi b^k\left(x_i+0.5\right)\right)\right]\right)-\\ &-D\sum_{k=0}^{k\max}\left[a^k\cos\left(\pi b^k\right)\right], a=0.5, b=3,\\ &k_{\max}=20, x\in[-0.5,0.5]^D; \end{aligned} \tag{5.19}$$

- функция Гриванка

$$\begin{aligned} &f(x)=1+\sum_{i=1}^{D}\frac{x_i^2}{4000}-\prod_{i=1}^{D}\cos\left(\frac{x_i}{\sqrt{i}}\right),\\ &x\in[-600,600]^D; \end{aligned} \tag{5.20}$$

- функция Экли

$$f(x) = -20\exp\left(-0.2\sqrt{\frac{1}{D}\sum_{i=1}^{D}x_i^2}\right) - \exp\left(\frac{1}{D}\sum_{i=1}^{D}\cos(2\pi x_i)\right) + 20 + e, x \in [-32,32]^D; \quad (5.21)$$

Эксперимент 1. В работе для тестирования предложенного метода использовались в качестве нелинейной системы две функции от двух переменных, которые задавались следующим образом:

$F_1(x_1, x_2)$, $n = 10$, $D = 2$:

$f_1, f_2, \ldots, f_{10}$: сферические функции

$\sigma_1, \sigma_2, \ldots, \sigma_{10} = [1, 1, \ldots, 1]$;

$\lambda_1, \lambda_2, \ldots, \lambda_{10}$ - равномерно распределенные в интервале [-5,5] случайные величины;

$$M = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix};$$

$F_2(x_1, x_2)$:

$f_1, f_2, \ldots, f_5$: сферические функции (5.17);

$f_6, f_7, \ldots, f_{10}$: функции Экли (5.21);

остальные параметры аналогичны.

Графики функций $F_1(x_1, x_2)$ и $F_2(x_1, x_2)$ приведены на рис.5.10, а на рис.5.11 показаны восстановленные с помощью ЭРБС поверхности. Обе модели были построены с помощью независимых ЭРБС. Начальная популяция состояла из 128 особей, а максимально допустимое количество нейронов было равно 30. На рис. 5.12 приведено изменение количества нейронов скрытого слоя для обеих сетей. На рис. 5.13 показано изменение значения фитнесс-функций сетей победителей при построении каждой модели. Фронт Парето, построенный с

использованием полученных моделей, приведен на рис 5.14.

Эксперимент 2. Для исследования влияния помехи на процесс многокритериального обучения использовалась функция от двух переменных, которая задавалась следующим образом:

$F(x_1, x_2)$, $n = 10$, $D = 2$:

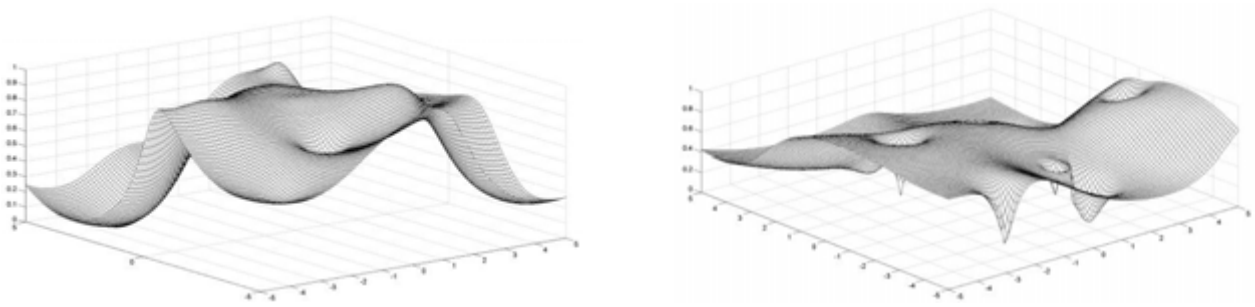
$f_1, f_2, ..., f_{10}$: сферические функции

$\sigma_1, \sigma_2, ..., \sigma_{10} = [1, 1, \ldots, 1]$;

$\lambda = [0.2 \quad 0.2 \quad 10 \quad 10 \quad 0.05 \quad 0.05 \quad 5/32 \quad 5/32 \quad 0.05 \quad 0.05]$;

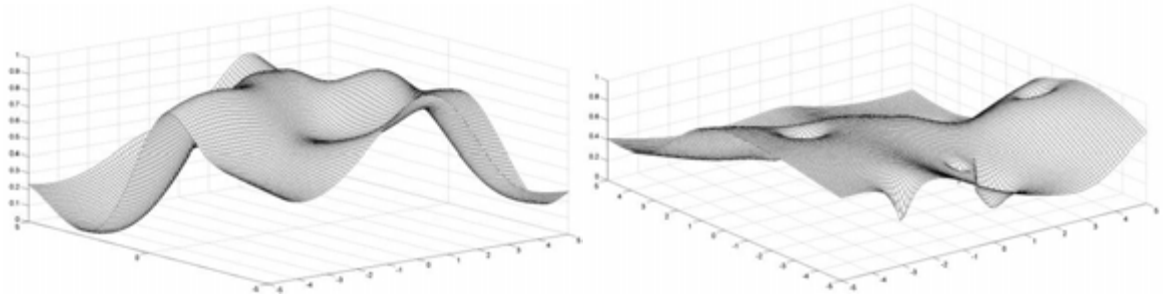
$M = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$.

График функции $F(x_1, x_2)$ приведен на рис. 5.15 -а). На рис. 5.15 -б) показана поверхность, зашумленная помехой ξ с нормальным распределением ($\sigma = 0.6$) при наличии «выбросов» также с нормальным распределением ($\sigma = 12.0$). Полученные при аппроксимации данной функции фронты Парето для двухслойного МП и РБС показаны на рис. 5.16-а), б) соответственно. Квадратиками обведены оптимальные сети, выбранные из фронта Парето с помощью информационного критерия Акаике. Оптимальный МП содержал 7 нейронов в первом скрытом слое и 5 во втором, а оптимальная РБС сеть состояла из 10 нейронов шаблонного слоя. На рис. 5.17 показана поверхность, восстановленная с помощью РБС. Все результаты аппроксимации заданной функции с помощью разных подходов к обучению при наличии помех измерений с различными законами распределения сведены в табл.5.1. Здесь приведены значения ошибок обучения для оптимальных МП и РБС. В скобках приведено число нейронов соответствующих сетей для каждого случая. Для МП первое число соответствует количеству нейронов в первом слое, а второе – во втором.



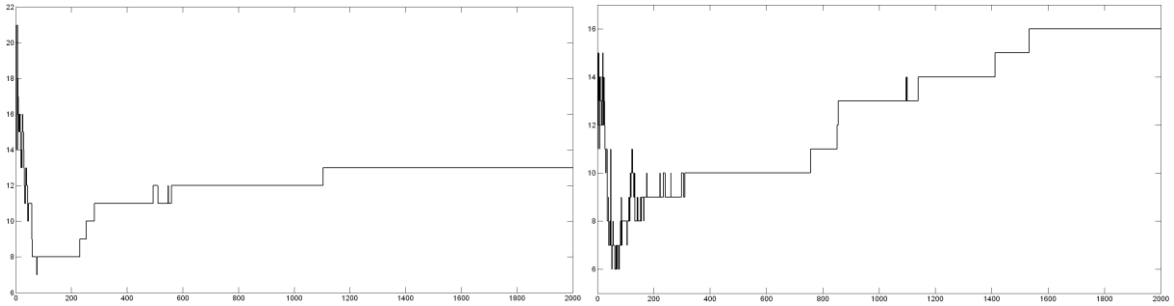
а) б)

Рисунок 5.10 – Графики функций $F_1(x_1, x_2)$ и $F_2(x_1, x_2)$



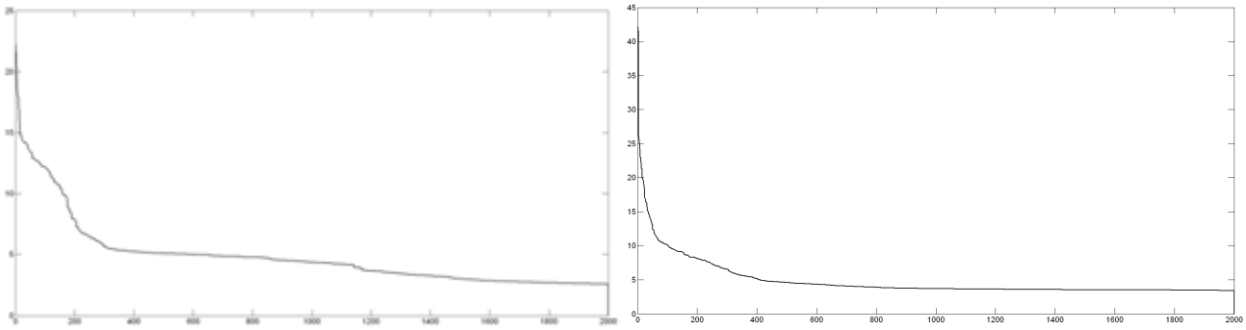
а) б)

Рисунок 5.11 – Восстановленные с помощью ЭРБС поверхности



а) б)

Рисунок 5.12 – Изменение количества нейронов скрытого слоя



а) б)

Рисунок 5.13 – Изменение значения фитнесс-функций сетей победителей

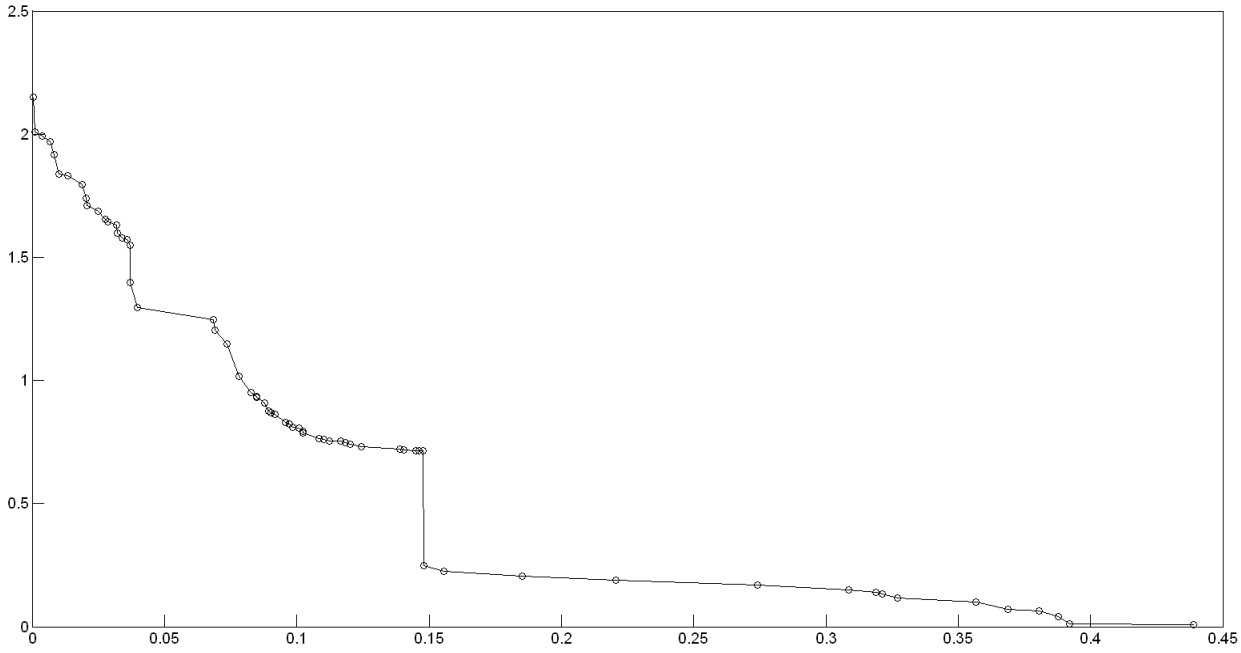
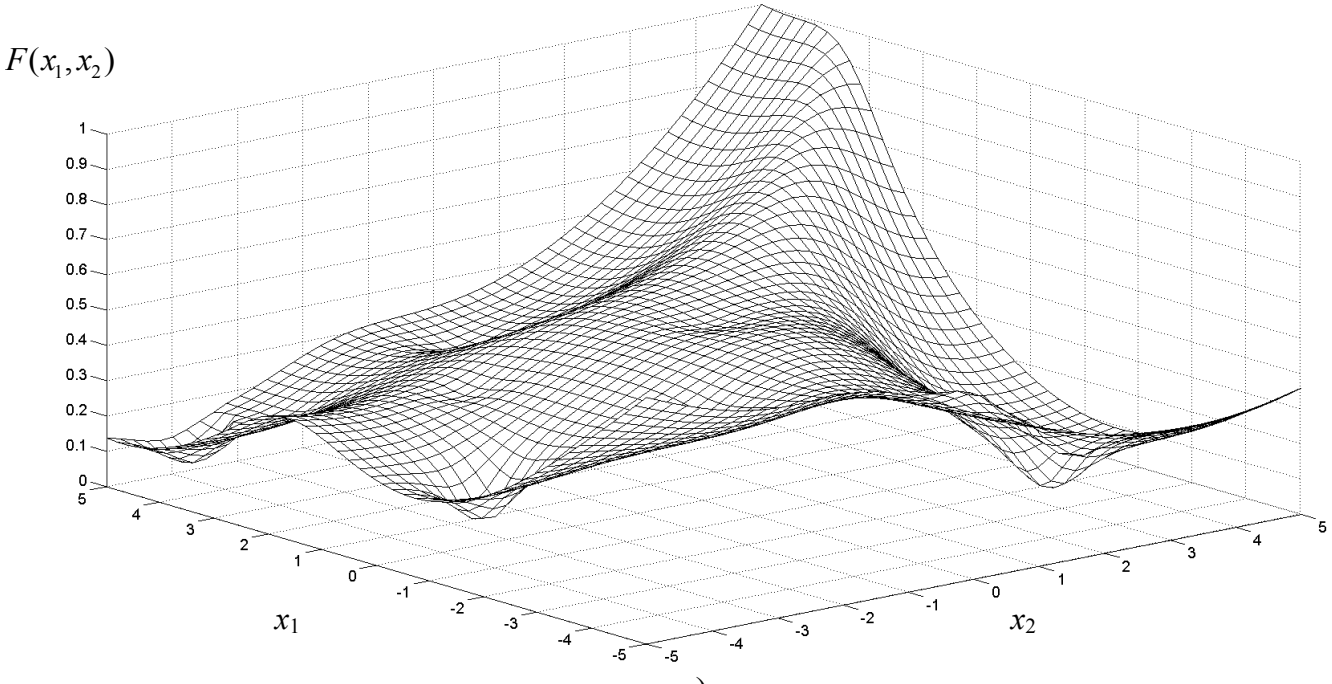
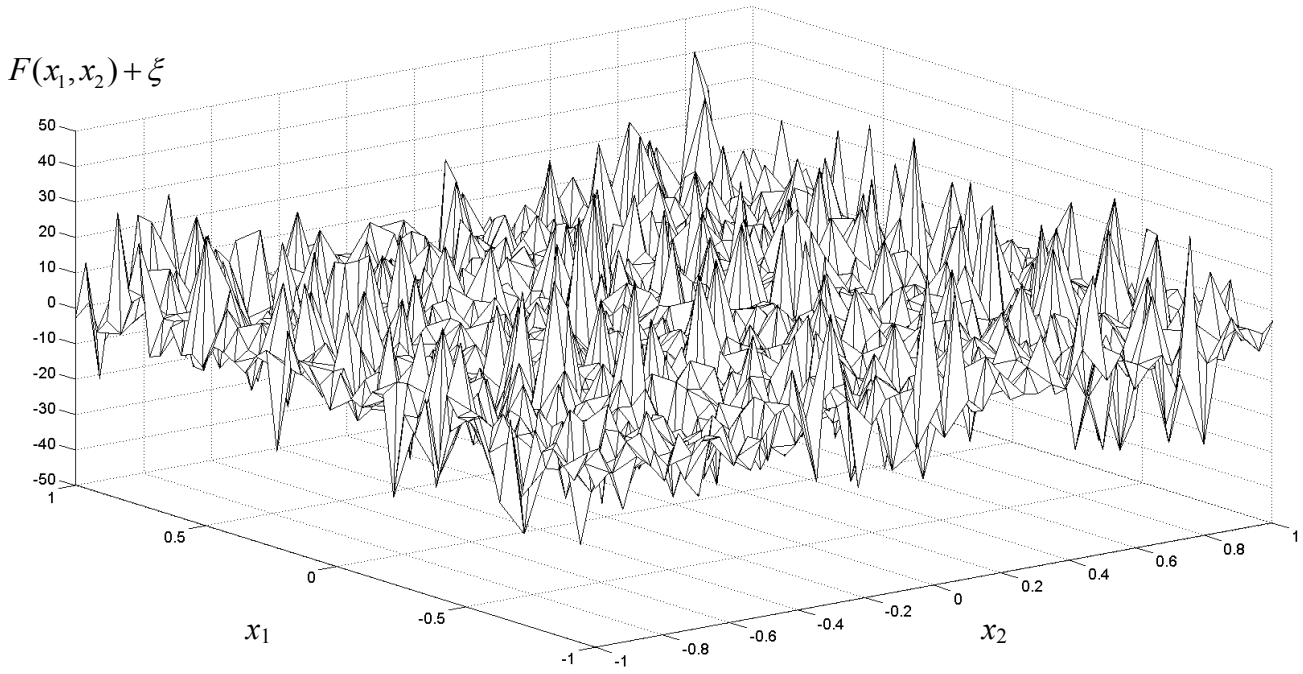


Рисунок 5.14 – Фронт Парето

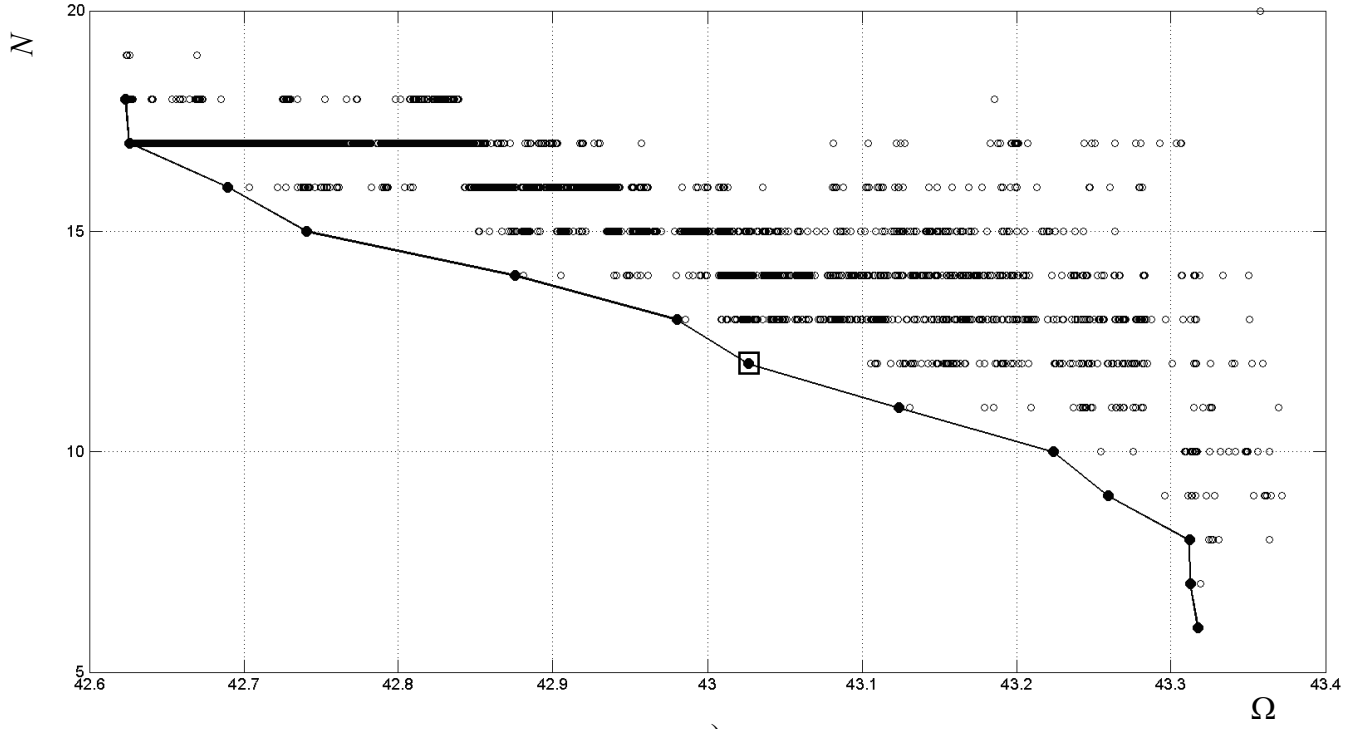


а)

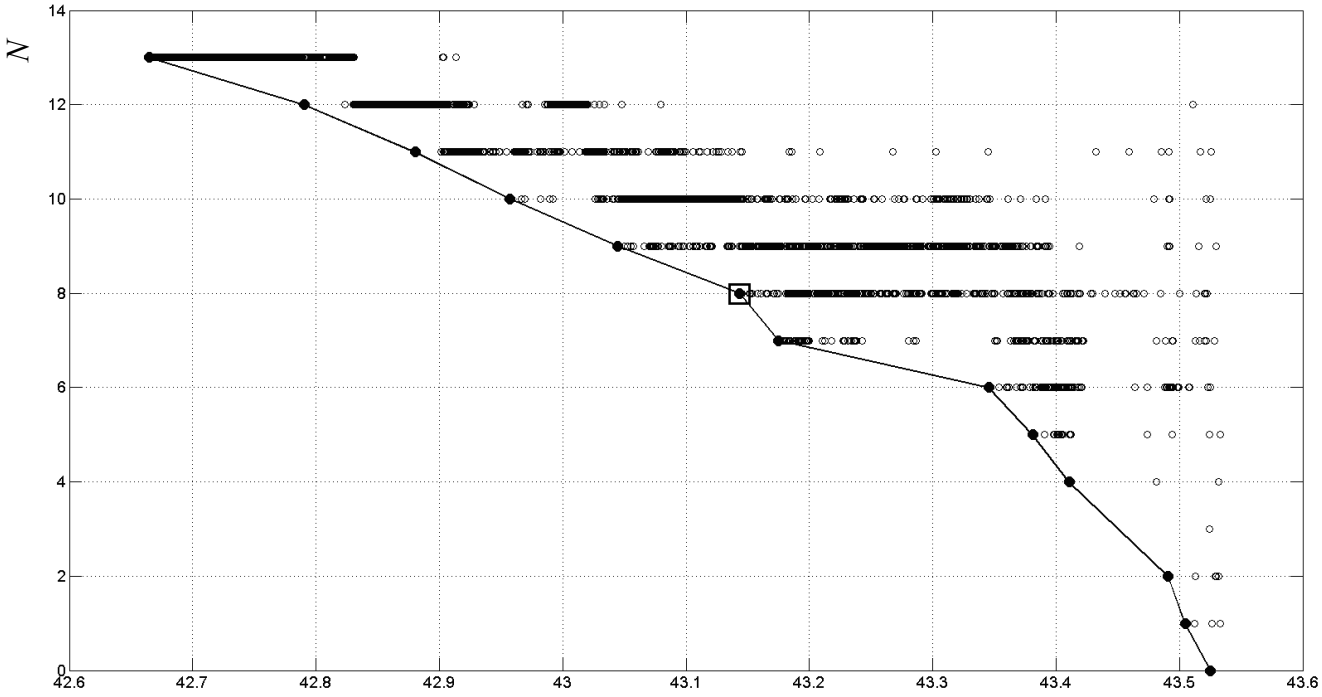


б)

Рисунок 5.15 – Графики функции $F(x_1,x_2)$



а)



б)

Рисунок 5.16 – Полученные при аппроксимации функции фронты Парето для двухслойного МП – а) и РБС – б)

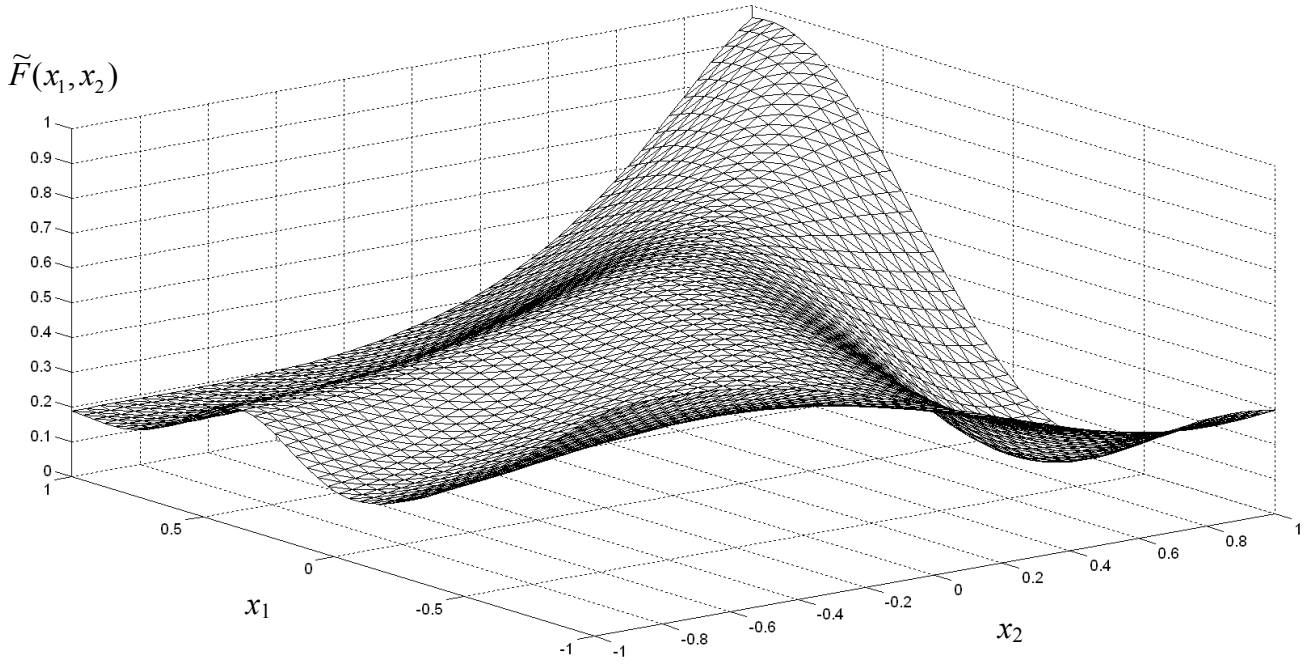


Рисунок 5.17 – Поверхность, восстановленная с помощью РБС

Таблица 5.1 – Результаты аппроксимации функции

Вид обучения	Тип сети	Ошибка обучения (количество нейронов)		
		Тип помехи		
		Без помехи	Нормальный	Лапласа
Одноцелевое обучение	МП	3.5989 (9-8)	69.5837 (10-9)	66.6294 (11-7)
	РБФ	5.2894 (18)	74.1397 (20)	68.2867 (18)
Скаляризованное обучение	МП	2.8479 (8-6)	50.1750 (9-9)	52.0112 (10-8)
	РБФ	4.9475 (15)	48.9896 (17)	49.4150 (18)
Обучение по Парето	МП	2.3454 (6-4)	43.02 (7-5)	35.6755 (7-7)
	РБФ	3.1502 (12)	43.1405 (8)	46.8705 (11)

5.7.2 Моделирование работы коэволюционирующих ИНС.

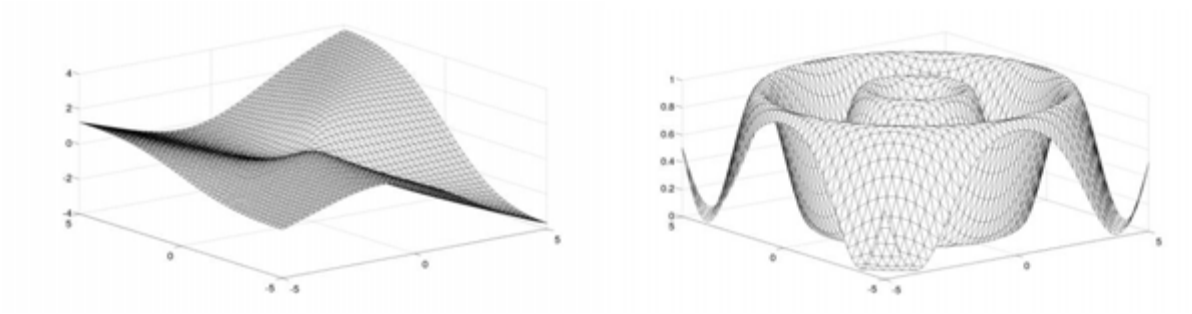
Эксперимент 1. Решалась задача аппроксимации функций двух переменных

$$f[x_1, x_2] = \frac{x_1 x_2 [x_1 + 2.5]}{1 + x_1^2 + x_2^2}, \tag{5.22}$$

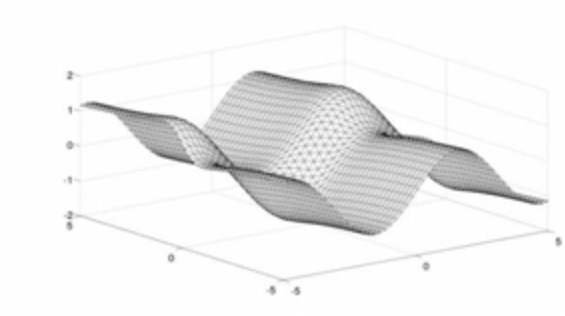
$$f[x_1, x_2] = 0.5 + \frac{\sin^2 \sqrt{x_1^2 + x_2^2} - 0.5}{1 + 0.001(x_1^2 + x_2^2)}, \tag{5.23}$$

$$f[x_1, x_2] = \sin(x_1) + \frac{x_2}{1 + x_2^2}, x_1, x_2 \in [-5, 5], \tag{5.24}$$

с помощью различных конкурирующих популяций ЭРБС, использующих только гауссовскую функцию активации (популяция 1), только обобщенную гауссовскую функцию (популяция 2), и ЭРБС, использующую одновременно обе функции активации (популяция 3). Поверхности, описываемые уравнениями (5.22)-(5.24) показаны на рис.5.18. Результаты аппроксимации функций (5.22)-(5.24) приведены в табл. 5.2.



а) б)



в)

Рисунок 5.18 - Поверхности, описываемые уравнениями (5.22)-(5.24)

Таблица 5.2 – Результаты аппроксимации функций (5.22)-(5.24)

Популяция	Функция	Длина хромосомы (число интронов)	Ошибка аппроксимации
1	(5.22)	103 (0)	0.13675
	(5.23)	68 (0)	0.40705
	(5.24)	82 (0)	0.03357
2	(5.22)	113 (0)	0.19032
	(5.23)	113 (0)	0.40863
	(5.24)	50 (0)	0.08114
3	(5.22)	95 (12)	0.11165
	(5.23)	123 (8)	0.27205
	(5.24)	85 (12)	0.01234

Как видно из результатов моделирования, популяция 3, одновременно использующая в шаблонном слое сети ЭРБС различные базисные функции, является победителем. В целом такой подход позволяет значительно повысить точность аппроксимирования сложных функции, т.к. предоставляет разработчику выбор из множества возможных решений.

Эксперимент 2. Решалась задача аппроксимации нелинейной функции «Сомбреро» (5.23) с использованием стратегии кооперативно-конкурирующей коэволюции. Было создано 20 популяций, каждая из которых содержала по 10 особей (РБС), максимально допустимое количество нейронов в каждой сети было ограничено 25. Все популяции решали задачу аппроксимации в полном объеме без взаимодействия между собой, конкурируя лишь за вычислительные ресурсы и ресурсы памяти. Первые 100 эпох популяции эволюционировали без внешнего вмешательства, затем каждые 25 эпох происходило удаление худшей популяции с наибольшим значением фитнес-функции. Освободившиеся ресурсы перераспределялись между оставшимися популяциями (увеличивалась их численность и количество мутировавших потомков). Кроме того, лучшая особь из удаляемой популяции клонировалась во все оставшиеся популяции для поддержания их разнообразия и устранения риска потери полезной информации, полученной в результате эволюции удаляемой популяции. Начиная с 575-ой эпохи эволюционировала лишь одна популяция, состоящая из 55 особей, которая получила полный доступ ко всем имеющимся у алгоритма ресурсам. После достижения критерия останова (1000 эпох обучения) из хромосомы (генотип) лучшей особи оставшейся популяции была образована РБС (фенотип), которая и получила право решать задачу аппроксимации функции (5.23). Данная сеть содержала 19 нейронов (7 с гауссовской БФ и 12 с БФ «мексиканская шляпа»). На рис.5.19-а) показана поверхность, описываемая уравнением (5.23), а на рис.5.19-б) – поверхность, восстановленная с помощью

ЭРБС. На рис.5.20 приведены графики изменения значения фитнесс функций наилучших особей каждой из популяций.

Эксперимент 3. В данном эксперименте задача аппроксимации функции (5.23) решалась с помощью иной кооперативно-конкурентной стратегии, заключающейся в параллельной эволюции множества популяций, решающих различные под-задачи идентификации. Для этого пространство входных сигналов разбивалось на кластеры с помощью дополнительной популяции, отвечающей за настройку центров кластеров с помощью ЭА. Было задано 25 кластеров, начальные значения центров которых задавались двумя способами – случайным распределением и равномерным. Задачу аппроксимации в каждом кластере решала отдельная популяция РБС. После 500 эпох обучения конечное решение было сформировано с помощью особей-победителей в каждом кластере. Результаты аппроксимации функции (5.23) с различным подходом к выбору начальных значений центров кластеров представлены на рис.5.21-5.22. На рис.5.21 а), б) показаны графики изменения фитнес-функций сетей-победителей в каждой популяции для случаев случайного и равномерного задания начальных значений центров кластеров соответственно. На рис.5.22 а), б) приведены восстановленные поверхности, а на рис.5.22 в), г) – результирующее разбиение пространства входных сигналов на кластеры. Как видно из результатов экспериментов, для данного случая начальное равномерное распределение пространства входных сигналов на кластеры дает лучшие результаты, однако в общем случае рекомендуется все же использовать случайное разбиение, так как зачастую вид аппроксимируемой функции является неизвестным.

а) б)

Рисунок 5.19 – Поверхность, описываемая уравнением (5.23) –а); поверхность, восстановленная с помощью ЭРБС – б).

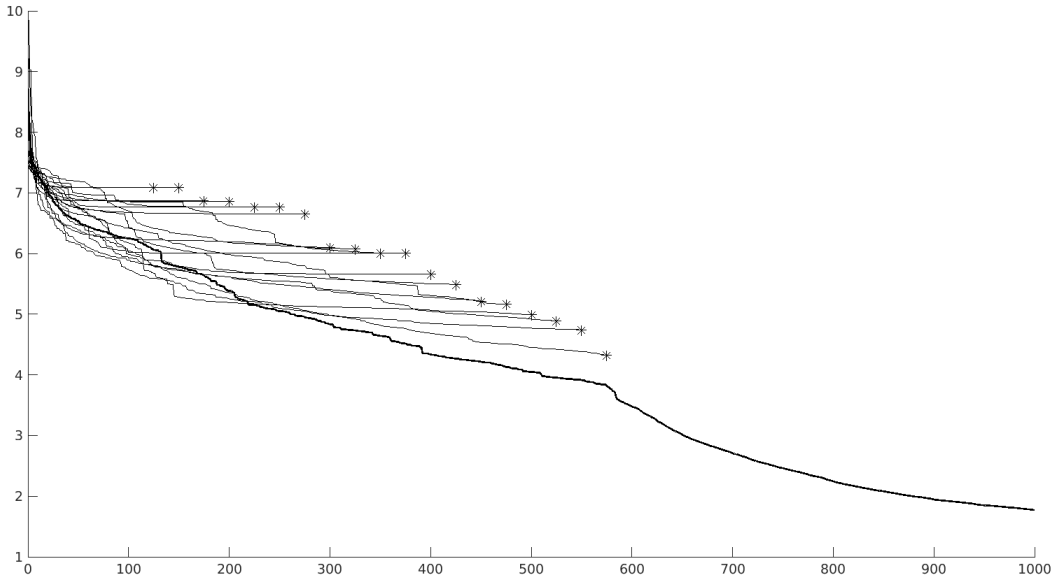
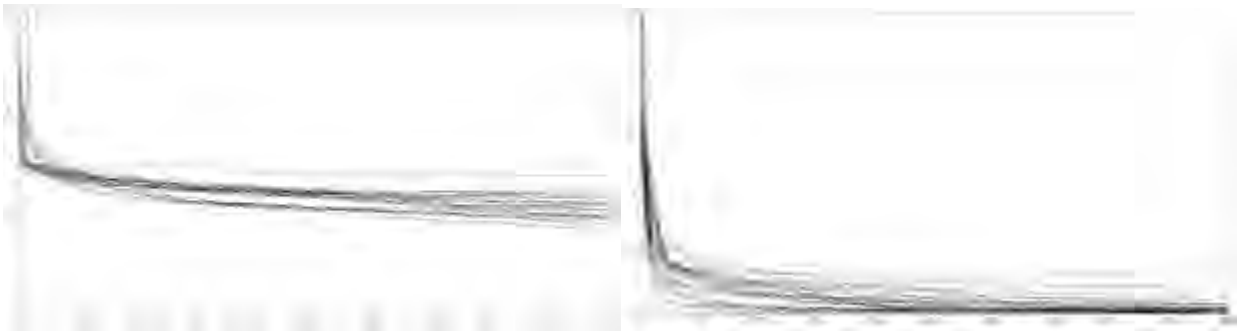


Рисунок 5.20 – Графики изменения значения фитнесс функций



а) б)

Рисунок 5.21 – Графики изменения фитнес-функций сетей-победителей

Эксперимент 4. В данном эксперименте проводилась аппроксимация функции Катковника, имеющей вид

$$F(x,y) = 0.5(x^2 + y^2)\left(1.6 + 0.8\cos(1.5x)\cos(\pi y) + 0.8\cos(\sqrt{5}x)\cos(3.5y)\right), \quad (5.25)$$

где $x,y \in [-2.5,2.5]$.

Все параметры эксперимента идентичны эксперименту 3. Отличие заключается в том, что в качестве ИНС вместо РБС был использован МП с двумя скрытыми слоями, содержащими не более 20 нейронов с сигмоидальными функциями активации каждый. Результаты моделирования приведены на рис.5.23.

Эксперимент 5. Рассматривалась задача идентификации нелинейных многомерных объектов: четырехмерного $F(x_1, x_2, x_3, x_4)$

$$F(x_1, x_2, x_3, x_4) = x_1 + \sin(\pi x_1)\cdot\cos(\pi x_2)\cdot\sin(\pi x_3)\cdot\left(\sin(\pi x_4)^2 - 1\right) \quad (5.26)$$

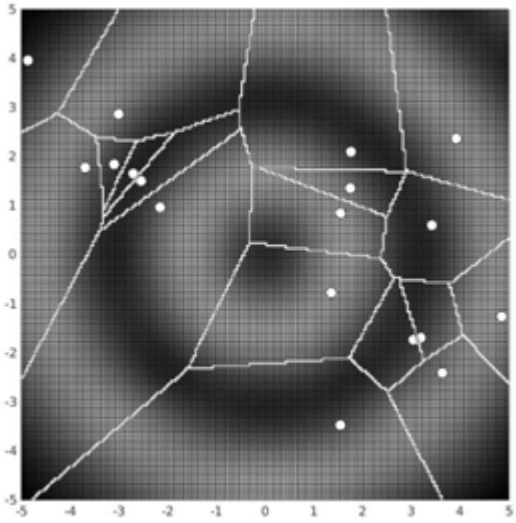
и шестимерного $F(x_1, x_2, x_3, x_4, x_5, x_6)$

$$\begin{aligned} F(x_1, x_2, x_3, x_4) &= sin(\pi x_1)\cdot sin(\pi x_2)\cdot sin(\pi x_3) + \\ &+ sin(\pi x_4)\cdot sin(\pi x_5)\cdot sin(\pi x_6) + \\ &+ exp\left(-\left(x_1^2 + x_2^2\right)\right) + exp\left(-\left(x_3^2 + 6x_3x_4 + 10x_4^2\right)\right) \end{aligned} \quad (5.27)$$

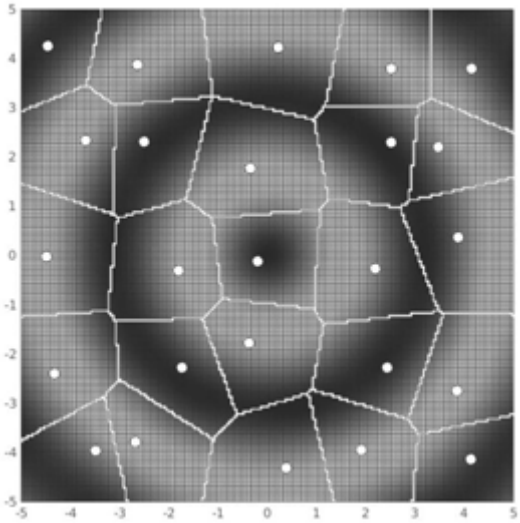
Сечение функции (5.26) при $x_2 = x_3 = 0.25$ показано на рис. 5.24.

а)

б)

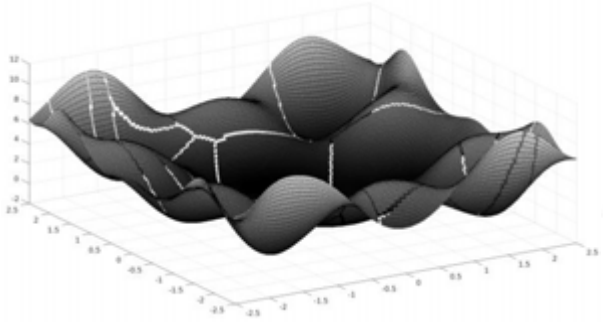


в)

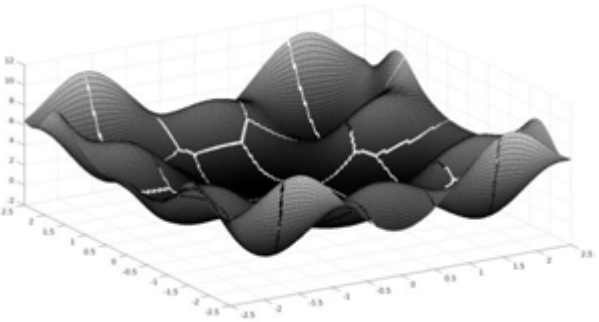


г)

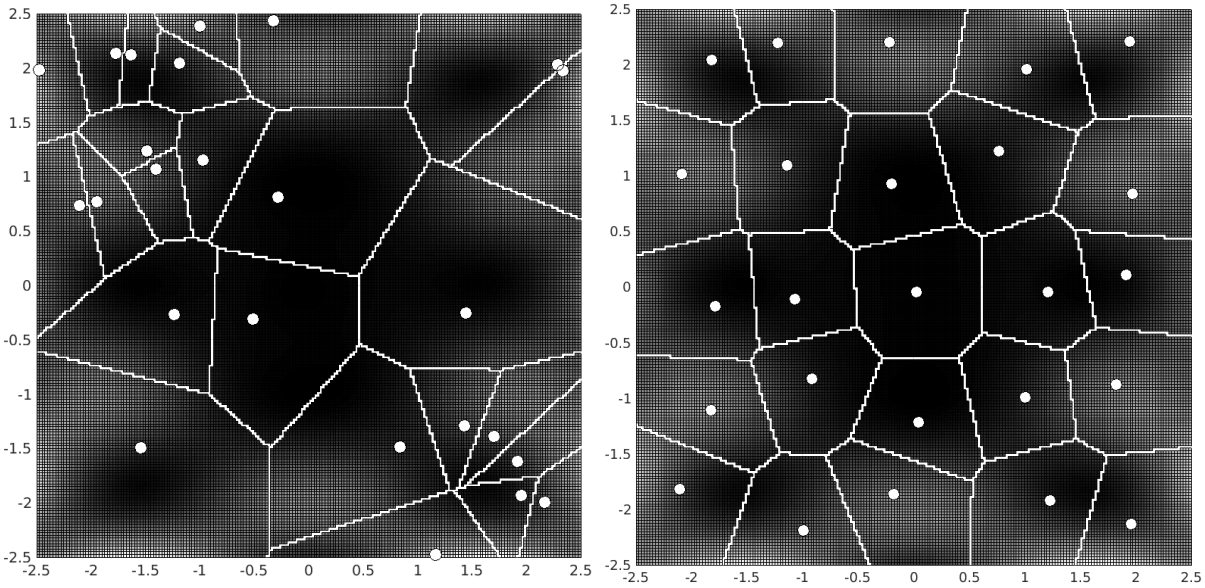
Рисунок 5.22 – Восстановленные поверхности и результирующее разбиение пространства входных сигналов на кластеры



а)



б)



в) г)

Рисунок 5.23 – Восстановленные поверхности и результирующее разбиение пространства входных сигналов на кластеры

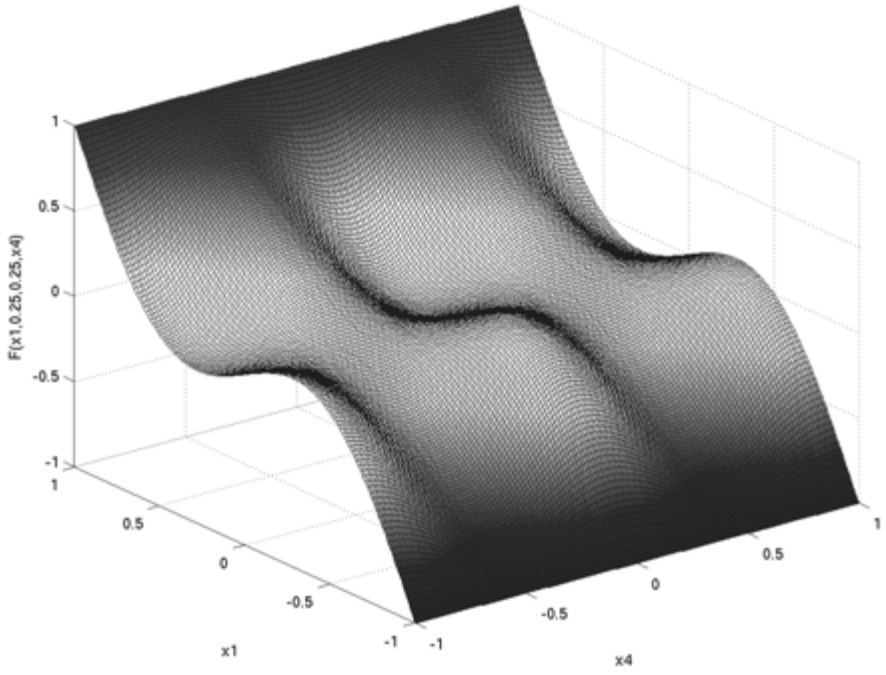
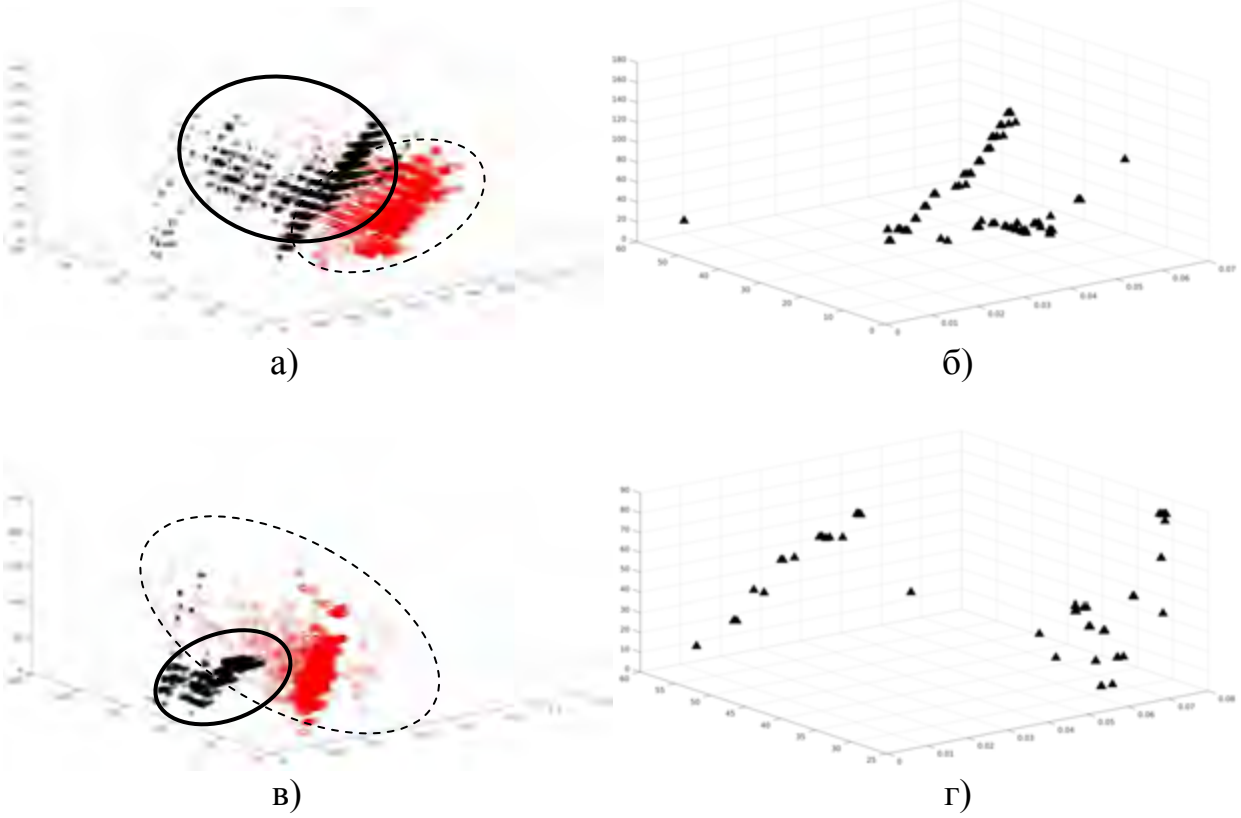


Рисунок 5.24 – Вид исходной функции $F(x_1, 0.25, 0.25, x_4)$

Данная задача решалась с использованием коэволюционного подхода на основе конкуренции. За получение права предоставления наилучшего решения задачи идентификации соревновались две популяции эволюционирующих ИНС, состоящих из РБС и МП. Обе популяции содержали по 150 особей и после каждой эпохи обучения по результатам вычисления значений фитнес-функции (2.10), измерения времени симуляции и определения количества настраиваемых параметров каждой особи, строился результирующий фронт Парето для обеих популяций. Все возможные решения после 1000 эпох обучения для функции (5.26) представлены на рис.5.25 а), а результирующий фронт Парето показан на рис.5.25 б). Аналогично для функции (5.27) результаты моделирования приведены на рис.5.25 в, г). Решения, полученные с помощью МП, на рисунках обозначены звездочками, а место их сосредоточения обведено сплошной жирной линией. Решения же, полученные с помощью РБС показаны на рисунках кружками, а область их скопления обведена пунктирной линией. Фронт Парето строился общий для обеих популяций и из него выбиралось лучшее решение в соответствии с критерием Акайке. Лучшее решение могло принадлежать как популяции, состоящей из МП, так и из РБС.

Как видно из результатов моделирования, для 4-ех мерного объекта популяция РБС дает лучшие результаты по всем минимизируемым критериям, а для 6-ти мерного объекта популяция МП дает лучшие решения по сложности модели и времени симуляции, несколько проигрывая, однако, по точности идентификации.



а) б)

в) г)

Рисунок 5.25 – Все возможные решения и результирующий фронт Парето для функций (5.26)-(5.27)

Выводы по разделу 5

1. Использование ИНС требует решения задач структурной и параметрической оптимизации, соответствующие выбору оптимальной топологии сети и ее обучению. В отличие от задачи определения структуры, являющейся дискретной оптимизационной (комбинаторной), поиск оптимальных параметров осуществляется в непрерывном пространстве с помощью классических методов оптимизации. Для обучения сетей прямого распространения с учителем применяются, как правило, алгоритмы, оптимизирующие некоторую целевую функцию. Однако традиционно во внимание принимается лишь одна цель в качестве стоимостной функции либо несколько целей объединяются в одну скалярную функцию. Однако

использованию скаляризованных целевых функций для многоцелевой оптимизации присущи два основных недостатка, во-первых, нетривиальной является задача определения оптимального параметра λ, во-вторых, при этом может быть получено только одно решение, которое в ряде случаев может быть неэффективным.

2. Более мощным по сравнению с обучением на основе скалярной стоимостной функции является многоцелевое обучение на основе подхода Парето, когда минимизируется векторная целевая функция, что обеспечивает получение некоторого количества Парето-оптимальных решений. Так как получаемая нейросетевая модель, с одной стороны, должна быть достаточно простой и удобной для использования ее в прикладных задачах, а с другой – наиболее полно отражать свойства исследуемого объекта, ее качество определяется некоторым набором критериев, т.е. задача построения нейромодели является многокритериальной.

3. Рассмотрена математическая формулировка задачи многокритериальной оптимизации по Парето и предложен обобщенный эволюционный алгоритм ее решения, включающий ГА и алгоритм, основанный на искусственных иммунных системах. Так как наиболее важным шагом процедуры оптимизации ИНС является выбор оптимального решения из фронта Парето, наиболее целесообразным для этого представляется использование информационного критерия, содержащего информацию об ошибке обучения, сложности и параметрах модели.

4. Рассмотрены вопросы синтеза коэволюционирующих ИНС, в которых реализуется адаптация в изменяющейся внешней среде и используются значения фитнес-функций относительно некоторых оппонентов, а не одной особи. Анализ двух основных форм коэволюционирующих систем конкуренции и кооперации позволил определить их преимущества и недостатки и реализовать многокритериальные методы обучения ИНС на основе

кооперативной и конкурентной коэволюции. При этом показано, что для всех типов сетей во всех субпопуляциях используются общие эволюционные процедуры (селекция, скрещивание, мутация и т.д.), а различия заключаются лишь в способе кодирования структуры и параметров ИНС. Рассмотрена возможность реализации многокритериального обучения ИНС с использованием алгоритмов кластеризации, в частности, алгоритм OLVQ.

5. С целью проведения имитационного моделирования алгоритмов многокритериальной оптимизации была использована специальная схема построения тестовых функций из некоторых базисных функций (сферической, функций Растригина, Вейерштрасса и т.д.), позволяющая строить сложные тестовые функции с хаотически расположенными глобальными оптимумами и несколькими случайно расположенными локальными оптимумами.

6. Как свидетельствуют результаты приведенного исследования, применение коэволюционного подхода в сетях прямого распространения является достаточно эффективным инструментом, позволяющим, во-первых, решать более сложные задачи, во-вторых, получать более простые структуры ИНС за счет уменьшения количества нейронов, в-третьих, значительно повысить робастность получаемой при этом сети по сравнению с решениями, основанными на одиночной ИНС. При использовании же векторного квантования для разбиения основной задачи на подзадачи открытым остается вопрос о выборе количества кластеров и начальных значений их центров. Наличие информации об аппроксимируемой функции позволяет несколько упростить решение этой задачи.

РАЗДЕЛ 6

ЭКСПЕРЕМЕНТАЛЬНЫЕ ИССЛЕДОВАНИЯ И РЕШЕНИЕ ПРАКТИЧЕСКИХ ЗАДАЧ

Имитационное моделирование является необходимым этапом исследования эффективности как выбранной архитектуры ИНС, так и применяемого алгоритма обучения сети.

В настоящее время наиболее массовым направлением нейрокомпьютинга является моделирование нейронных сетей на персональных компьютерах. Успехи в моделировании позволяют создавать обучающие программы, дающие пользователю полное представление об ИНС и их возможностях.

Однако независимо от используемого пакета результатом экспериментального исследования является определение областей наиболее эффективного применения того или иного типа ИНС и разработка рекомендаций по их практическому использованию.

В настоящее время популярны и широко используются следующие программные нейросимуляторы:

– ***MATLAB Neural Network Toolbox*** – предоставляет большие возможности по исследованию свойств алгоритмов обучения ИНС. Программный пакет представляет собой пакет программ, ориентированный на решение широкого спектра задач с использованием нейросетевых алгоритмов. В нем предусмотрена реализация большого количества разновидностей нейронных сетей, а также возможность создания пользовательских сетей практически любой конфигурации. Существенным недостатком данного пакета является его высокая стоимость.

– ***Scilab 5.5.2*** – пакет прикладных математических программ, предоставляющий открытое окружение для инженерных (технических) и научных расчётов. Это самая полная общедоступная альтернатива MATLAB. С 1994 года распространяется вместе с исходным кодом через сеть Интернет. В 2003 году для поддержки Scilab был создан консорциум Scilab Consortium. Сейчас в него входят 25 участников, в том числе Mandriva, INRIA и ENPC (Франция). Scilab содержит сотни математических функций, и есть возможность добавления новых,

написанных на различных языках (C, C++, Fortran и т. д.). Также имеются разнообразные структуры данных (списки, полиномы, рациональные функции, линейные системы), интерпретатор и язык высокого уровня. Scilab имеет схожий с MATLAB язык программирования. В состав пакета входит утилита, позволяющая конвертировать документы Matlab в Scilab. Программа доступна для различных операционных систем, включая Linux и Microsoft Windows. Возможности Scilab могут быть расширены внешними программами и модулями, написанными на разных языках программирования. Программа имеет открытый исходный код, что позволяет как свободное коммерческое использование и распространение неизменённых версий, так и некоммерческое распространение измененных версий, которые должны включать в себя исходный код. Для коммерческого распространения измёненных версий необходимо согласование с INRIA. Начиная с версии 5.0 программа распространяется под совместимой с GNU GPL 2 лицензией CeCILL.

– ***NeurophStudio 2.92*** – платформа для работы с объектно-ориентированными нейронными сетями. Может быть использована для создания и обучения нейронных сетей в Java программах. NeurophStudio предоставляет Java библиотеку, а также инструмент с графическим интерфейсом easyNeurons для интерактивной работы с нейронными сетями. NeurophStudio позволяет эмулировать работу достаточно большого количества алгоритмов нейронных сетей, является достаточно универсальным и многофункциональным среди свободно распространяемых программных нейросимуляторов. Предоставляется под лицензией Apache 2.04.

– ***Torch 7*** - библиотека для научных вычислений с широкой поддержкой алгоритмов машинного обучения. Разрабатывается Idiap Research Institute, New York University и NEC Laboratories America, начиная с 2000г., распространяется под лицензией BSD. Библиотека реализована на языке Lua с использованием C и CUDA. Чрезвычайно быстрый язык сценариев LuaJIT, созданный на основе языке программирования Lua, в совокупности с технологиями SSE, OpenMP, CUDA позволяют Torch показывать неплохую скорость по сравнению с другими библиотеками. На данный момент поддерживаются операционные системы Linux, FreeBSD, Mac OS X. Основные модули также работают и на Windows. Библиотека

состоит из набора модулей, каждый из которых отвечает за различные стадии работы с нейросетями. Установка дополнительных модулей позволяет расширить функционал библиотеки. Torch позволяет создавать сложные нейросети с помощью механизма контейнеров.

– ***Theano*** – это расширение языка Python, позволяющее эффективно вычислять математические выражения, содержащие многомерные массивы. Библиотека получила свое название в честь имени жены древнегреческого философа и математика Пифагора – Феано (или Теано). Theano разработана в лаборатории LISA для поддержки быстрой разработки алгоритмов машинного обучения. Библиотека реализована на языке Python, поддерживается на операционных системах Windows, Linux и Mac OS. В состав Theano входит компилятор, который переводит математические выражения, написанные на языке Python в эффективный код на C или CUDA. Theano предоставляет базовый набор инструментов для конфигурации нейросетей и их обучения. Возможна реализация многослойных полностью связанных сетей (Multi-Layer Perceptron), сверточных нейросетей (CNN), рекуррентных нейронных сетей (Recurrent Neural Networks, RNN), автокодировщиков и ограниченных машин Больцмана. Также предусмотрены различные функции активации, в частности, сигмоидальная, softmax-функция, кросс-энтропия. В ходе обучения используется пакетный градиентный спуск (Batch SGD).

– ***Caffe*** – Разработка ведется с сентября 2013 г. Начало разработки положил Yangqing Jia во время его обучения в калифорнийском университете в Беркли. С указанного момента Caffe активно поддерживается Центром Зрения и Обучения Беркли (The Berkeley Vision and Learning Center, BVLC) и сообществом разработчиков на GitHub. Библиотека распространяется под лицензией BSD 2-Clause. Caffe реализована с использованием языка программирования C++, имеются интерфейсы для Python и MATLAB. Официально поддерживаемые операционные системы – Linux и OS X, также имеется неофициальная реализация на Windows. Caffe использует библиотеку BLAS (ATLAS, Intel MKL, OpenBLAS) для векторных и матричных вычислений. Для ускорения вычислений Caffe может быть запущена на GPU с использованием базовых возможностей технологии CUDA или библиотеки примитивов глубокого обучения cuDNN. Разработчики Caffe

поддерживают возможности создания, обучения и тестирования полностью связанных и сверточных нейросетей.

В данной работе эволюционирующие нейронные сети прямого распространения были реализованы как расширения пакета *NeurophStudio* и все исследования проводились в среде *NeurophStudio*.

6.1 Решение задачи идентификации

Рассмотрим задачу идентификации нелинейного динамического объекта, представленного NARMAX моделью [234]

$$\tilde{y}(k) = f[y(k\text{-}1), \ldots, y(k\text{-}m), u(k\text{-}1), \ldots, u(k\text{-}n), k] + \xi(k), \qquad (6.1)$$

где $\tilde{y}(i)$, $u(i)$ – выходной и входной сигналы объекта в момент времени i соответственно; m, n – порядки запаздывания по выходному и входному каналам соответственно; $f(\bullet)$ – неизвестная нелинейная функция; ξ – помеха измерения выходного сигнала.

Обозначим вектор обобщенного входного сигнала размерности (n+m)x1 так

$$x(k)=[y(k\text{-}1), y(k\text{-}2), \ldots, y(k\text{-}m), u(k\text{-}1), u(k\text{-}2), \ldots, u(k\text{-}n)].$$

Тогда уравнение (6.1) может быть записано следующим образом:

$$\tilde{y}(k) = f[x(k), k] + \xi(k).$$

Задача идентификации заключается в оценивании функции $f(\bullet)$ по измерениям входных $u(k)$ и выходных $y(k)$ переменных.

Представление объекта искусственной нейронной сетью, позволяет свести задачу идентификации к обучению сети, заключающемуся в

настройке ее весовых параметров. При этом в качестве критерия обучения (минимизируемого функционала) обычно выбирается квадратичный функционал ошибки

$$J = e^2(k) = M\left\{\left[\tilde{y}(k) - \hat{y}(k)\right]^2\right\},$$

минимизация которого осуществляется с помощью рассмотренных выше алгоритмов.

Эксперимент 1. Рассматривалась задача многокритериальной идентификации нелинейного стационарного объекта [224, 290-291]

$$y(k) = 0.725\beta(k)\sin\left(\frac{16u(k-1) + 8y(k-1)}{\beta(k)\left(3 + 4u^2(k-1) + 4y^2(k-1)\right)}\right) + 0.2u(k-1) + 0.2y(k-1), \tag{6.2}$$

где $u(k)$ - входной сигнал, представляющий собой стационарную случайную последовательность с равномерным законом распределения в интервале [-1, 1], генерируемую датчиком случайных чисел.

Поверхность, описываемая уравнением (6.2) представлена на рис. 6.1-а.

Для решения данной задачи использовалась популяция, состоящая из 150 особей (сетей). Максимально возможное количество нейронов в каждой сети было ограничено 15. Таким образом, каждая хромосома состояла из 106 генов. В качестве базисных были использованы функции №1 и №2 из табл.2.1.

Первые 100 шагов селекция особей для скрещивания производилась лишь на основе значения фитнес-функции, которая имела вид

$$f_i(x_j) = \frac{1}{K}\sum_{j=1}^{K}\left|y_j^*(x_j) - \hat{y}_j(x_j)\right|, \tag{6.3}$$

где K=2500 – размер выборки.

Такое предварительное обучение обусловлено тем, что на начальном этапе эволюции популяции не имеет смысла применение алгоритмов МО, так как ошибки всех моделей независимо от числа параметров являются достаточно большими.

После 100 шагов предварительного обучения для отбора особей помимо значения фитнес-функции дополнительно использовался критерий, характеризующий сложность модели. Использовалось два подхода: при первом в качестве такого параметра использовалось количество активных нейронов сети, а при втором – примененный в работе [234] критерий оценки сложности РБФ сети, имеющий вид

$$f_i = \sum_{j=1}^{N} \left| \frac{w_j}{\sigma_j} \right|. \qquad (6.4)$$

На этапе селекции для обоих случаев строился фронт Парето из которого затем выбирались особи для скрещивания и получения нового поколения.

Результирующие фронты Парето и восстановленные поверхности некоторыми входящими в него сетями после 1000 эпох обучения представлены на рис.6.2-6.3. Квадратиком обозначена особь, для которой критерий Акаике является минимальным.

Также проводилась многокритериальная идентификация сильнозашумленного нелинейного стационарного объекта (6.2) при наличии помехи $\xi(k)$, описываемой моделью

$$\xi(k) = (1-\varepsilon)q_1(k) + \varepsilon q_2(k),$$

где $\varepsilon = 0{,}1$; $q_1(k)$, $q_2(k)$ - нормально распределенные помехи с математическими ожиданиями $m_1 = m_2 = 0$ и дисперсиями $\sigma_1 = 0{,}6$; $\sigma_2 = 12$ соответственно.

Зашумленная таки образом поверхность (6.2) приведена на рис.6.1-б.

В данном эксперименте наличие помех обусловило необходимость модификации фитнес-функции в соответствии с (4.6), (4.15) следующим образом

$$f_i(x_j) = \frac{1}{M}\sum_{j=1}^{M}\frac{(y_j^*(x_j) - \hat{y}_j(x_j))^2}{1 + (y_j^*(x_j) - \hat{y}_j(x_j))^2}. \qquad (6.5)$$

$$f_i(x_j) = \frac{1}{M}\sum_{j=1}^{M} arctg\left|y_j^*(x_j) - \hat{y}_j(x_j)\right|; \qquad (6.6)$$

Полученный фронт Парето при использовании фитнес-функции (6.6) обозначен на рис.6.4 треугольниками, а для функции (6.5) – кружками. В первом случае, особь, выбранная из фронта Парето с помощью критерия Акаике и обведенная квадратом на рисунке, содержала 6 нейронов (2 с БФ вида (№1 табл.2.1) и 4 с БФ (№2 табл.2.1)), а во втором – 7 (3 с БФ вида (№1 табл.2.1) и 4 с БФ (№2 табл.2.1)).

Эксперимент 2. Решалась задача идентификации стационарного динамического объекта, описываемого уравнением (6.2), в котором $\beta(k)$ принимался равным 1.

Для решения данной задачи использовалась популяция, состоящая из 128 особей (сетей). Максимально возможное количество нейронов в каждой сети было ограничено 15. Таким образом, хромосома состояла из 106 генов. В качестве базисных были использованы функции, представленные в табл.2.1. После 1000 шагов обучения значение фитнесс-функции для сети-победителя было равным 1.5. Полученная сеть состояла из 12 нейронов (1 нейрон с БФ №1 (см. табл.2.1); 1 – №2; 3 – №3; 3 – №4; 3 – №5; 1 – №6). На

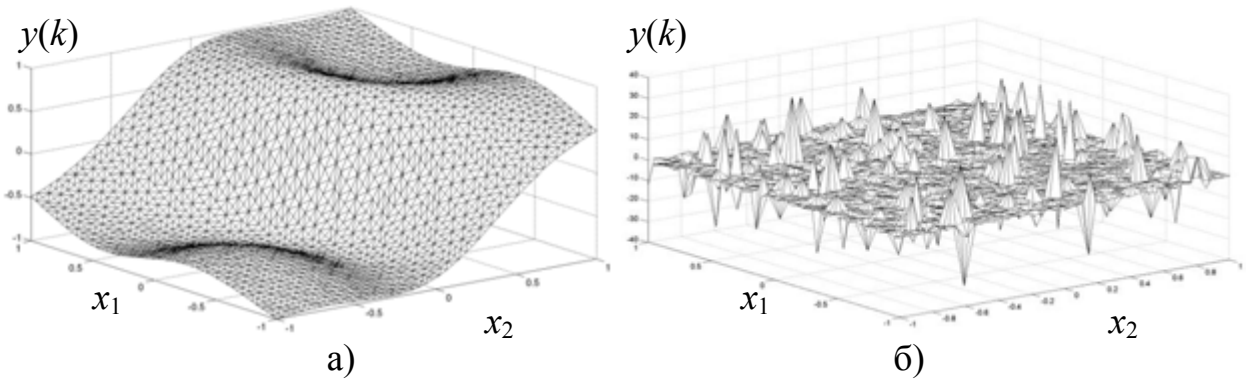


Рисунок 6.1 – Поверхность, описываемая уравнением (6.2)

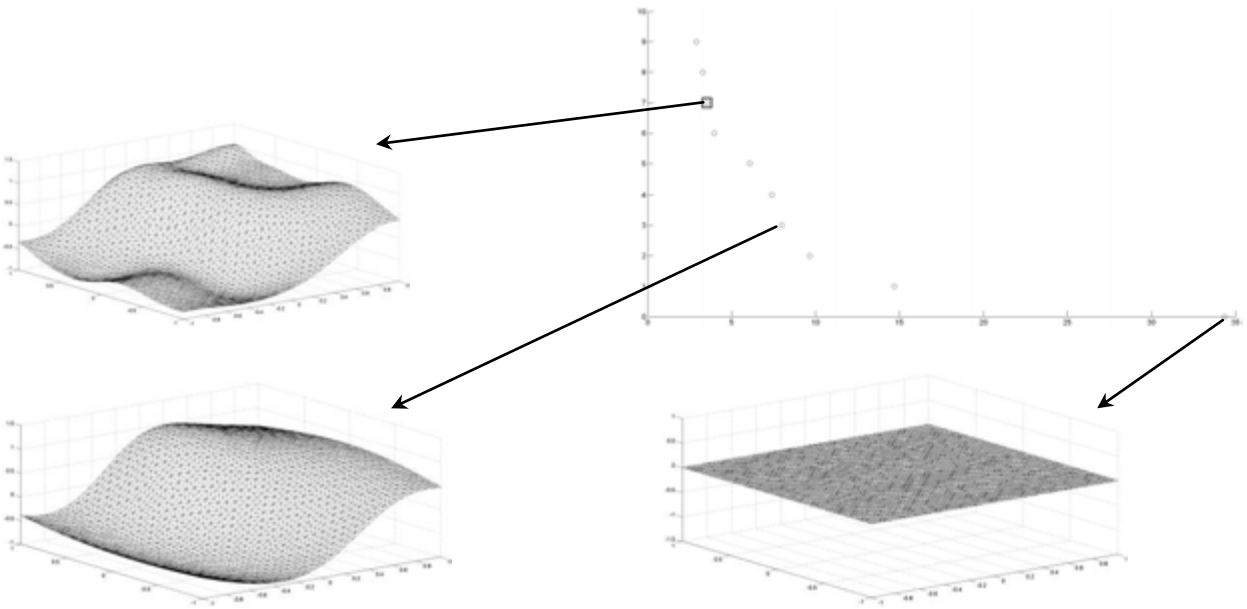


Рисунок 6.2 – Фронты Парето и восстановленные поверхности

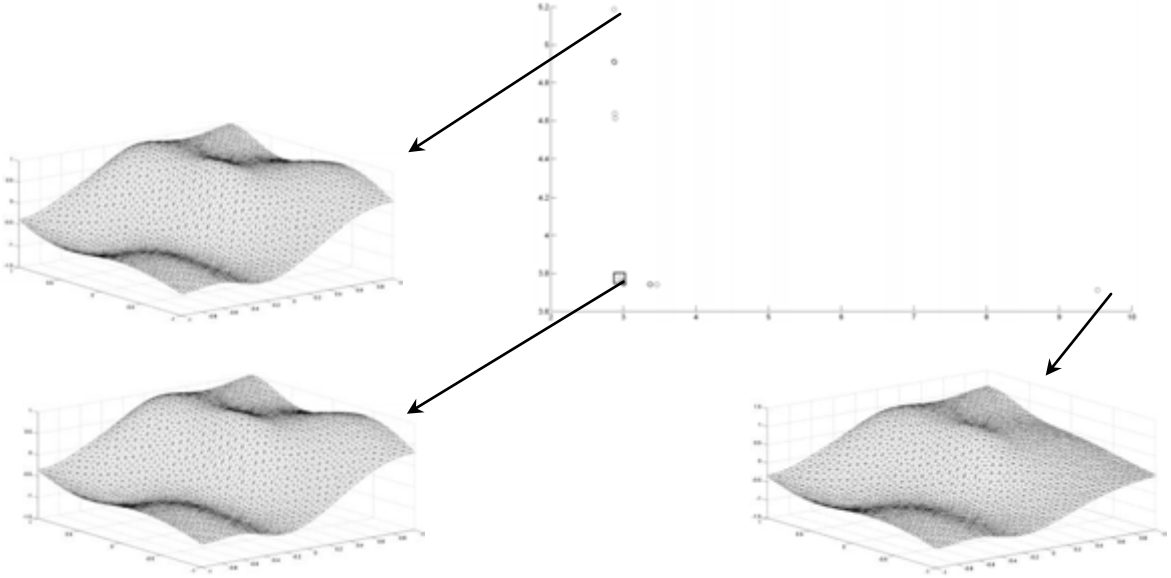


Рисунок 6.3 – Фронты Парето и восстановленные поверхности

рис. 6.5-а) показана восстановленная с помощью сети-победителя поверхность. Графики изменения количества активных нейронов сети-победителя и значения ее фитнесс-функции в каждом поколении приведены на рис. 6.5-б) и 6.5-в) соответственно.

Эксперимент 3. Проводилась идентификация нестационарного динамического объекта, описываемого уравнением (6.2) с

$$\beta(k) = \begin{cases} 1, & 1 \le k \le 500, \\ 0{,}6, & 501 \le k \le 1000. \end{cases}$$

После 500 эпох обучения сеть состояла из 13 нейронов (2 нейрона с БФ №1 (см. табл.2.1); 4 – №2; 3 – №3; 1 – №4; 3 –№5).

Так как параметры объекта на 501-ом шаге изменились, генетическому алгоритму обучения необходимо было перенастроить как структуру сети, так и ее параметры. В связи с этим, на 1000 шаге оптимальной была уже другая сеть с 12 нейронами (2 - №1; 2 – №2; 1 – №3; 1 – №4; 2 – №5, 3 - №6, 1 - №7).

На рис.6.6 а-б) представлены восстановленные поверхности после 500-го и 1000-ого шагов обучения. На рис. 6.6 в-г) в свою очередь показаны графики изменения количества нейронов и фитнесс-функции сети-победителя. Как видно из рисунка 6.6-г), на 501-ом шаге в связи с изменениями параметра объекта произошло резкое изменение значения фитнесс-функции, так как наилучшая сеть оказалась неприспособленной к новым параметрам объекта, однако за последующие шаги алгоритм обучения успешно настроил сеть с учетом изменившихся параметров.

Эксперимент 4. В данном эксперименте решалась задача идентификации нестационарного многомерного объекта (MIMO), который описывался следующими уравнениями:

$$y_1(k)=\begin{cases}\dfrac{15u_1(k-1)y_2(k-1)}{2+50\left[u_1(k-1)\right]^2}+0.5u_1(k-1)-0.25y_2(k-1)+0.1;\ \textit{при}\ 1<k\le 1000;\\ \dfrac{5u_1(k-1)y_2(k-1)}{2+15\left[u_1(k-1)\right]^2}+0.5u_1(k-1)-0.25y_2(k-1)+0.1;\ \textit{при 1000}<k\le 2000;\end{cases}$$

$$y_2(k)=\begin{cases}\dfrac{\sin(\pi u_2(k-1)y_1(k-1))+2u_2(k-1)}{3};\ \textit{при}\ 1<k\le 1000;\\ \sin(\pi u_2(k-1)y_1(k-1));\ \ \textit{при 1000}<k\le 2000.\end{cases}$$

(6.7)

Как видно из уравнений (6.7), после 1000-го шага изменились не только параметры объекта, но и его структура.

Соответствующие поверхности показаны на рис.6.7. На рис.6.7 а-б) показаны поверхности до изменения параметров объекта (6.7) на 1000-ом шаге, а на рис. 6.7 в-г) – после.

Для решения данной задачи использовалась популяция из 128 сетей с четырьмя входами и двумя выходами. В качестве базисных функций использовались гауссовская (№1 в табл.2.1) и вейвлет «мексиканская шляпа» (№2 в табл.2.1). Результаты идентификации объекта (6.7) после 1000 итераций работы генетического алгоритма приведены на рис.6.8 а-б), а после 2000 – на рис.6.8 в-г). Графики, отображающие процесс изменения значения фитнесс-функции для наилучшей особи популяции и структуры соответствующей ей сети, показаны на рис.6.9 а-б).

Эксперимент 5. Решалась задача идентификации многомерного объекта (MIMO), который описывался следующими уравнениями:

$$\begin{aligned}y_1(k)&=\frac{15u_1(k-1)y_2(k-1)}{2+50\left[u_1(k-1)\right]^2}+0.5u_1(k-1)-0.25y_2(k-1)+0.1+\xi_1(k);\\ y_2(k)&=\frac{\sin(\pi u_2(k-1)y_1(k-1))+2u_2(k-1)}{3}+\xi_2(k);\end{aligned}\qquad(6.8)$$

где u_1, u_2 - входные сигналы; y_1 и y_2 - выходные сигналы; ξ_1 и ξ_2 - помехи измерений.

Помеха ξ_1 описывалась моделью (4.3) и имела те же параметры, что и в эксперименте 1. Помеха ξ_2 имела распределение Релея (Ray(1,6)). Таким образом, выходные сигналы объекта были зашумлены различными помехами. Кроме того, так как данный объект является многосвязным, окончательный вид распределений установить весьма сложно. После 2000 эпох обучения были получены следующие оценки параметров помех (независимые по каждому выходу объекта): $\sigma_1^1 = 1.2042$; $\sigma_2^1 = 7.4612$; $m_1^1 = 0.0$; $m_2^1 = 0{,}6765$ - для первого выхода; $\sigma_1^2 = 2.5593$; $\sigma_2^2 = 1.5492$; $m_1^2 = 0.0$; $m_2^2 = 2{,}0294$ - для второго выхода.

Сеть содержала 17 нейронов (9 нейронов с БФ №1 (см. табл.2.1) и 8 – с БФ №2). Результаты моделирования приведены на рис. 6.10. На рис. 6.10-а), б) представлены восстановленные поверхности, на рис. 6.10-в) - график изменения значения фитнесс-функции особи-победителя, на рис. 6.10-г) – график изменения количества активных нейронов особи-победителя.

6.2 Нейросетевое управление многомерными нелинейными объектами

Структура системы адаптивного управления приведена на рис.6.11. Наибольшее распространение при решении данных задач получили многослойный персептрон и радиально-базисные сети [292-294].

Обозначим требуемое значение вектора выходных сигналов объекта как $y^*(k)$. Задача управления заключается в определении значений $u(k)$, обеспечивающих минимум функционалу

$$J(k) = \left\| \boldsymbol{\varepsilon}(k) \right\|^2 = \frac{1}{2} \left\| \boldsymbol{y}^*(k) - \hat{\boldsymbol{y}}(k) \right\|^2 . \qquad (6.9)$$

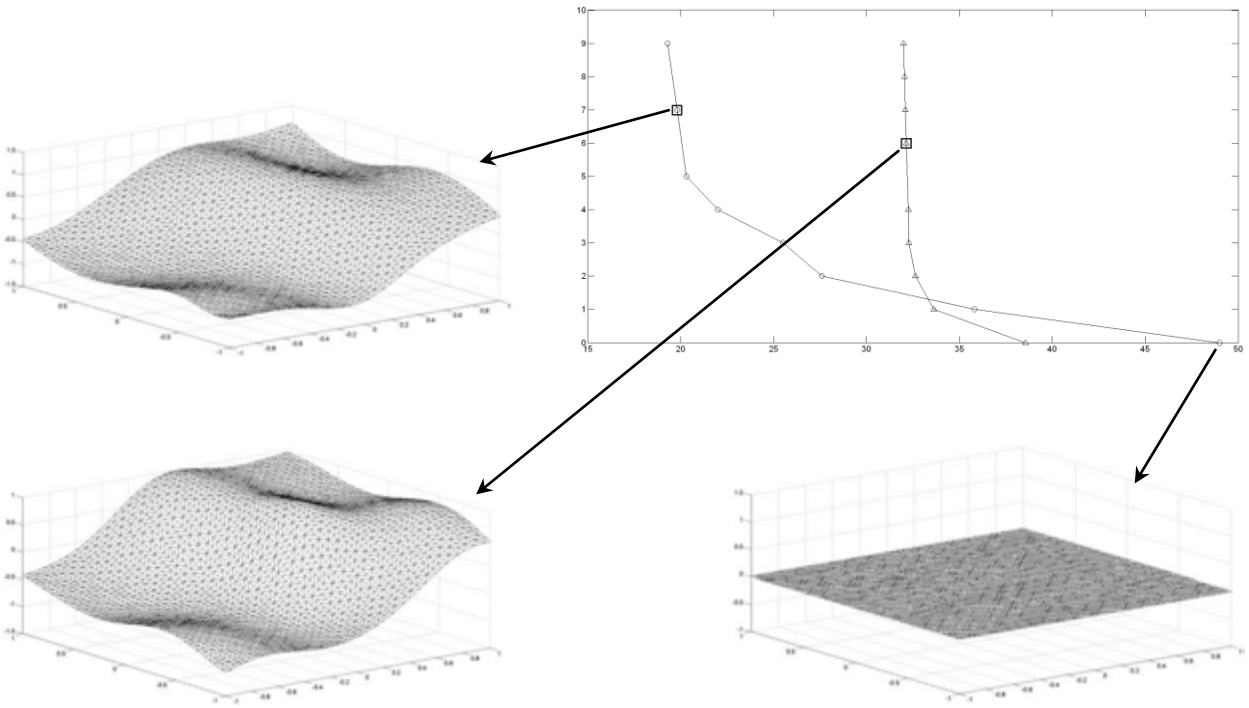
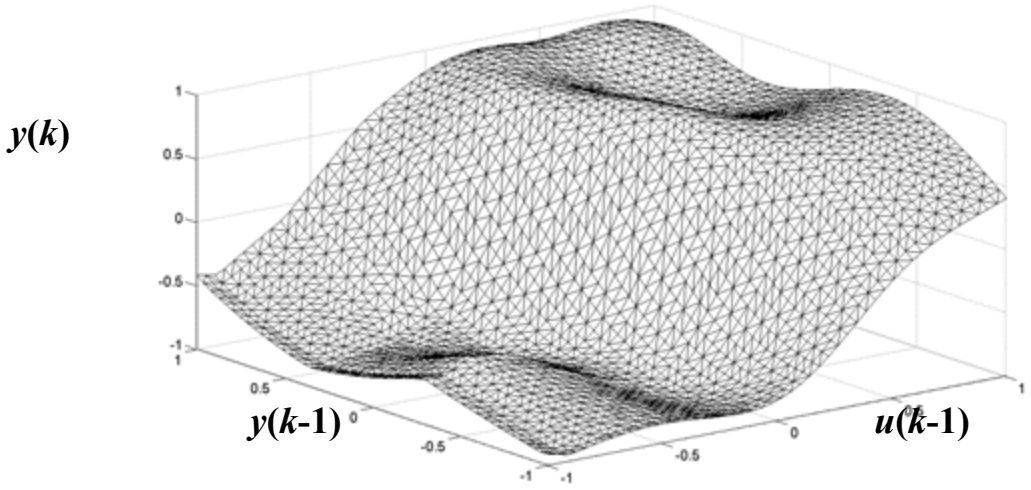
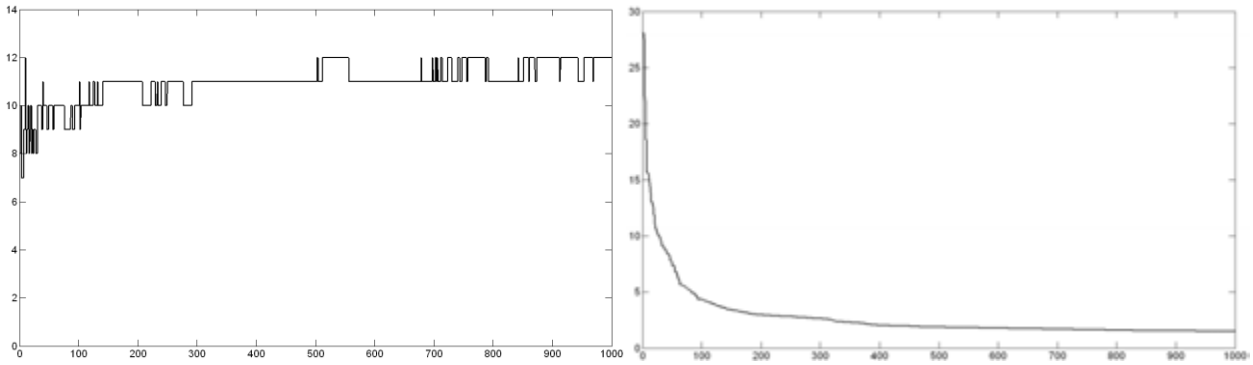


Рисунок 6.4 – Фронты Парето и восстановленные поверхности



а)



б) в)

Рисунок 6.5 – Результаты эксперимента 2

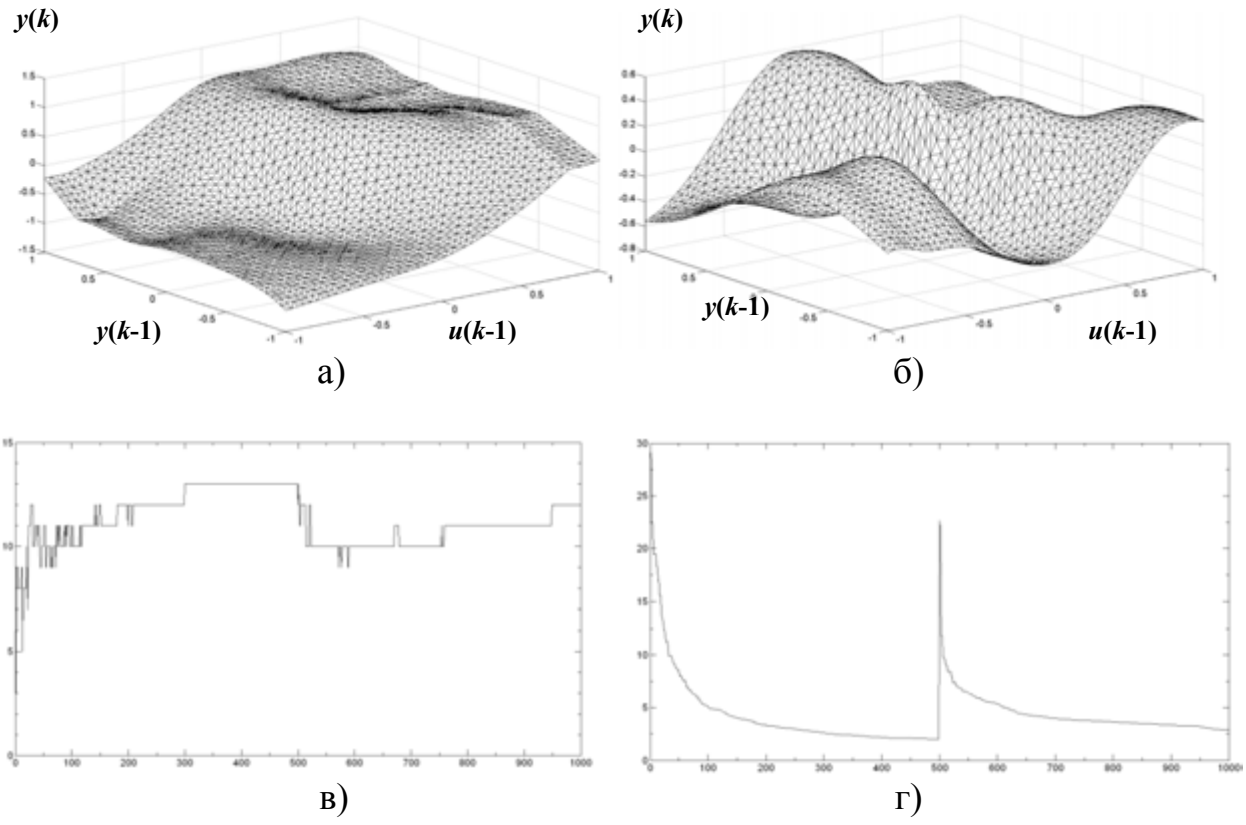


Рисунок 6.6 – Результаты эксперимента 3

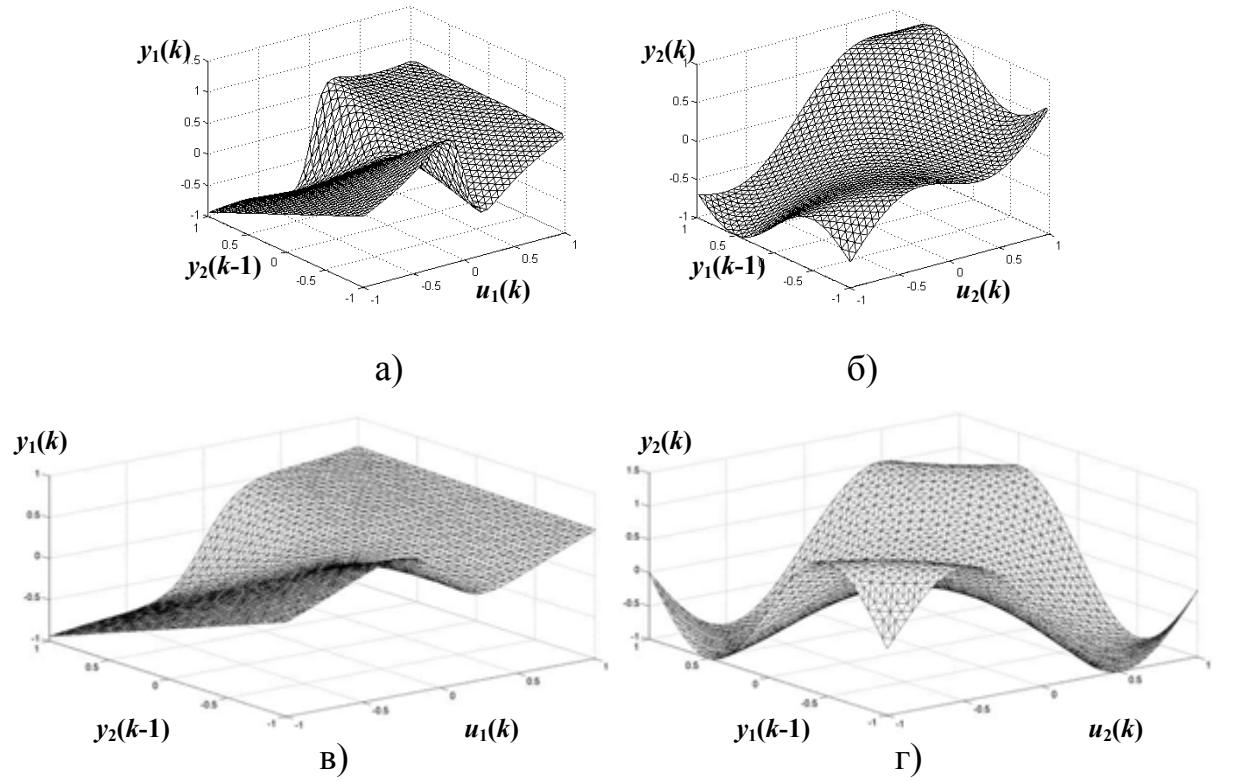


Рисунок 6.7 – Результаты эксперимента 4

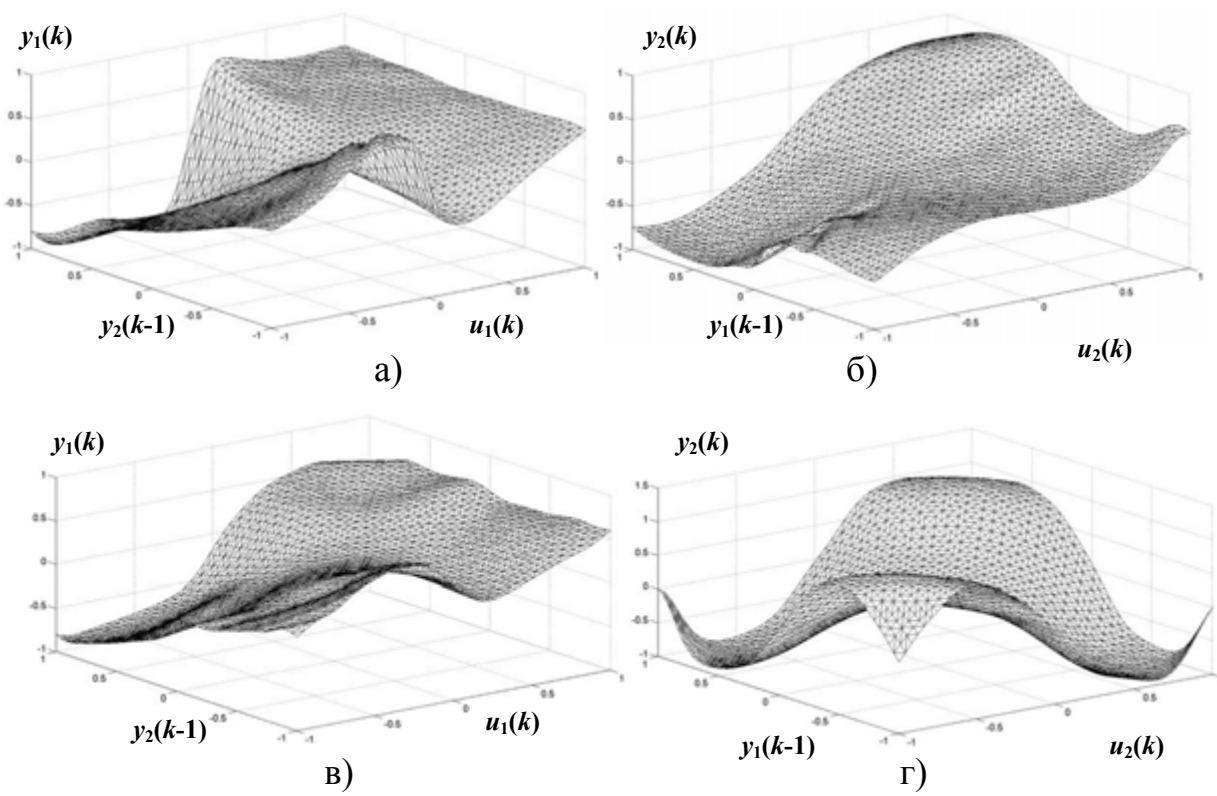


Рисунок 6.8 – Результаты эксперимента 4

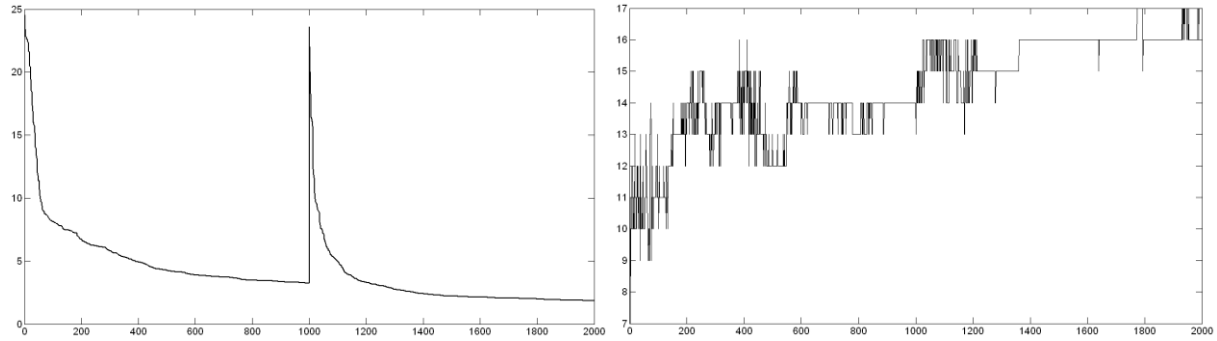
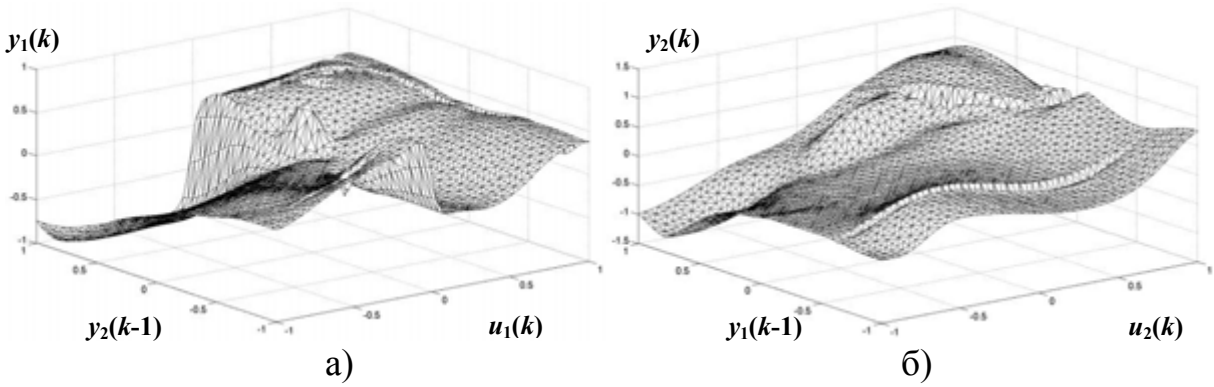
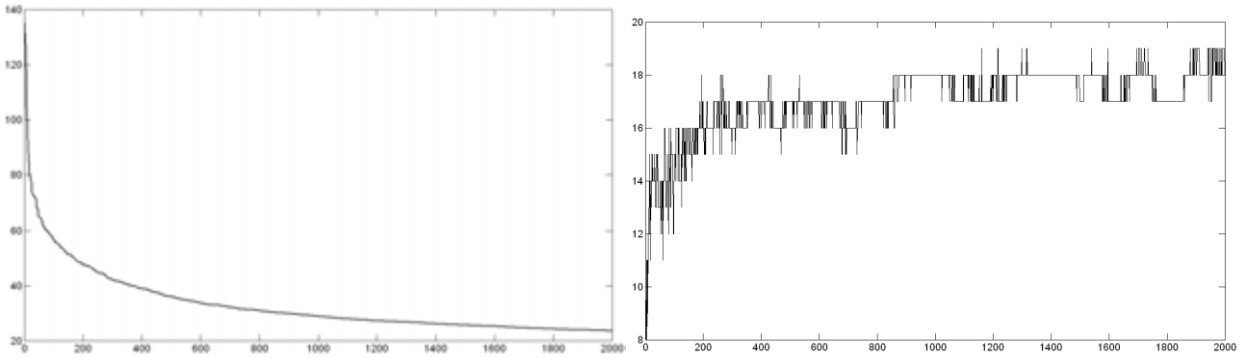


Рисунок 6.9 – Результаты эксперимента 4





в) г)

Рисунок 6.10 – Результаты эксперимента 5

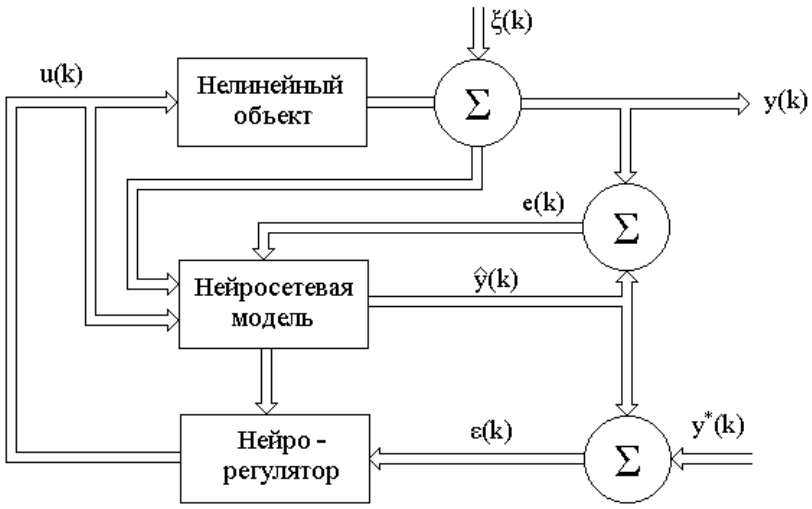


Рисунок 6.11 – Структура системы адаптивного управления

Как и в случае идентификации, для нахождения минимума функционала (6.9) могут быть использованы различные рекуррентные алгоритмы, в частности, градиентные вида

$$u(k) = u(k-1)+\gamma(k)(\nabla_u \varepsilon(k)), \qquad (6.10)$$

где $\nabla_u \varepsilon(k)= \dfrac{\partial \boldsymbol{\varepsilon}(k)}{\partial \boldsymbol{u}(k)}$; $\gamma(k) > 0$.

Выбор оптимального значения $\gamma(k)$, обеспечивающего максимальную скорость сходимости алгоритма (6.10), приводит, как известно, к алгоритму Качмажа [95], который с учетом того, что при управлении объектом (6.1) используется только вектор $u(k–1)$, может быть записан так

$$\boldsymbol{u}(k) = \boldsymbol{u}(k-1) + \nabla_u \boldsymbol{\varepsilon}(k)\left[\delta \boldsymbol{I} + \nabla_u^T \boldsymbol{\varepsilon}(k)\nabla_u \boldsymbol{\varepsilon}(k)\right]^{-1} \boldsymbol{\varepsilon}(k), \qquad (6.11)$$

где $\delta > 0$.

Параметр $\delta > 0$ играет в (6.11) регуляризирующую роль, т.е. используется в алгоритме (6.11) для повышения его вычислительной устойчивости.

Эксперимент 1. Изучалась задача управления нелинейными динамическими объектами, описываемыми различными уравнениями. В качестве входных сигналов при обучении сети использовались некоррелированные случайные последовательности с равномерным законом распределения в интервале [– 1, 1]. Объекты предполагались стационарными. Для обучения сети использовался алгоритм (3.84)-(3.85) с $\lambda = 0.995$, а для управления – (6.11) с различными значениями δ. Обучении осуществлялось на основе предъявления сети 5000 обучающих пар. Некоторые результаты моделирования представлены на рис. 6.12 – рис. 6.15.

Результаты исследования динамического объекта, описываемого уравнениями (6.8) приведены на рис. 6.12-рис.6.14.

На рис. 6.12. а), б) показаны поверхности, описываемые первым и вторым уравнениями системы (6.8) соответственно и восстановленные радиально-базисной сетью с числом нейронов, равным 57. Структура сети уточнялась с использованием неравенств (1.49), (1.50) при задании α_i=0.01, β = 0.8.

Результаты работы нейрорегулятора, реализующий алгоритм (6.7) с δ = 0.05, приведены на рис.6.13 – 6.14. На всех этих рисунках

пунктирной линией показан требуемый выходной сигнал $y_i^*(k)$, сплошной – реальный $\hat{y}_i(k)$, а линией с кружками – соответствующее изменение управляющего сигнала $u_i(k)$ (i=1,2). Требуемые значения выходных сигналов задавались следующие:

$$y_1^*(k) = 0.3\sin(\pi k / 100);$$
$$y_2^*(k) = \begin{cases} 0.1 \;\; при \;\; k = \overline{1,500;} \\ -0.1 \;\; при \;\; k = \overline{501,1000}. \end{cases}$$

Рис.6.13 отражает работу регулятора при отсутствии помех измерения ($\xi(k)$=0) (см. рис.1), а рис. 6.14 – при наличии равномерно распределенной в интервале [-0.3, 0.3] случайной помехи $\xi(k)$. Появление данной помехи привело к тому, что при построении нейросетевой модели увеличилось количество нейронов и стало равным 69, а при управлении моделью – изменился вид управляющего сигнала (особенно это заметно для $u_2(k)$), показанного на рис.6.14 б)). Кроме того, возросла ощибка $\xi_2(k)$.

На рис. 6.15 приведены результаты работы адаптивного нейрорегулятора при управлении объектом, описываемым уравнениями

$$\begin{aligned} y_1(k) &= \frac{y_1(k-1)}{1+\left[y_2(k-1)\right]^2} + u_1(k-1); \\ y_2(k) &= \frac{y_1(k-1)y_2(k-1)}{1+\left[y_2(k-1)\right]^2} + u_2(k-1). \end{aligned} \qquad (6.12)$$

Структура нейросетевой модели определялась при задании $\alpha_i = 0.01$, β = 1.2. Требуемая точность идентификации была достигнута после обучении сети, содержащей 51 нейрон.

Управление данным объектом осуществлялось с помощью алгоритма (6.11) с δ = 0.1. На рис. 6.15 приведены результаты управления объектом (6.12) при задании

$$y_1^*(k) = \cos(\pi k / 100);$$
$$y_2^*(k) = \begin{cases} 1 - 0.004k & при \quad k = \overline{1,500;} \\ 0.5 & при \quad k = \overline{501,1000.} \end{cases}$$

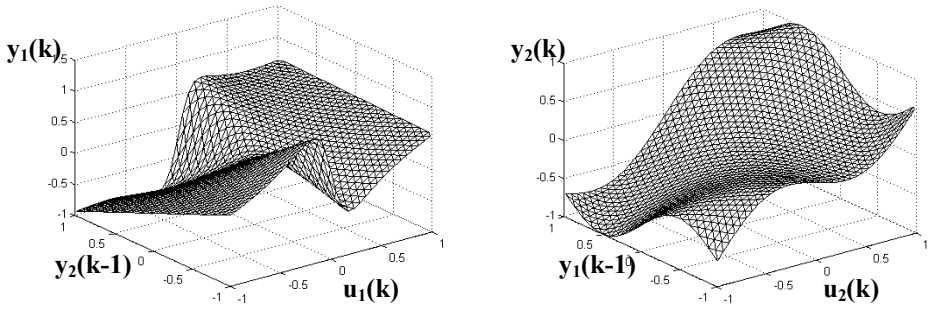
Как видно из рисунков, достигаемая точность управления является высокой.

Выходной сигнал объекта $y_1(k)$ практически совпадает с $y_1^*(k)$. Вид требуемого сигнала $y_1^*(k)$ приводит к необходимости выработки периодического сигнала $u_1(k)$. Эта периодичность сказывается на виде как выходного сигнала $y_2(k)$, так и управления $u_2(k)$.

Резкое изменение на 501 шаге значения требуемого выходного сигнала $y_2^*(k)$ с -1 до +0.5 (рис. 6.15- б) сопровождается возникновением отчетливо видного на этих рисунках переходного процесса, длящегося около 30 тактов. После окончания переходных процессов точность управления вновь остается высокой.

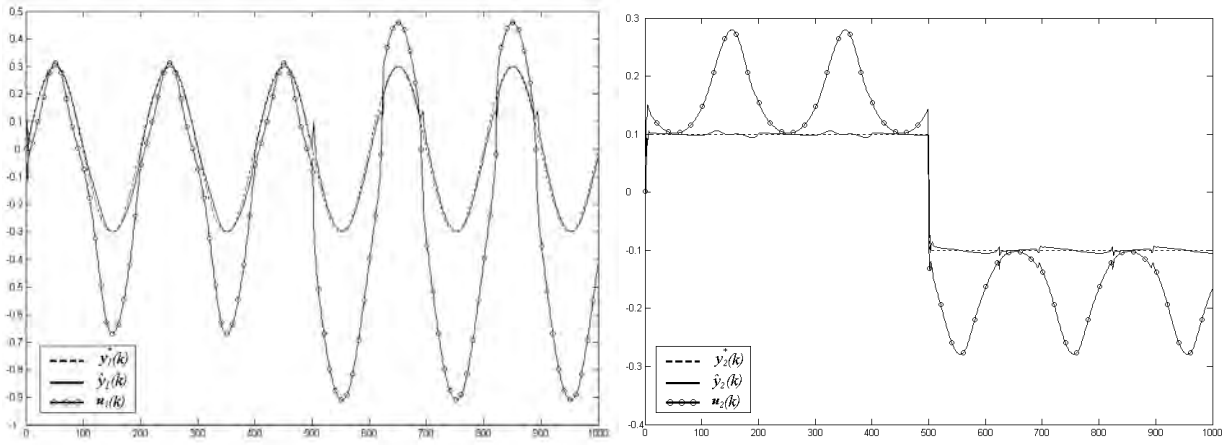
6.3 Задача нейросетевого прогнозирующего управления

Отличительной особенностью получивших в последнее время широкое распространение методов прогнозирующего управления динамическими объектами [295-299] является способность построенных на данном принципе контроллеров работать достаточно длительное время без вмешательства оператора. Среди существующих различных методов проектирования систем управления на основе концепции прогнозирующего управления с моделью (ПУМ), наиболее часто используются такие стратегии, как Dynamic matrix



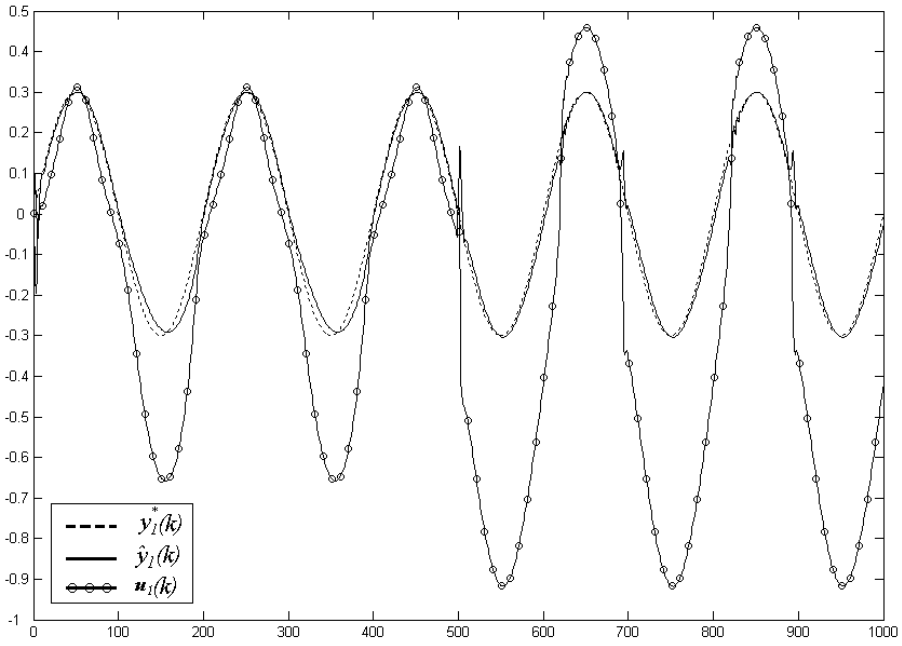
а) б)

Рисунок 6.12 – Поверхности, описываемые уравнениями (6.8)

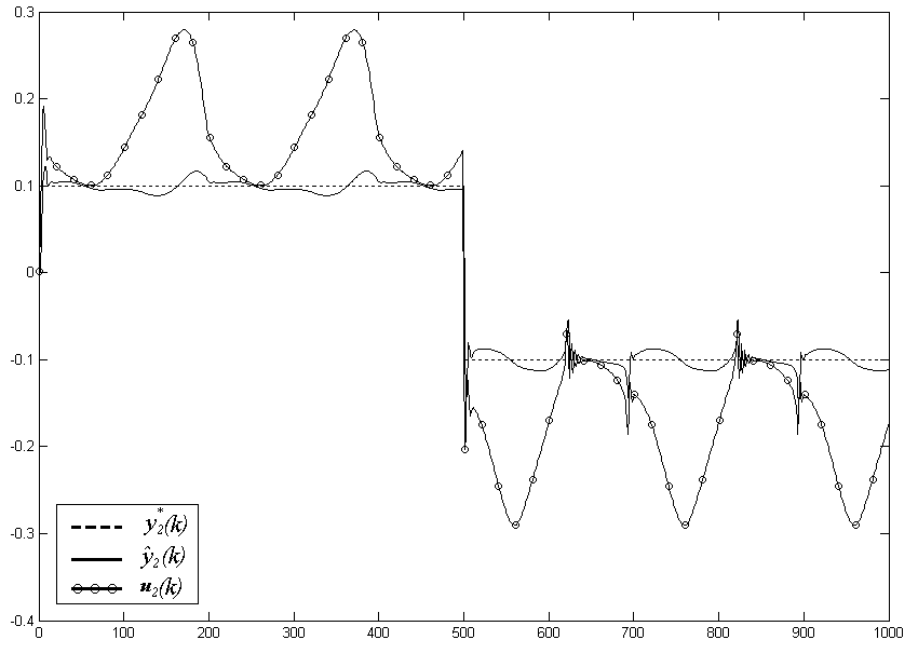


а) б)

Рисунок 6.13 – Управление объектом, описываемым уравнениями (6.8): изменение выходного сигнала $y_1(k)$ и управления $u_1(k)$ *а);* изменение выходного сигнала $y_2(k)$ и управления $u_2(k)$ *б*).

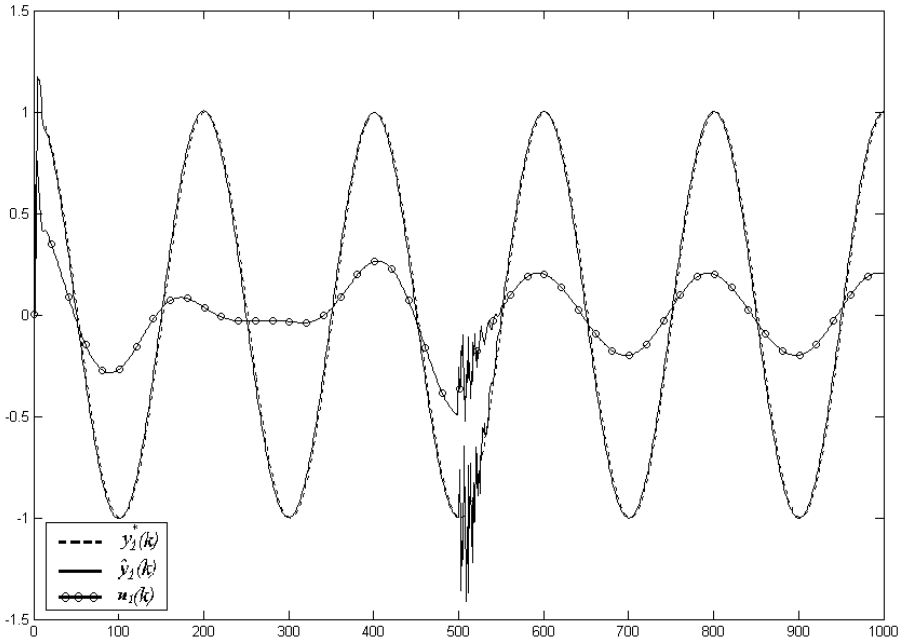


а)

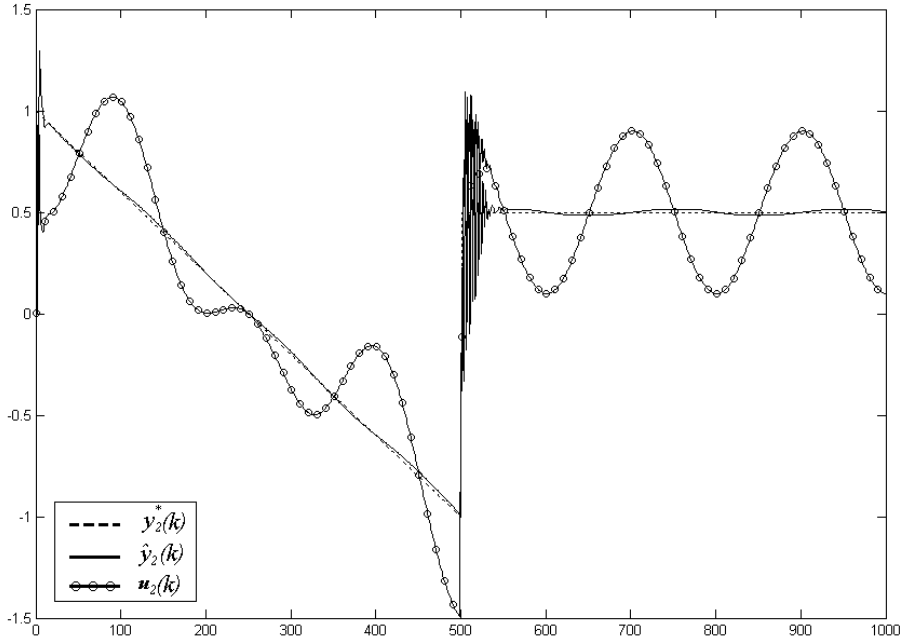


б)

Рисунок 6.14 – Управление объектом, описываемым уравнениями (6.8), при наличии помехи $\xi(k)$: изменение выходного сигнала $y_1(k)$ и управления $u_1(k)$ а)*;* изменение выходного сигнала $y_2(k)$ и управления $u_2(k)$ б).



а)



б)

Рисунок 6.15 – Управление объектом, описываемым уравнениями (6.12): изменение выходного сигнала $y_1(k)$ и управления $u_1(k)$ *а);* изменение выходного сигнала $y_2(k)$ и управления $u_2(k)$ *б*).

control (DMC), Model algorithmic control (MAC), Predictive functional control (PFC), Extended prediction self-adaptive control (EPSAC), Extended horizon adaptive control (EHAC) и Generalized predictive control(GPC) [300].

ПУМ явно использует точную модель процесса для прогнозирования будущего его поведения с целью расчета оптимальной траектории управления (последовательности управляющих сигналов) за счет оптимизации некоторой целевой функции в пределах заданного горизонта предсказания.

В настоящее время достаточно хорошо разработаны методы прогнозирующего управления с линейной моделью, которые применимы и для процессов, не являющихся существенно нелинейными. Действительно, в некоторых случаях оправдывает себя замена нелинейной модели линейной и нахождение параметров регулятора, которые обеспечивали бы наилучшее управление в некоторой компромиссной точке. Хотя такой подход и позволяет в ряде случаев существенно уменьшить априорную неопределенность и реализовать достаточно эффективное управление, ограничение линейными моделями далеко не всегда обеспечивает получение требуемого результата. Поэтому более эффективным представляется разработка систем управления на основе адаптивного подхода в сочетании с методами теории ИНС, что позволяет, с одной стороны, построить довольно простые нелинейные нейросетевые модели, а с другой – адаптивно корректировать параметры как самих моделей, так и регулятора в соответствии с изменяющимися условиями. При этом целесообразным является применение эволюционного подхода к настройке ИНС, позволяющего не только осуществить выбор параметров сети, но и осуществить их настройку.

Рассмотрим задачу управления нелинейным объектом в дискретном времени, описываемом уравнением (6.1).

Традиционная схема управления объектом (6.1) с прогнозирующей моделью с предсказывающей процедурой показана на рис. 6.16. В текущий

момент времени модель используется для предсказания влияния будущего управляющего сигнала $\tilde{u}$ (или «манипулирующей переменной») на выходной сигнал объекта $\tilde{y}$. Предсказание производится на несколько шагов вперед. Учитывая будущие значения эталонного сигнала $\tilde{r}$ и выходного сигнала $\tilde{y}$, будущее изменение управляющего сигнала $\tilde{u}$ определяется с помощью оптимизационного процесса с целью уменьшения предсказанной ошибки.

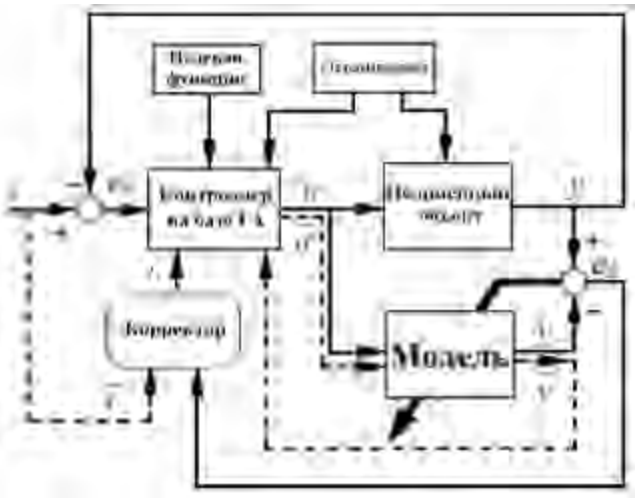


Рисунок 6.16 – Схема управления с прогнозирующей моделью

Принцип работы прогнозирующего контроллера представлен на рис.6.17. На основе модели, которая предполагается известной, определяются будущие значения выходов объекта $y(k+l)$, $l=1,2,...,N_H$, где N_H - горизонт предсказания, которые зависят от его текущих состояний и будущих значений управляющих сигналов, $u(k+m)$, $m=1,2,...,N_C$, где N_C - горизонт управления и $N_C \le N_H$. Прогнозирующий контроллер вычисляет возможные будущие значения сигналов управления таким образом, чтобы значения будущих выходных сигналов были наиболее близкими к требуемым значениям $r(k+l)$, $l=1,2,...,N_H$.

Задача управления при этом заключается в точном отслеживании эталонной траектории $r(k)$ при отсутствии значительных колебаний

управляющего сигнала и с соблюдением при этом всех наложенных ограничений на диапазон управляющего сигнала, градиент управляющего сигнала и допустимый диапазон состояний процесса:

$$\begin{aligned} &u_{\min} \le u(k) \le u_{\max} \ \forall k, \\ &-\Delta u_{\max} \le \Delta u(k) \le u_{\max} \ \forall k, \\ &y_{\min} \le y(k) \le y_{\max} \ \forall k. \end{aligned} \tag{6.13}$$

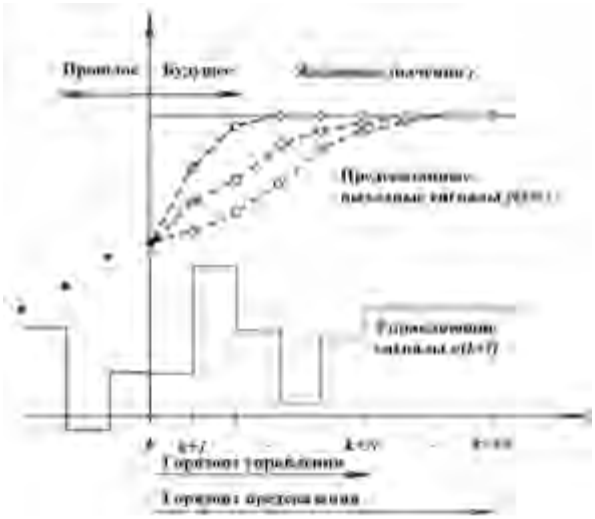


Рисунок 6.17 – Принцип работы прогнозирующего контроллера

Математически цель процесса управления может быть сформулирована как минимизация некоторой целевой функции. В качестве целевой функции при прогнозирующем управлении широко используется функционал вида

$$J(N_0, N_p, N_c) = \sum_{i=N_0}^{N_p} \lambda_p(i)\left[\hat{y}(k+i \mid k) - r(k+i)\right]^2 + \sum_{j=1}^{N_c} \lambda_c(j)\left[\Delta u(k+j-1)\right]^2, \tag{6.14}$$

где N_0 – начало интервала предсказания; λ_p и λ_c – некоторые весовые параметры.

При реализации нейросетевого прогнозирующего управления в качестве модели объекта (6.1) используется нейросетевая модель, для

построения которой наиболее часто применяются многослойный персептрон (МП) и радиально-базисная сеть (РБС).

Обучение многослойного персептрона, содержащего несколько (чаще всего не более двух) скрытых слоев, осуществляется обычно с помощью алгоритма обратного распространения ошибки, реализация которого связана с существенными вычислительными трудностями. Более простой архитектурой обладают радиально-базисные сети (они состоят из одного слоя нейронов). Однако для их обучения также используются градиентные алгоритмы, требующие значительных вычислительных затрат, что усложняет их применение в системах управления в реальном времени. В связи с этим возникла новая область, объединяющая нейронные и эволюционные парадигмы [4, 5], в основном направленная на объединение алгоритмов обучения с теорией эволюции (эволюция весов ИНС, архитектуры, алгоритмов обучения и активационных функций).

Подробно процесс построения моделей нелинейных объектов на основе эволюционирующих ИНС прямого распространения рассматривается в работах [118, 145, 301-302].

После окончания процесса обучения сеть используется для реализации алгоритма управления с предварительной коррекцией эталонной траектории.

Как и в случае идентификации, для нахождения минимума функционала (6.14) могут быть использованы различные рекуррентные алгоритмы, в частности, градиентные вида (6.10).

При использовании модели засорения (4.3) можно с помощью некоторых процедур настроить величины s_1^2 и s_2^2, являющиеся оценками σ_1^2 и σ_2^2, и осуществить коррекцию эталонной траектории управления с помощью следующей процедуры

$$r_c(k+h) = r(k+h) - \hat{e}^2(k), \qquad (6.15)$$

где $h = N_0,...,N_p$;

$$\hat{e}^2(k) = \begin{cases} \dfrac{\left(y_j^*(x_j) - \hat{y}_j(x_j)\right)^2}{s_1^2(k)}, \text{если } \left|y_j^*(x_j) - \hat{y}_j(x_j)\right| \le 3s_1(k); \\ \dfrac{\left(y_j^*(x_j) - \hat{y}_j(x_j)\right)^2}{s_2^2(k)}, \text{в противном случае.} \end{cases} \quad (6.16)$$

Скорректированная с помощью процедуры (6.15) эталонная траектория подается на контроллер, который и осуществляет задачу минимизации функционала (6.14).

Эксперимент 1. Решалась задача управления нелинейным объектом, описываемым уравнением

$$y(k+1) = \sin(u(k)) + \frac{y(k)}{1 + y^2(k)}, \quad (6.17)$$

где $u(k)$ – сигнал управления.

Вначале производилась идентификация объекта с помощью эволюционирующей РБС. Начальная популяция состояла из 128 особей (ИНС). На каждой итерации алгоритма каждой особи предъявлялось $M = 5000$ обучающих пар.

После окончания процесса идентификации производилось прогнозирующее управление объектом (6.17) с помощью контроллера, использующего для вычисления управляющего воздействия ГА. Популяция состояла из 20 особей. Каждая особь была закодирована хромосомой, содержащей информацию о будущих N_H управляющих воздействиях и о внутренних параметрах контроллера и предиктора.

Желаемый закон изменения выходного сигнала объекта представлял собой пилообразный сигнал, изменяющийся в диапазоне $[-0.5,0.5]$. Результаты управления объектом (6.17) при различных значениях горизонта предсказания приведены на рис.6.18.Так для рис. 6.18-а) $N_H = 10$, для рис.

6.18- б) - $N_H = 20$, а для рис. 6.18-в) - $N_H = 30$. На всех рисунках сплошной линией обозначен желаемый выходной сигнал, пунктирной – реальный выходной сигнал, сплошной с кружками – управляющий сигнал.

Как следует из рис. 6.18, с увеличением горизонта предсказания происходит сглаживание результатов и повышается точность управления.

Эксперимент 2. Решалась задача идентификации зашумленного объекта, который описывался уравнением

$$y(k+1) = \frac{u(k)[\cos(y(k)) + 0.5]}{2 + y^2(k)} + \xi(k), \qquad (6.18)$$

где $\xi(k)$ - нормально распределенная помеха с $\sigma^2 = 0.6$.

Желаемое значение выходного сигнала определялось по формуле

$$r(k+1) = 0.6 + 0.07\sin(\pi k/200) - 0.05\cos(\pi k/100). \qquad (6.19)$$

Для идентификации объекта (6.18) использовалась популяция, состоящая из 150 особей. На каждом из 500 этапов эволюции каждой особи предъявлялось 5000 обучающих пар. В ходе работы алгоритма управления были оценены параметры помехи в соответствии с моделью (4.3). При этом были получены следующие значения: $s_1^2 = 0.6074$, $s_2^2 = 2.128$, $\varepsilon = 0.0021$.

Результаты управления объектом приведены на рис.6.19. Здесь, как и в предыдущем эксперименте сплошной линией обозначен желаемый выходной сигнал, пунктирной – реальный выходной сигнал, сплошной с кружками – управляющий сигнал.

Результаты моделирования свидетельствуют о том, что рассматриваемый подход позволяет эффективно решать задачу управления при наличии помех измерений.

Эксперимент 3. Изучалась задача прогнозирующего управления многомерным нелинейным динамическим объектом, описываемым уравнениями

$$\begin{aligned} y_1(k) &= \frac{15u_1(k-1)y_2(k-1)}{2+50[u_1(k-1)]^2} + 0.5u_1(k-1) - 0.25y_2(k-1) + 0.1 + \xi_1(k); \\ y_2(k) &= \frac{\sin(\pi u_2(k-1)y_1(k-1)) + 2u_2(k-1)}{3} + \xi_2(k), \end{aligned} \tag{6.20}$$

где $\xi_1(k)$ и $\xi_2(k)$ - помехи измерений.

В качестве входных сигналов при обучении сети использовались некоррелированные случайные последовательности с равномерным законом распределения в интервале $[-1,1]$. Для обучения сети использовалась популяция, состоящая из 250 особей. На каждом из 500 этапов эволюции каждой особи предъявлялось 10000 обучающих пар. После окончания процесса идентификации, осуществлялось управление многомерным объектом (6.20). Требуемые значения выходных сигналов задавались следующие:

$$y_1^*(k) = \begin{cases} -0.5 + k/100, k = \overline{1,100}; \\ 1.5 - k/100, k = \overline{101,200}; \\ -2.5 + k/100, k = \overline{201,300}; \\ 3.5 - k/100, k = \overline{301,400}; \\ -4.5 + k/100, k = \overline{401,500}; \\ 0.2\sin(\pi k/100) + 0.05\cos(\pi k/200), k = \overline{501,1000}; \end{cases}$$

$$y_2^*(k) = \begin{cases} 0.2, k = \overline{1,500}; \\ 0.1, k = \overline{501,1000}. \end{cases}$$

Результаты работы нейрорегулятора показаны на рис.6.20. На всех рисунках пунктирной линией показан требуемый выходной сигнал $r_i(k)$,

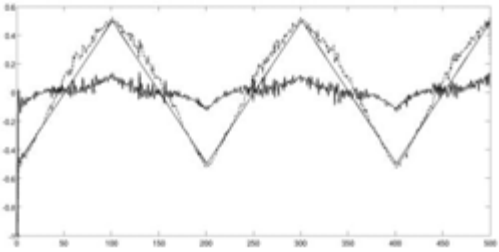
сплошной – реальный $\hat{y}_i(k)$, а линией с кружками – соответствующее изменение управляющего сигнала $u_i(k), i = 1,2$. Рисунки 6.20-а) и 6.20-б) отражают результаты моделирования при отсутствии помех измерений, т.е. $\xi_1(k) = 0$, $\xi_2(k) = 0$. На рисунках 6.20-в) и 6.20-г) приведены результаты моделирования при наличии помех измерений по обоим каналам управления, при этом $\xi_1(k)$ являлась нормально распределенной помехой с $\sigma^2 = 0.6$, а $\xi_2(k)$ - с $\sigma^2 = 0.8$. Оценки параметров помех в соответствии с моделью (4.3) дали следующие значения: $s_1^2 = 0.5912, s_2^2 = 1.9558$, $\varepsilon = 0.0023$ для первого канала управления и $s_1^2 = 0.7982, s_2^2 = 2.5847$, $\varepsilon = 0.0028$ - для второго.

Результаты моделирования свидетельствуют о том, что развиваемый подход позволяет синтезировать нейросетевой контроллер для управления многомерными нестационарными объектами при наличии помех измерений.

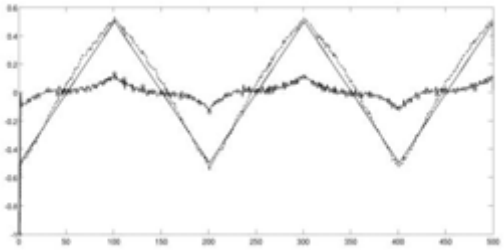
6.4 Нейросетевое сжатие изображений на основе ЭИНС

Непрерывное возрастание объемов перерабатываемой, в частности, зрительной, информации обусловливает необходимость ее компактного представления. Применение с этой целью традиционных методов, использующих префиксное или арифметическое кодирование либо обеспечивающих сжатие без потерь, требует значительных вычислительных ресурсов. В то же время наличие в мультимедийной информации операции дискретизации изображения и звука позволяет ставить и решать задачу эффективного сжатия информации с потерями.

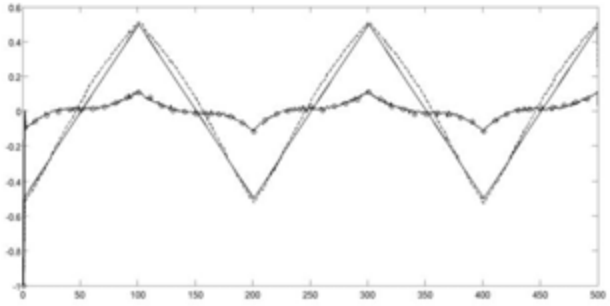
В этих условиях перспективным представляется развитие подхода, в основе которого лежат ИНС. При этом ИНС могут использоваться как при сжатии без потерь, так и при реализации сжатия с потерями, как например, в стандарте JPEG 2000, основанном на вейвлет-преобразовании.



а)



б)



в)

Рисунок 6.18 – Результаты управления объектом (6.17)

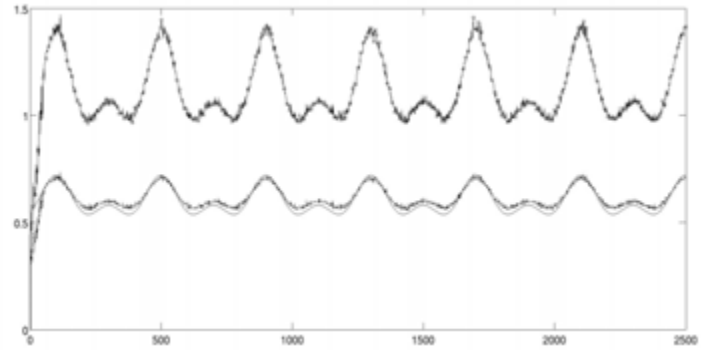
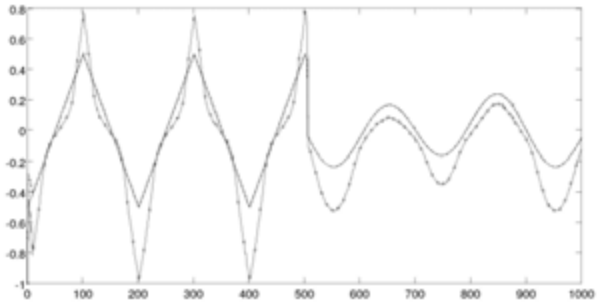
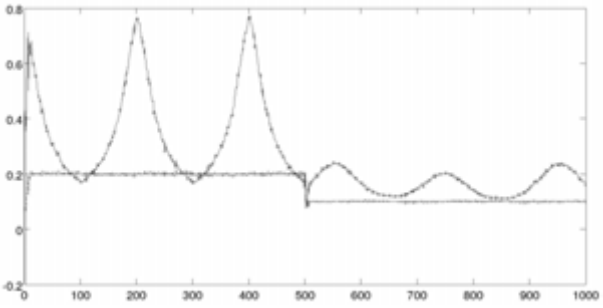


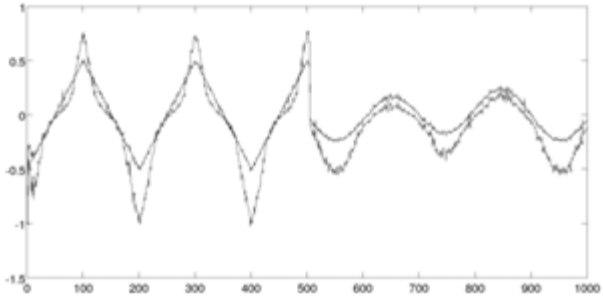
Рисунок 6.19 – Результаты управления объектом (6.18)



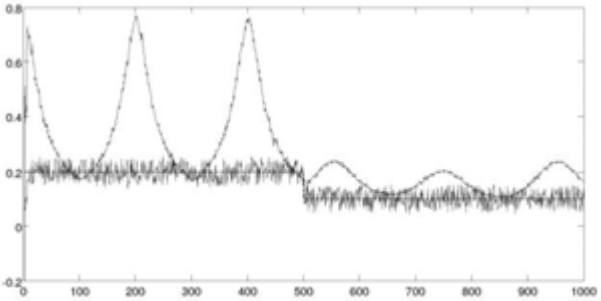
а)



б)



в)



г)

Рисунок 6.20 – Результаты управления объектом (6.20)

Наиболее значительные результаты в этом направлении получены на основе самоорганизующихся карт Кохонена [303] и сетей встречного распространения, также содержащих слой нейронов Кохонена [1, 38]. Самоорганизация в этих сетях представляет собой процесс кластеризации образов, осуществляемой аналогично методу главных компонент.

В работе [141] предложена модификация сети Кохонена, использующая уравнивание длин всех входных векторов к одной величине, не к единице, как в классической сети Кохонена, а к некоторому другому значению. Это осуществляется путем использования дополнительной компоненты входного сигнала X_{mod}, вычисляемой следующим образом:

$$X_{\text{mod}} = \sqrt{A_{\max} - \sum_{i=0}^{n} X_i^2},$$

где $A_{\max}$ – максимально представимое число либо максимально допустимая разрядность числа вычислительного устройства или АЦП.

Очевидно, что в данном случае для вычисления нейрона-победителя можно также использовать выходы сети, а не вычислять евклидово расстояние. При этом операция по вычислению дополнительного входа будет занимать меньше времени, чем операция их нормализации, так как на каждом шаге обучения вычисления проводятся только для одного входа, а не для каждого из входов, как в обычной нормализации. Хотя добавление дополнительного входа увеличивает время вычисления выхода, однако это всё же быстрее, чем вычисление аналогичного евклидового расстояния.

Следует отметить, что такая модификация не влечет за собой никаких изменений в способе обучения сети, так как для сети, по сути, добавился просто еще один вход, поэтому обучение модифицированной сети Кохонена осуществляется по стандартному алгоритму [141].

Предложенная модификация может быть без труда применена практически к любым уже существующим реализациям сети Кохонена.

Несложно увидеть, что после обучения сети, веса каждого из нейронов (за исключением веса, связанного с дополнительным входом) можно использовать как вектор-представитель класса. Таким образом, после данной модификации, сеть Кохонена может быть применена для решения тех задач, которые решались только с применением сетей векторного квантования.

Такая же модификация может быть легко применима к известной сети «нейро-газ», имеющей такую же структуру, как и сеть Кохонена, и отличающаяся от последней способом обучения [304-306].

Кроме того, в работе [141] предложена аналогичная модификация РБС. Хотя РБС работает с нормализованными данными лучше, в задаче сжатия изображений нормализация входных данных не представляется возможной, так как сеть восстановит нормализованное изображение, что приведет к потере качества сжатия. Однако РБС также может быть использована для сжатия изображений, но уже не путем кластеризации пикселей, а методом, известным как "бутылочное горлышко". Суть его заключается в том, что нейронная сеть имеет скрытый слой меньшей размерности, чем размерности входного и выходного слоев, которые как правило, одинаковы.

Само сжатие осуществляется благодаря тому, что вместо данных изображения для каждого блока изображения сохраняются значения выходов скрытого слоя. Таким образом, сеть восстановит весь входной вектор, включая нормализующую компоненту. На выходе же для получения реального исходного значения входного вектора эта компонента должна быть отброшена.

Таким образом, по аналогии с рассматриваемыми выше сетями, РБС следует модифицировать путем добавления кроме нормализующей компоненты входного вектора, нормализующую компоненту выходного сигнала. Тогда сеть будет пытаться восстановить входной вектор, который на одну размерность больше чем реальный, но при этом нормализован.

При моделировании процесса сжатия для оценки качества сжатия использовались показатели PSNR (peak signal-to-noise ratio,) – отношение

максимально возможного уровня сигнала к уровню искажающего его шума (поскольку большинство сигналов имеют очень широкий динамический диапазон, PSNR обычно представляют в логарифмическом масштабе), и MSE (mean squared error) - среднеквадратичная ошибка.

Для двух монохромных изображений I и K размерностью $m \times n$ (где одно из изображений является зашумленным представлением второго) MSE вычисляется следующим образом:

$$MSE = \frac{1}{mn} \sum_{i=0}^{m-1} \sum_{j=0}^{n-1} \left\| I(i,j) - K(i,j) \right\|^2 .$$

Для цветных изображений с тремя RGB компонентами MSE определяется как сумма всех квадратичных разностей, деленная на размер изображения и на 3.

Показатель PSNR определяется так:

$$PSNR = 10 \cdot \log_{10} \left(\frac{MAX_I^2}{MSE} \right) = 20 \cdot \log_{10} \left(\frac{MAX_I}{\sqrt{MSE}} \right),$$

где MAX_I представляет собой максимальное значение пикселя в изображении. В случае, если пиксели представлены 8-ми битовыми значениями, MAX_I = 255. В общем случае при использовании для представления B бит максимально возможное значение для MAX_I равно 2^B – 1.

Обычно для алгоритмов сжатия PSNR находится в диапазоне 30-40 dB.

В эксперименте использовалось эталонное изображение «Lenna» (рис.6.21-а) размером 512x512 пикселей, размеры кодируемой области выбирались 1x1 и 4x4 пикселей.

При моделировании модифицированной сети Кохонена использовалось 128 нейронов. В алгоритме обучения выбиралось $\alpha(t) = \alpha_0 \left(\alpha_f / \alpha_0\right)^{t/t_{\max}}$ (в данном эксперименте α_0 – начальное значение (1,0); α_f – конечное значение (0,005)). При исследовании сети «нейро-газ» также использовалось 128 нейронов, в алгоритме обучения $\alpha(t)$ применялось как в сети Кохонена, а правило изменения $\lambda(t)$ было аналогично правилу для $\alpha(t)$, за исключением лишь начальных параметров (для данного эксперимента λ_0 =10; λ_f = 0,01, выбраны экспериментально).

При исследовании работы модифицированной РБС входной и выходной слои содержали по 48 нейронов, а количество нейронов скрытого слоя 15 выбрано экспериментальным путем на основании правила, что число нейронов скрытого слоя должно быть гораздо меньше числа нейронов скрытого слоя. Количество циклов обучения составляло 100. В качестве базисной функции для РБС использовалась функция (2.7), а обучение осуществлялось с помощью алгоритма (3.132)-(3.133).

На рис. 6.21-б) показаны некоторые результаты сжатия изображения «Lenna» предложенными модифицированными сетями. Так как полученные результаты визуально мало чем отличаются для большей наглядности они сведены в табл. 6.1, 6.2. Здесь использованы следующие сокращения: СВР – сеть встречного распространния, МСК – модифицированная сеть Кохонена, РБС – радиально-базисная сеть, МСБР – модифицированная РБС, НГ – сеть «нейро-газ», МНГ – модифицированная НГ. Как следует из результатов моделирования, введение дополнительного нормализующего входа в ряде случаев позволяет существенно увеличить качество сжатия информации. Для сети «нейро-газ» результаты сжатия с нормализующим входом качественно немного хуже, скорее всего это связано с тем, что веса всех (включая добавленный) нейронов участвуют в вычислении расстояния между вектором весов и входным образом.

а) б)

в) г)

Рисунок 6.21 – Эталонное изображение «Lenna» а) и результаты сжатия изображения сетью встречного распространения б); модифицированной сетью РБС в); сетью «Нейро-Газ» с нормализующим входом г)

Таблица 6.1 – Сравнение значений коэффициент PSNR

Название компоненты	Значение коэффициента PSNR					
	СВР	МСК	РБС	МРБС	НГ	МНГ
R	27,16	34,98	21,46	25,72	35,26	34,36
G	30,09	35,50	24,40	26,25	35,48	34,41
B	31,87	34,88	24,14	25,75	34,93	33,68
Среднее по компонентам	29,27	35,12	23,33	25,91	35,23	34,15

Таблица 6.2 – Сравнение времени, затраченного на процесс компрессии и декомпрессии

Название сети	Время, затраченное на процесс компрессии и декомпрессии	Количество настраиваемых весов	Количество циклов обучения сети
Для области 1x1 пиксель			
Сеть встречного распространения	00:06.06	768	2
Модифицированная сеть Кохонена	00:05.01	5122	2
Сеть РБФ	02:04.30	180	100
Модифицированная сеть РБФ	02:32.71	240	100
Для области 4x4 пикселя			
Сеть встречного распространения	00:05.01	12288	2
Модифицированная сеть Кохонена	00:03.39	8192	2
Сеть РБФ	01:36.80	2880	100
Модифицированная сеть РБФ	01:39.82	3840	100
Сеть «нейро-газ»	00:37.77	384	2
Модифицированная сеть «нейро-газ»	00:42.29	512	2

6.5 Управление технологическими процессами в сахарной промышленности

Современное сахарное производство представляет собой сложный и энергоемкий процесс. Основными этапами производства сахарной продукции являются: приемка и хранение свеклы; транспортировка, очистка, мойка; извлечение сока; очистка сока; выпаривание; кристаллизация и получение белого сахара; обессахаривание оттеков и возврат желтого сахара [307-309]. Упрощенная структурная схема современного сахарного завода представлена на рис. 6.22.

Кратко опишем структурную схему производства сахара. Выращенную сахарную свеклу отправляют на завод, где ее сохраняют до надобности либо сразу пускают в переработку. Технология переработки заключается в том, что

- очищают от тяжелых и легких примесей;
- моют;
- взвешивают;
- измельчают и извлекают из свекловичной стружки сок путем выщелачивания (диффузией) веществ клеточного сока при помощи воды;
- сок обрабатывают известью и осаждают вещества, образующие с ионами кальция нерастворимые соли (дефекация);
- избыток извести осаждают углекислым газом и дополнительно отделяют несахаристые вещества адсорбцией на поверхности углекислого кальция (1-я сатурация);
- полученный осадок отделяют фильтрацией, а сок повторно обрабатывают известью и углекислым газом (2-я сатурация);
- сок снова фильтруют, обрабатывают сернистым газом для обесцвечивания (сульфитация) и фильтруют;
- очищенный сок выпаривают в многокорпусной выпарной станции, сульфитируют и снова фильтруют.

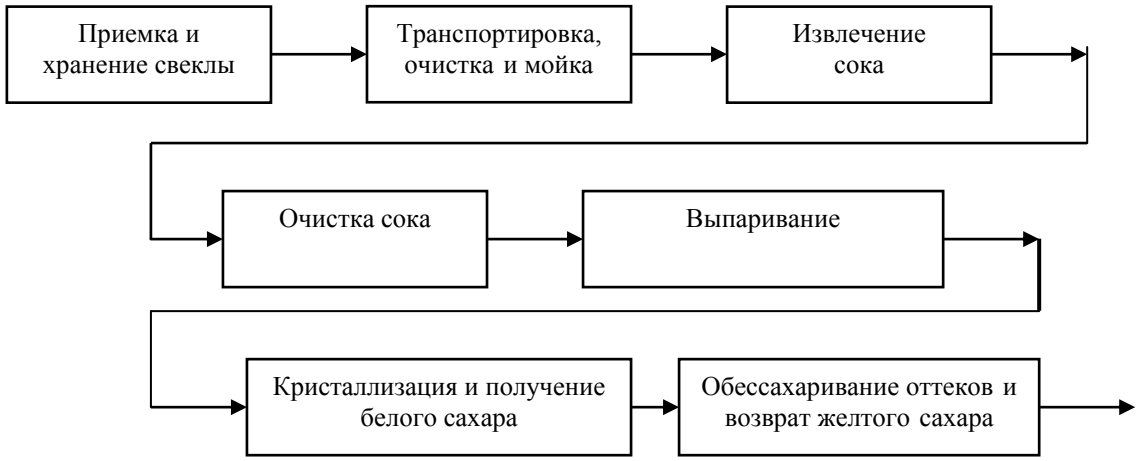


Рисунок 6.22 – Упрощенная структурная схема сахарного завода

Окончательным этапом производства сахара является уварка очищенного сиропа, при которой растворитель (вода) удаляется, раствор становится насыщенным и из него выделяется кристаллическая сахароза. Кристаллы сахарозы отделяют центрифугированием, промывают и высушивают, т.е получают товарный сахар.

Стандартная технологическая схема переработки сахарной продукции представлена на рис. 6.23.

Анализ схемы производства сахара показывает, что основным отделением, которое задает и определяет количество переработанной сахарной свеклы, является диффузионное отделение (рис.6.24).

Уравнение теплообмена для любой зоны сокостружечной смеси получают составляя уравнение теплового баланса.

Как видно из модели теплообменной части диффузионного аппарата (ДА), данный объект является многосвязным. Поэтому для управления таким объектом использовались многомерные оптимальные регуляторы [310, 311].

Структурная технологическая схема работы оборудования в диффузионном отделении сахарного завода представлена на рис. 6.24.

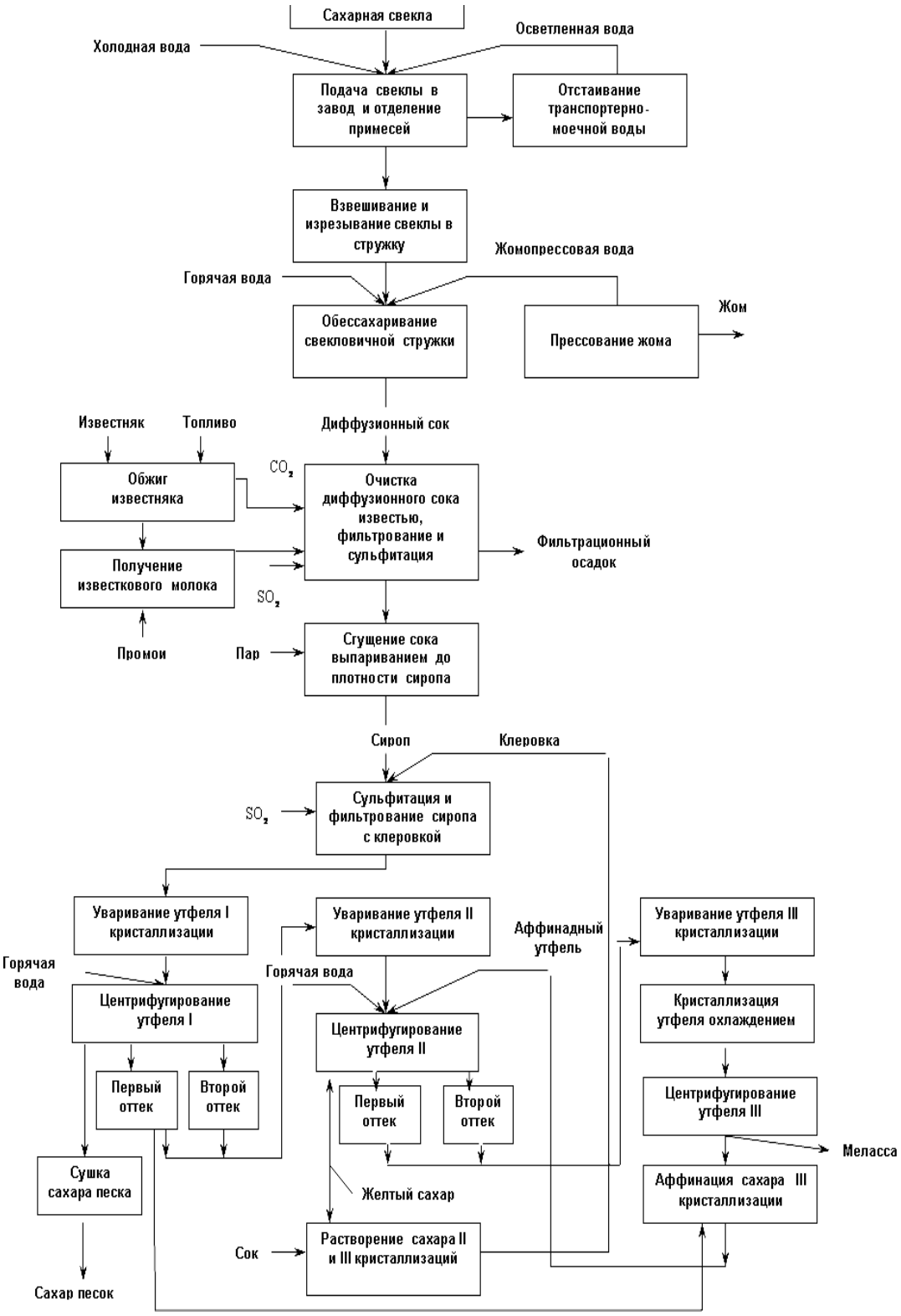


Рисунок 6.23 – Стандартная технологическая схема переработки сахарной продукции

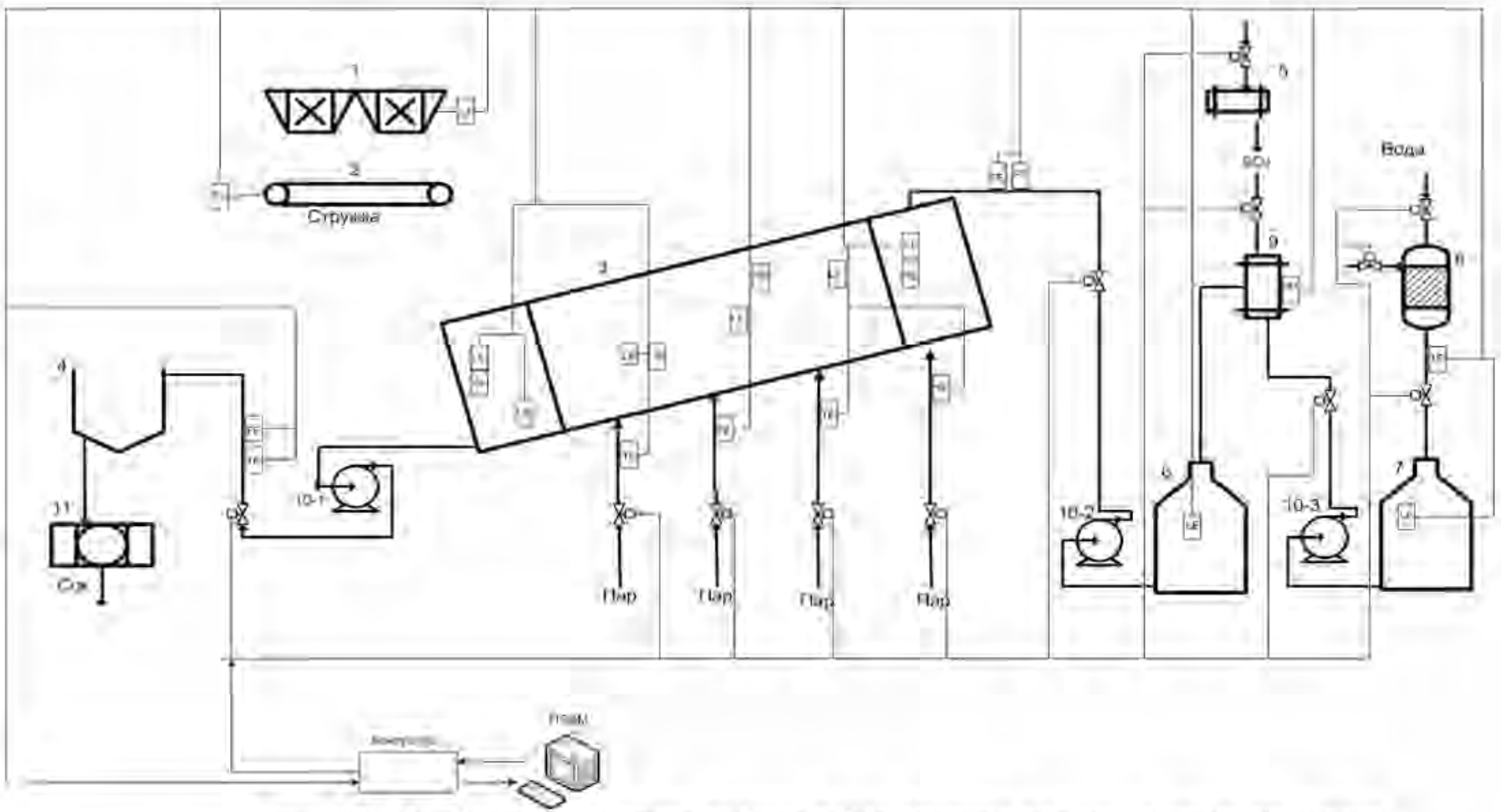


Рисунок 6.24 – Структурная схема работы диффузионного отделения сахарного завода

где 1 – свеклорезки с бункером свеклы; 2 – ленточный конвейер; 3 – диффузионный аппарат DC-8; 4 – бункер диффузионного сока; 5 - печь; 6 – сборник барометрической сульфитированной воды; 7 – сборник барометрической воды; 8 – подогреватель воды; 9 – сульфитатор; 10-(1,2,3) – насосы; 11 – пульповловушка; LE - уровнемер; FE- расходометр; TE- датчик температуры; pH – метр.

Одной из основных задач при разработке автоматизированной системы управления ДА является поддержание заданного температурного режима по зонам наклонной диффузионной установки. Кроме того, при работе диффузионной установки необходимо поддерживать оптимальные значения качественных показателей, в частности концентрацию сухих веществ в диффузионном соке и количество сахара в жоме.

Типовые схемы систем автоматизации ДА предполагают [312-314]: стабилизацию удельной нагрузки аппарата, стабилизацию концентрации диффузионного сока, стабилизацию температурных режимов по зонам диффузионного аппарата, стабилизацию уровня в главной части аппарата; измерения потерь стружки, диффузионного сока, жомовой и сульфитированой воды; pH сокостружечной смеси.

При производстве сахарной продукции в настоящее время расчеты производятся на основании тепловых балансов, так как на всех этапах производства присутствуют жесткие ограничения по температурным режимам. Поэтому и уравнение теплообмена для любой зоны сокостружечной смеси в диффузионном отделении получают, составляя уравнение теплового баланса (пример таких уравнений приведен в Приложении И). Получаемые при этом модели являются линейными, стационарными и описываемыми уравнениями в частных производных.

Учет нестационарности исследуемых процессов и отсутствие достаточно полной статистической информации возможен при реализации нейросетевого адаптивного управления.

При построении нейросетевых моделей диффузионного отделения (рис. 6.24) рассматривались два случая:

1. В качестве выходной величины отделения, а соответственно и диффузионного аппарата (ДА), так как ДА является основным оборудованием, задающим режим работы не только всего отделения, а и всего завода, использовалась одна выходная переменная - расход диффузионного сока $(G_{дс}, м^3 / ч)$.

2. По аналогии с дифференциальными уравнениями, приведенными в Приложении И и описывающими зависимости температур сокоструженой смеси отдельно в каждой зоне ДА ($\theta_i = \overline{1,4}$) от других параметров процесса, определялись регрессионные зависимости этих температур от управляемых параметров аппарата.

Некоторые результаты моделирования процесса диффузии (для 4-х выходных параметров) приведены на рис. 6.25. Данные нелинейные модели по различным каналам $y_i(k) = f(u_1(k),\ldots,u_{11}(k))$, $i = \overline{1,4}$, $\mathrm{k} = \overline{1,100}$ были получены в работах [315-316] с помощью пяти двухслойных персептронов 2-7-5-1, обучение которых осуществлялось по алгоритму Левенберга-Маркуардта, время обучения составило ≈ 3 минуты. Следует отметить, что несколько лучшие результаты были достигнуты при использовании более сложных МСП: 2-20-20-1, 2-15-10-1. Однако для их обучения потребовалось значительно больше времени – 6 и 5 минут соответственно. Упрощение этих структур путем удаления наименее значащих весов и привело к получению окончательной персептронной модели 2-7-5-1, точность которой оказалась вполне приемлемой.

На рис. 6.26 показано изменение реальных и модельных выходных переменных этого же отделения диффузии (один выходной параметр).

График изменения выходной величины модели диффузионного аппарата после управления по алгоритму (6.11) приведен на рис. 6.27

Процесс стабилизации температур сокоструженой смеси о всех зонах диффузионной установки с помощью разработанных нейроэволюционных алгоритмов приведен на рис.6.28. Здесь темными точками показано реальное изменение выходных переменных при существующем управлении, светлыми – при нейросетевом адаптивном. На рисунках не показан переходный режим (режим идентификации), который составил для применяемого метода примерно 30 тактов.

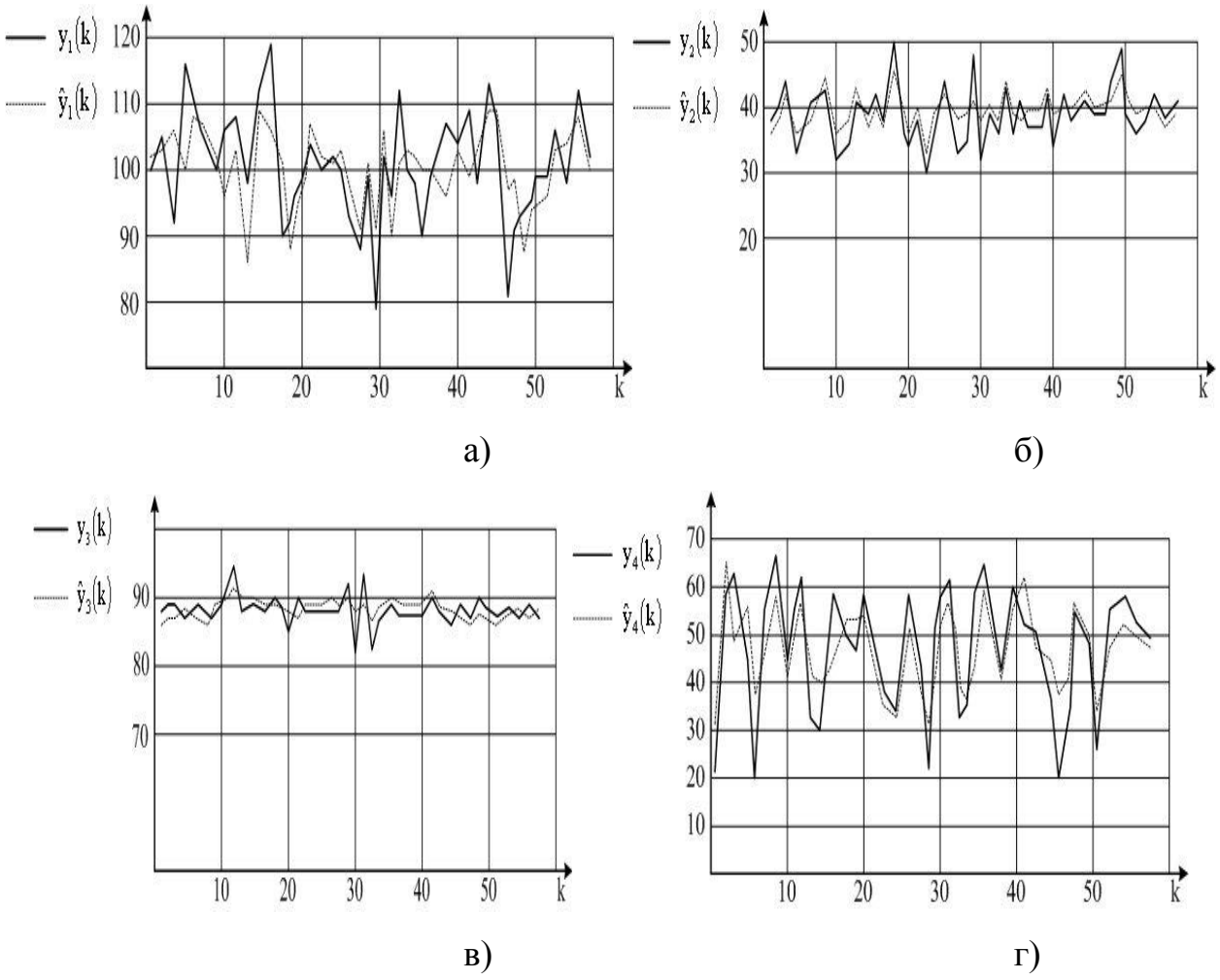


Рисунок 6.25 – Изменение реальных и модельных выходных переменных отделения диффузии

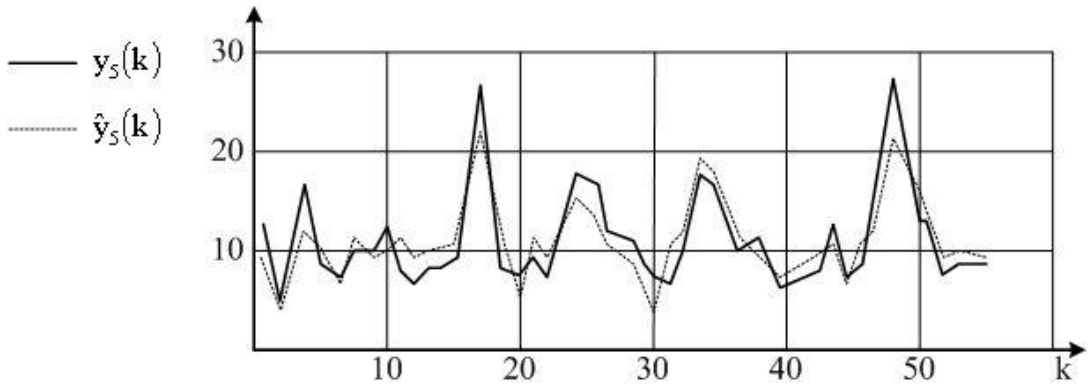


Рисунок 6.26 – Изменение реальных и модельных выходных переменных отделения диффузии для модели с одним выходом

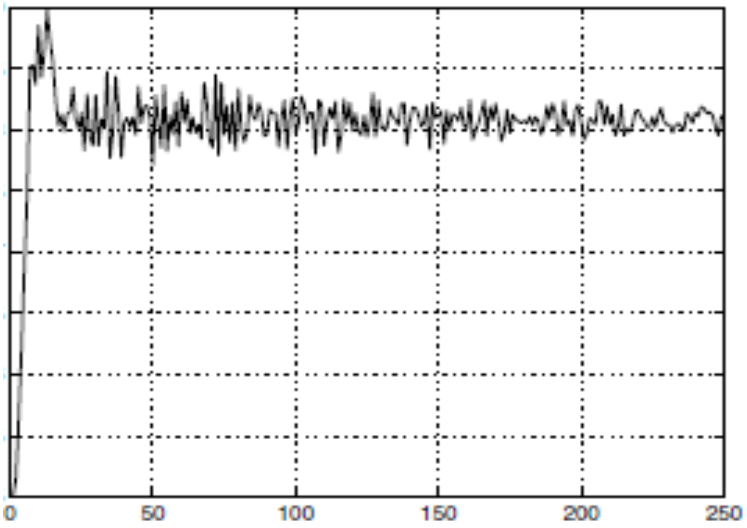
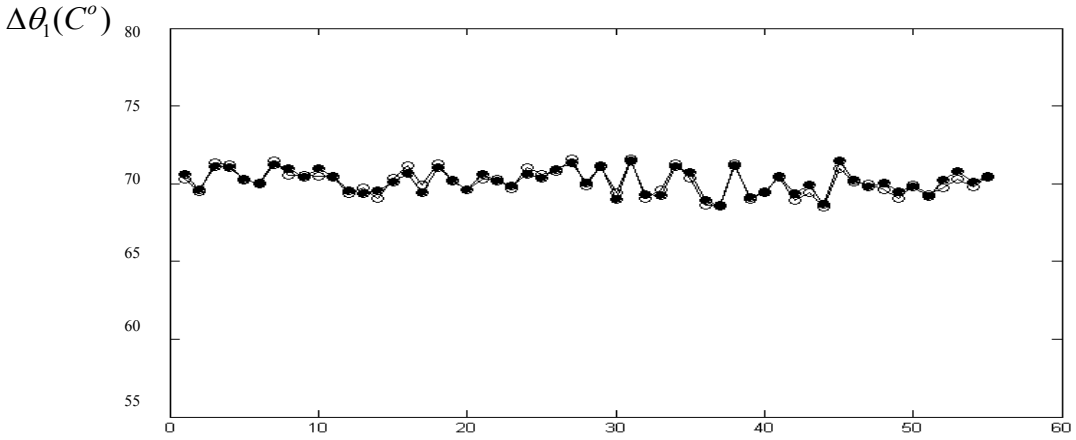
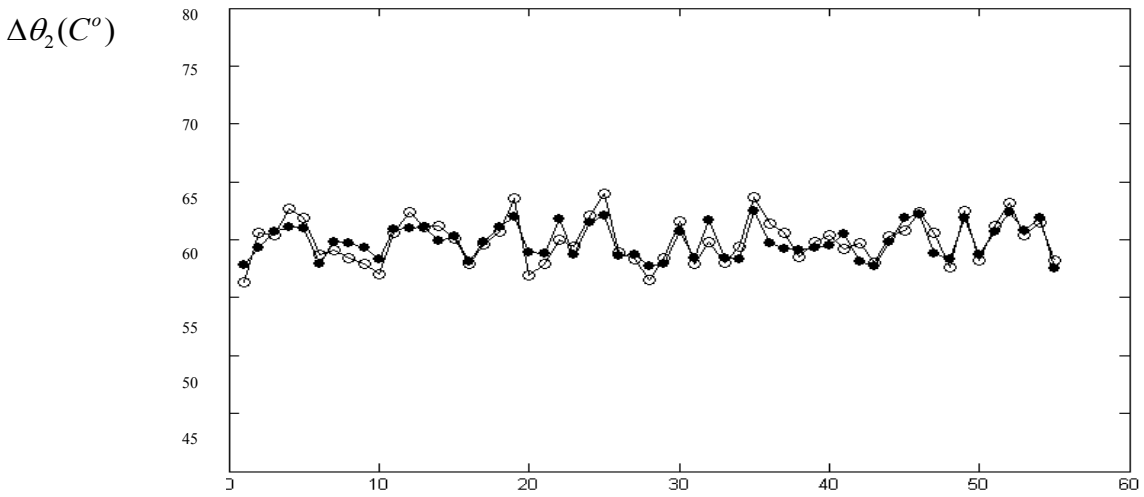


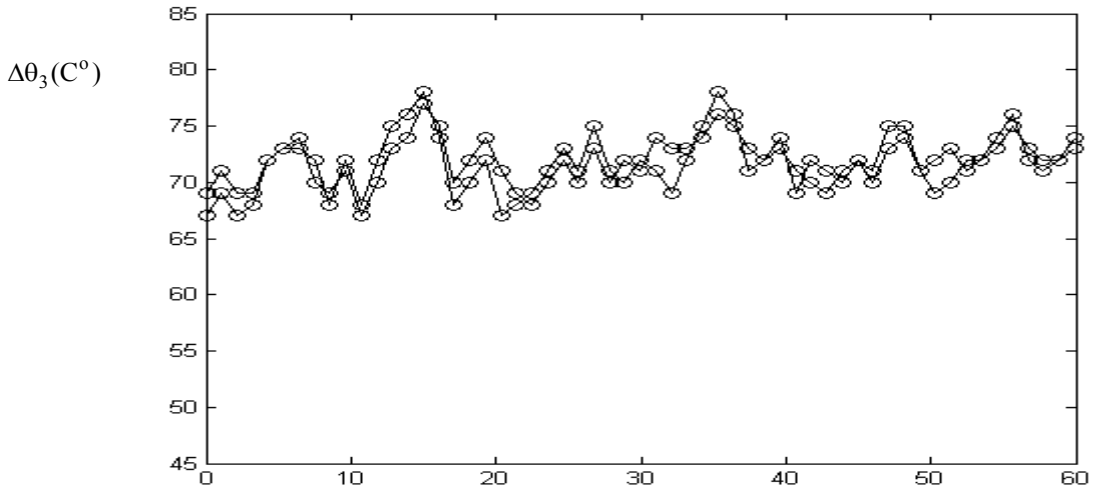
Рисунок 6.27 – График изменения выходной величины модели после управления



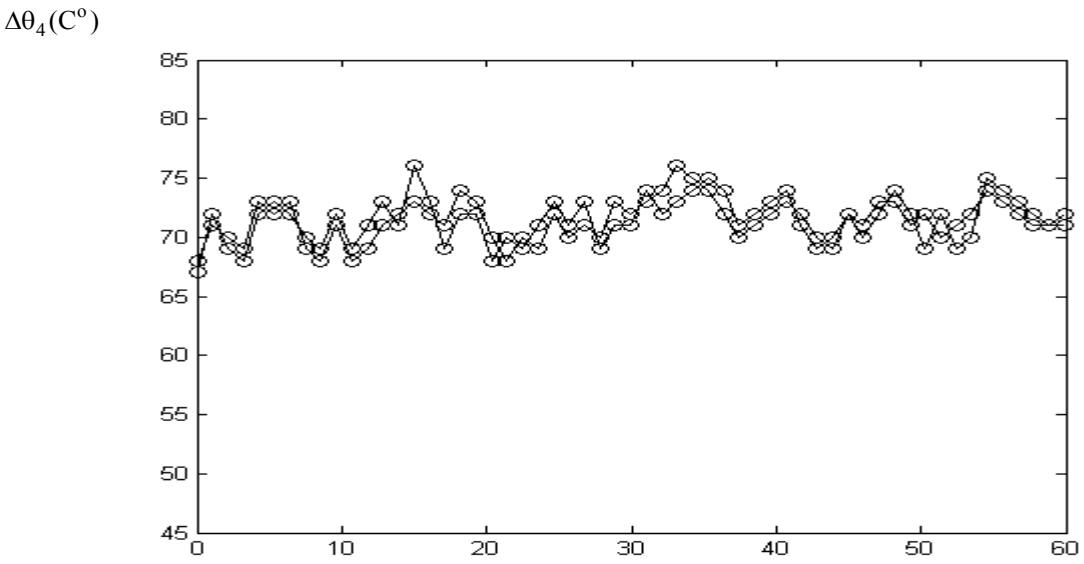
а)



б)



в)



г)

Рисунок 6.28 – Изменение температур сокоструженой смеси в зонах диффузионной установки

6.6 Нейросетевое управление длиной петли в травильных ваннах

Схема процесса травления в одной из травильных ванн непрерывного травильного алгоритма (НТА) приведена на рисунке 6.29. Отклонения длины петли от заданных технологическим регламентом значений (в частности, высокочастотные колебания длины) снижают качество травления и могут

вызвать механические деформации, а также незапланированный перетрав или недотрав стальной полосы. Основной задачей управления является сохранение постоянной длины петли между механизмами вращения (протяжки) (1) и (2). Механизму вращения (2) задается постоянная скорость вращения, определяемая производительностью линии. Механизм (1) должен обеспечивать такую скорость вращения, чтобы длина петли оставалась постоянной. Длина петли измеряется оптическим датчиком, сигнал которого используется для коррекции скорости механизма (1).

Сбой в работе блока управления механизмом (1) приводит к изменению его скорости вращения (и, соответственно, к изменению длины петли), вследствие чего длина петли может превысить допустимые значения (максимальное или минимальное), что вызовет остановку технологической линии. Динамичные изменения длины петли (в частности, высокочастотные колебания) снижают качество травления и могут вызвать механические деформации ленты проката. Очевидно, что рассматриваемый объект управления может быть отнесен к классу критических. На рисунке 6.30 приведена схема управления длиной петли в травильной ванне с применением двух регуляторов. Такое управление осуществляется здесь путем изменения скорости вращения V_{s1} в соответствии с заданной скоростью V_{s2}.

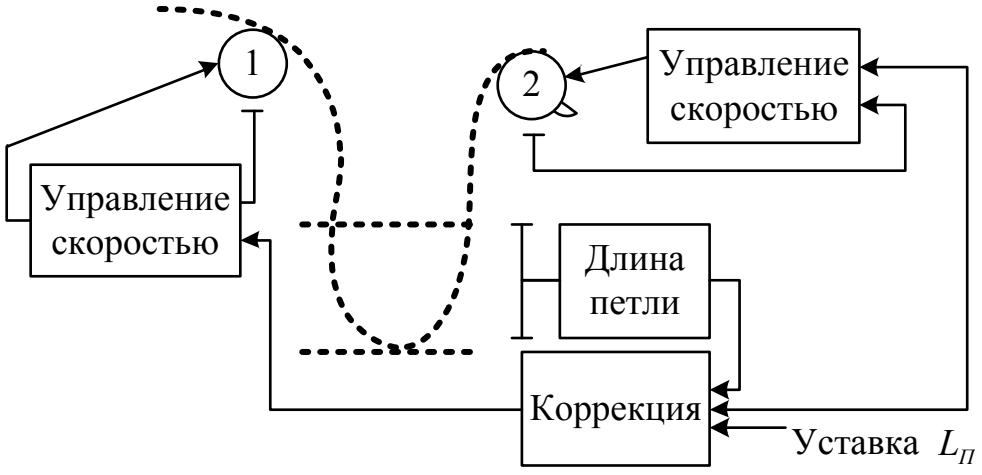


Рисунок 6.29 – Принцип управления длиной петли в травильной ванне

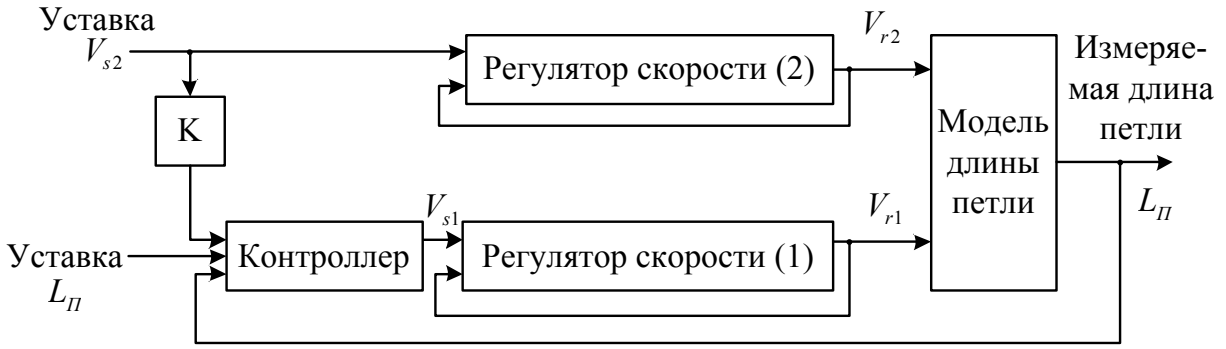


Рисунок 6.30 – Схема управления длиной петли в травильной ванне

Выходными переменными регуляторов скорости Рег1 и Рег2 являются, соответственно, измеряемые скорости вращения V_{r1} и V_{r2}, а выходной переменной всей замкнутой системы управления является измеряемая длина петли $L_П$.

Анализ реальных данных НТА показывает, что ограниченные возможности управления скоростью вращения механизма (1) не позволяют поддерживать требуемую производительность линии. Эта проблема может быть решена путем усовершенствования системы управления механизмом (1). Для этого целесообразно осуществить моделирование процесса управления длиной петли с помощью искусственной нейронной сети (ИНС), обучаемой на реальных данных техпроцесса. Особый интерес представляет построение динамической модели, отражающей зависимость между измеряемой длиной петли, реальными скоростями и задающими воздействиями для регуляторов скорости механизмов вращения (1) и (2). Рассмотрим следующую структуру модели:

$$L_П(t) = F_{ИНС}\left[L_П(t-1), L_П(t-2), V_{r1}(t-1), V_{r1}(t-2), V_{r2}(t-1), V_{r2}(t-2), V_{s1}(t), V_{s2}(t)\right], \quad (6.21)$$

где $F_{ИНС}$ – структура ИНС, полученная в процессе обучения по данным для нормальных режимов работы технологической линии [317].

Ошибка работы системы управления длиной петли определяется разностью:

$$\varepsilon(t) = L_{П}(t) - L_{П}(t), \qquad (6.22)$$

где $L_{П}(t)$ и $L_{П}(t)$ – текущие значения реальной и заданной длины петли.

В нормальном режиме $\varepsilon(t) \in D_{норм}$, $D_{норм}$ – допустимая область, которая обеспечивает оптимальное качество проката; $\varepsilon(t)$ – нормально распределенная величина, характеризуемая средним значением $\bar{\varepsilon}$ и дисперсией σ.

Нештатная ситуация характеризуется попаданием величины ε(t) в недопустимый диапазон $D_{сб.}$ - $\varepsilon(t) \in D_{норм}$. Переход процесса в такой режим вызывает соответствующие структурные и параметрические изменения рассматриваемой модели, т.е.:

$$\Delta F_{ИНС} = F_{ИНС}^{сб.} - F_{ИНС},$$

где $F_{ИНС}^{сб.}$ – модель для нештатных (сбойных) ситуаций.

Если $\lim\limits_{t \to 0} \Delta F_{ИНС} \to 0$, то ошибка является устранимой в установившемся состоянии; если $\lim\limits_{t \to 0} \Delta F_{ИНС} \to \Delta F_{ИНС}^{0}$, то ошибка является статической.

Изменения $\Delta F_{ИНС}$ вызывают эквивалентные изменения ошибки управления длиной петли, что оказывает соответствующее влияние на качество проката.

Недостаточная адекватность существующих моделей реальным процессам, нелинейность и нестационарность их характеристик,

обуславливает применение нейросетевого подхода для моделирования и управления НТА.

В общем случае процесс травления может быть описан нелинейной ARMA-моделью (NARMA) [318-319], аппроксимируемой ИНС и обучаемой с помощью алгоритма обратного распространения ошибки так, как это показано на рис.6.31.

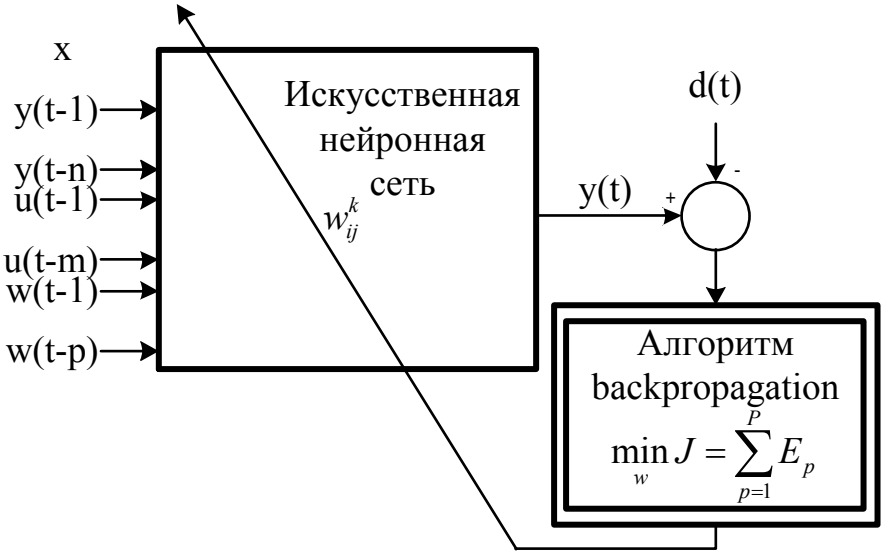


Рисунок 6.31 – Обучение ЭИНС с использованием BP-алгоритма

При моделировании принималось: размерности вектора входа и вектора выхода – 3 и 1 соответственно; количество слоев – 2; количество нейронов в первом и выходном слоях – 28 и 1 соответственно; импульс – $\alpha = 0.65$; скорость обучения – $\eta = 0.75$.

На рисунке 6.32 показаны входные и выходные данные, использованные для обучения ИНС. Рисунок 6.33 подтверждает сходимость процесса обучения нейросетевой модели.

Для решения задачи управления длиной петли стальной полосы в травильных ваннах использовался также нейросетевой подход.

На рисунке 6.34-а показана динамика измеренных и расчетных выходов, а на рисунке 6.34-б – динамика остатков.

По модели процесса и новым входным данным для разных состояний процесса были получены графики изменения остатков (рисунок 6.34).

Качество проката оценивалось по значению индекса качества Q, заданного как функция от среднего значения и дисперсии остатка.

Для обучения ИНС были использованы данные, приведенные на рисунке 6.35.

Полученная нейросетевая модель имела следующие характеристики: размерности вектора входа и вектора выхода – 4 и 1 соответственно; количество слоев – 2; количество нейронов в первом и выходном слоях – 26 и 1 соответственно; импульс – $\alpha = 0.85$; скорость обучения – $\eta = 0.75$.

На рисунке 6.36 приведены результаты нейросетевого моделирования. ИНС обучалась по данным, полученным в нормальном режиме работы технологической линии. Статистические свойства остатка для длины петли, т.е. $\bar{x}_n$ и σ_n, коррелированы с индексом качества Q_n. Это подтверждается динамикой обучения, показанной на рисунке 6.36, б.

Очевидно, что сбои оборудования приводят к прямым или косвенным потерям качества продукции. В общем случае, технологический процесс осуществляется с применением целого ряда технологических блоков. Качество конечной продукции зависит от текущего состояния этих блоков.

Общее состояние линии или состояние каждого блока j можно определить по значению остатка $\varepsilon_j(t)$ и последствиям изменения качества на j-м шаге техпроцесса $q_j(t)$.

Так как глобальный индекс качества $Q(t)$ является сложной функцией от $q_j(t)$, рассматривалось лишь часть техпроцесса травления листовой стали, связанная с возможностью возникновения сбоев, определяемых остатком $\varepsilon(t)$, и вызывающих опасность механической деформации перетрава и недотрава стальной полосы. Тестирование оценок качества осуществлялось по новым наборам данных. Результаты тестирования для четырех режимов приведены на рисунке 6.37.

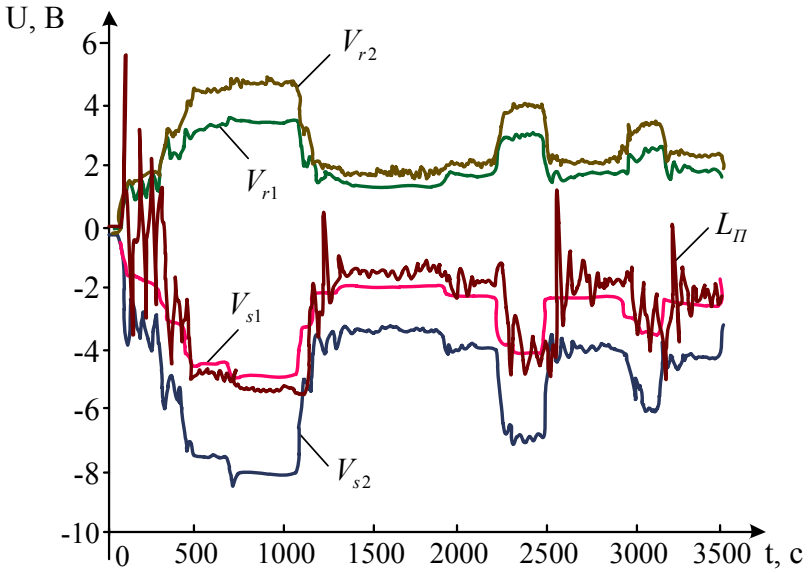


Рисунок 6.32 – Данные по входным и выходным переменным

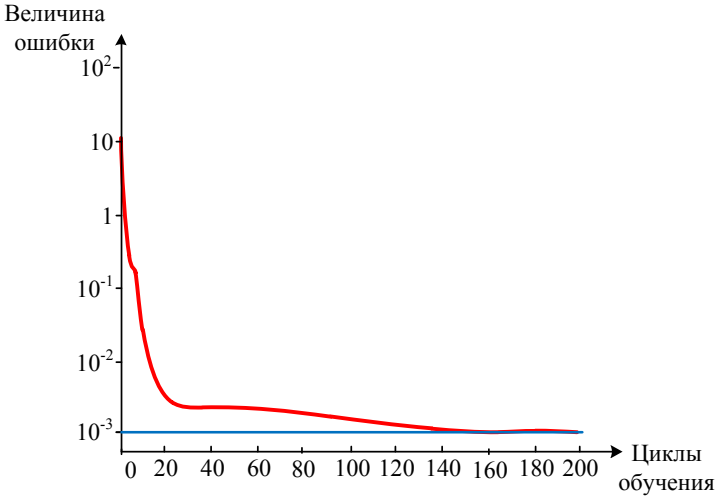
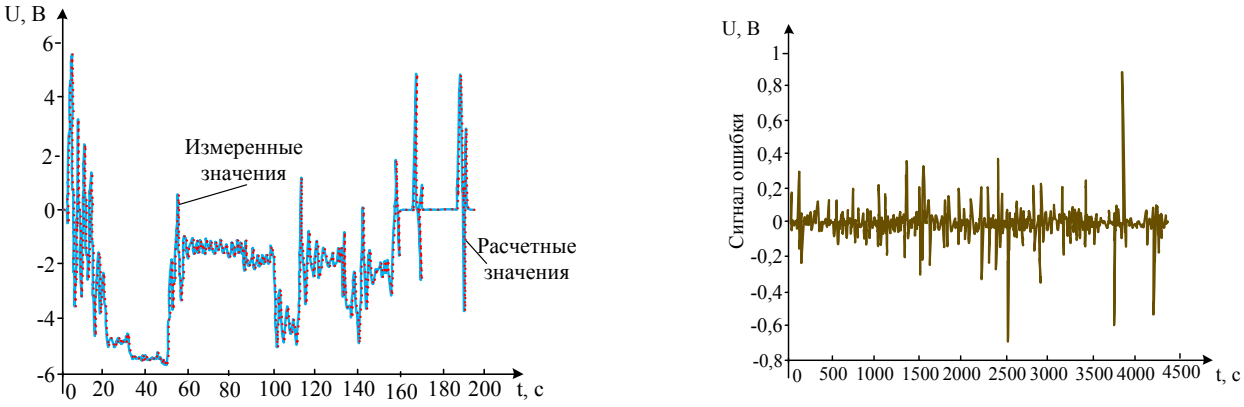


Рисунок 6.33 – Сходимость процесса обучения



а) б)

Рисунок 6.34 – Обучение нейросетевой модели: расчетные и измеренные выходы – а); динамика остатков – б)

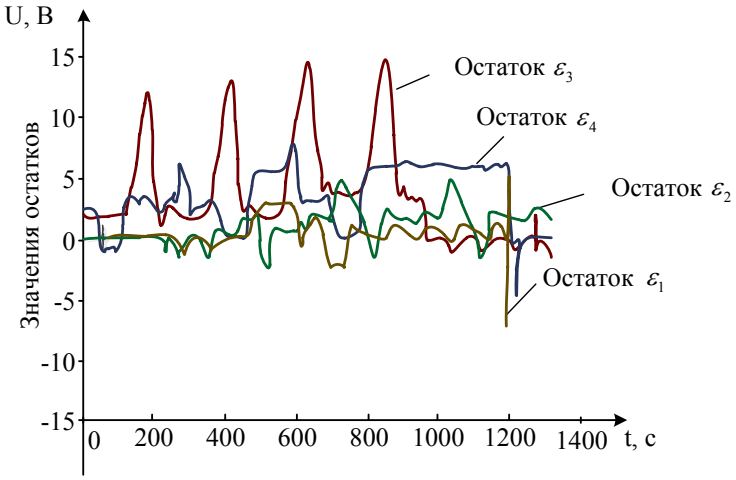
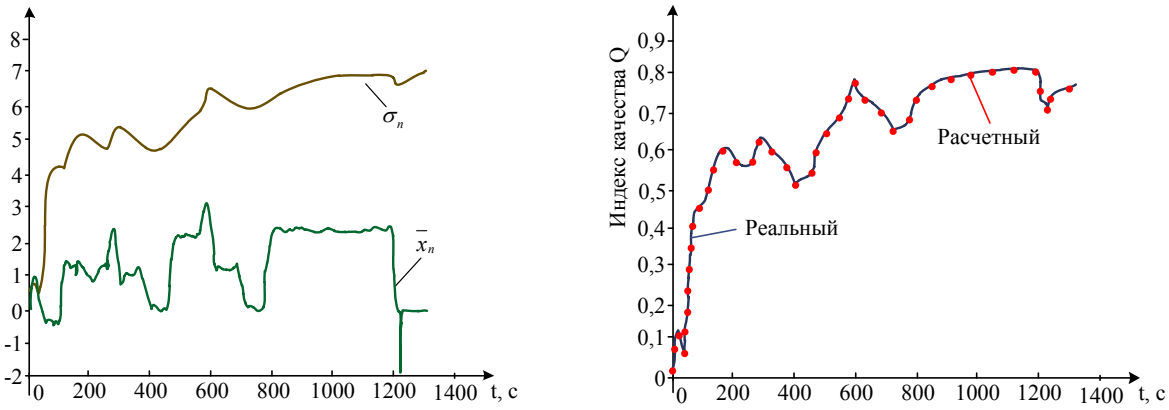
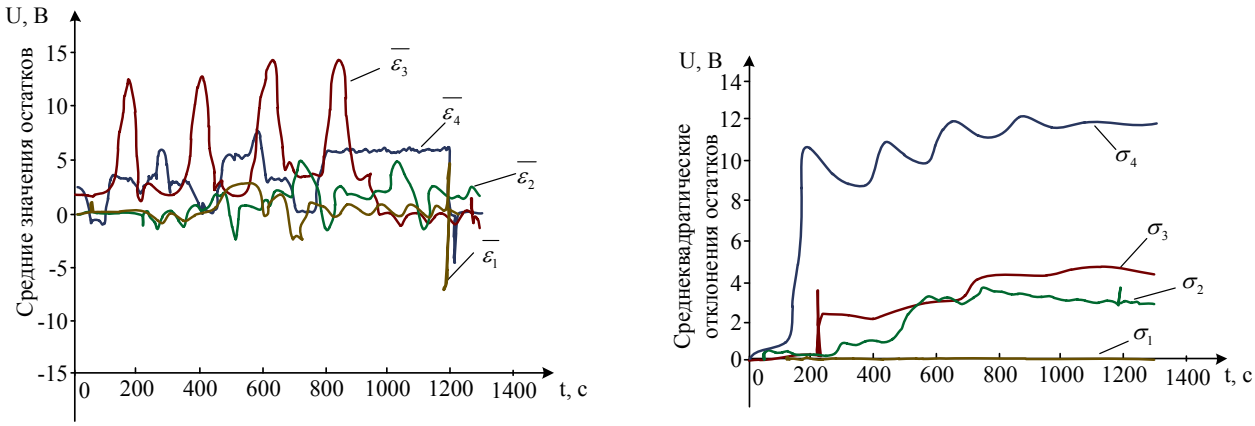


Рисунок 6.35 – Состояние остатков

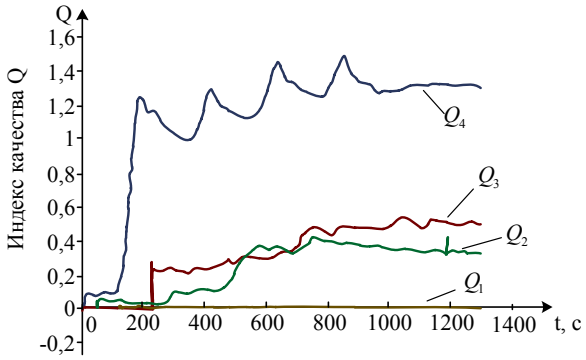


а) б)

Рисунок 6.36 – Индекс качества Q: адаптация средних значений и среднеквадратичного отклонения остатков – а); реальный и расчетный индексы качества Q_n – б)



а) б)



в)

Рисунок 6.37 – Оценка качества продукции: адаптация средних значений $\bar{\varepsilon}_n$ остатков – а); адаптация среднеквадратичного отклонения σ_n остатков – б); расчетные и измеренные значения индекса качества Q_n

Очевидно, что индекс качества зависит от колебания листа, определяемых статистическими свойствами остатков (например, $\bar{\varepsilon}_n$ и σ_n). Индекс качества Q_n является индикатором, который классифицирует локальный дефект качества в рабочем окне *n*. Последствия возникновения сбоев оборудования и их влияние на качество продукции были оценены путем анализа исходных данных и результатов моделирования.

6.7 Использование нейросетевого подхода для контроля распределения электрической мощности в процессе плавки

В основе разработанной системы автоматизированного контроля распределения электрической мощности, подводимой к ванной расплавов в процессе плавки, действующая на руднотермических печах Побужского ферроникелевого комбината лежит нейросетевой подход.

Это объясняется тем, что исследуемые технологические процессы являются нестационарными и нелинейными и поэтому трудноописываемыми и анализируемыми традиционными методами.

Ядро нейропроцессора в рассматриваемой автоматизированной системе на функциональной схеме обозначено как «Специализированный процессор быстрой обработки больших информационных массивов».

Структурная и функциональная схемы системы автоматизированного измерения распределения электрической мощности, подводимой в ванной расплавов в процессе плавки (АВРПП) приведены на рис. 6.38, 6.39. Принцип работы системы АВРПП рассмотрен в Приложении К.

Эксплуатация систем дала возможность на базе большого количества накопленного статистического материала разработать практические методы и компьютерные программы быстрой обработки получаемых при измерениях данных на предмет исследования корреляционных связей между различными параметрами, характеризующими процесс плавки: уровнями ванны расплава шлака и суточными количествами выплавляемого металла, уровнями ванны расплава шлака и количеством огарка, что проплавляется, проводимостью расплава шлака и суточным количеством металла, наплавляемый и др.

Статистическое сопоставление формы графиков распределения потенциалов по глубине ванны расплавов с количественными показателями работы печей: суточным количеством наплавленного металла, суточным количеством проплавленного огарка, суточным потреблением электроэнергии позволило выявить, что временная равномерность потребления электрической мощности (без учета ее комплексного распределения в объеме расплавов) , неадекватная временной равномерности плавки огарка и выплавке металла вследствие того, что часть электрической энергии, и, зачастую значительная, расходуется не на плавку огарка, а непродуктивно: на перегрев «тела» электродов, металла, футеровки и подины печи, на лишнее восстановления кремния.

Получено стали количественные зависимости между графиками распределения проводимостей, получаемые системой и производительностью печей.

Получены конкретные величины значений уровня расплава шлаков, его проводимости и других параметров плавки, при которых по одинаковой мощности, подводимой существенно больше проплавляется огарка и выплавляется металла.

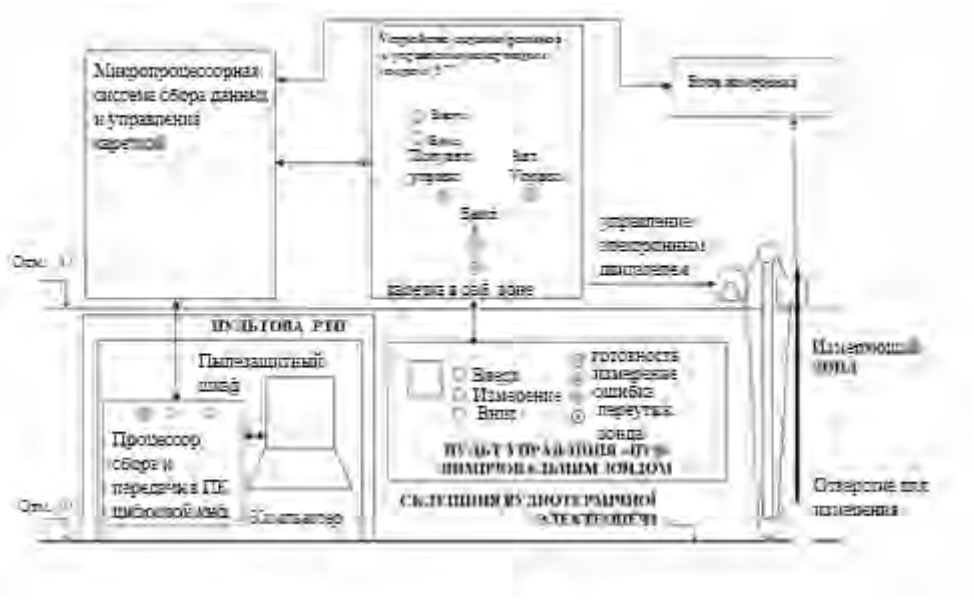


Рисунок 6.38 – Структура электронно-механического комплекса контроля распределения электрической мощности, подводимой в ванной расплавов руднотермической электропечи

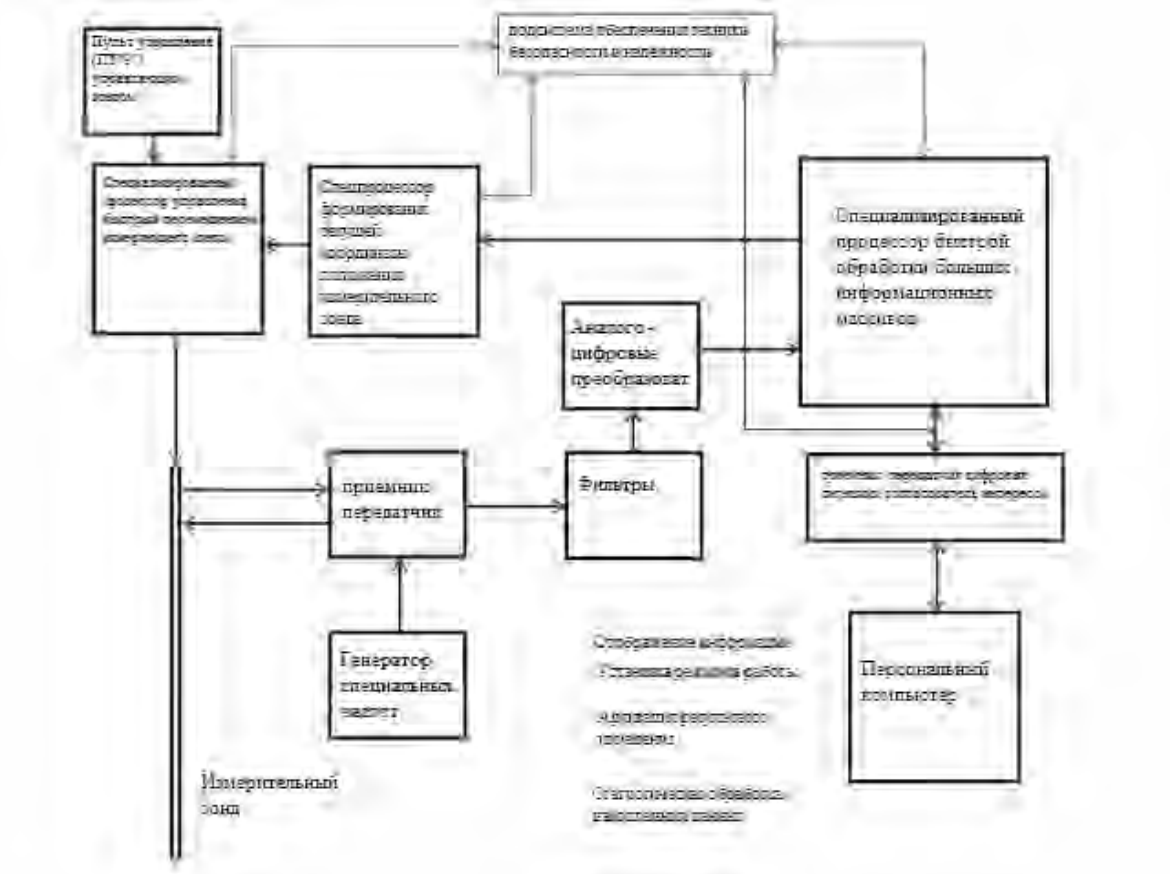


Рисунок 6.39 – Функциональная схема автоматизированной системы распределения электрической подведенной мощности, в ванной расплавов в процессе плавления

Разработана методика определения для конкретных печей (с установленной на них системой) оптимального диапазона допустимых пределов изменения графика распределения потенциалов подводимой мощности, по глубине ванны расплавов, при котором выплавляется максимальна в единицу времени количество металла при заданном расходе электроэнергии.

Статистическое сравнение видов получаемых графиков и поломок электродов позволило определить технологические ситуации, в результате которых они возникают и которые необходимо по графику контролировать и избегать.

Например, были выявлены причины поломок электродов вследствие влияния на их точечные участки, которые находятся в зоне огарка, тока высокой плотности, «закорачивающей» электроды из сегрегированных с высокой электропроводностью антрацитовый восстановитель. С целью предотвращения поломок электродов из-за действия этого фактора была скорректирована технология подготовки огарка. Были приняты меры по улучшению качества перемешивания антрацита с рудой, не предполагая появления высокопроводящие сегрегированных антрацитовой фракции.

На рисунках 6.40-6.41 показаны примеры графиков зависимости распределения потенциалов электрической подведенной мощности от глубины погружения измерительного зонда.

Визуализация реального распределения в ванной расплавов руднотермической электропечи электрической мощности, подводимой для плавки, дает возможность операторам печей поддерживать режим, соответствующий оптимальной плавники.

На рисунке 6.40 приведен график с пояснительными надписями, помогают оценивать полученную информацию.

Красная линия - это измеренная напряжение при перемещении зонда вниз, а синяя линия - это измеренная напряжение при перемещении зонда вверх.

Проведенные измерения показывают, что при безаварийному режиме работы печи напряжение в расплаве шлака колеблется в пределах 30-70 В.

При этом характерно такое поведение графиков: при перемещении зонда вниз наблюдается напряжение 0 В до момента касания зонда расплава шлака. В момент касания шлака напряжение на зонде резко меняется до 30-70 В и держится примерно постоянной к погружению в расплав металла.

Однако касания расплава металла при перемещении вниз зонд не воспринимает мгновенно, очевидно, потому, что при погружении в шлак зонд покрывается слоем застывшего на нем шлака (большая разница температур тела зонда и расплава шлака). Помещенный в расплав металла конец зонда постепенно освобождается от застывшего шлака, отображается на графиках резким изменением напряжения на красной линии в нижней части графика. Это изменение напряжения может произойти позже (не у перемещении зонда, а во время его недвижимости в нижней точке измерения). Этим процессом освобождения конца зонда от шлака определяется время пребывания зонда в расплаве металла.

На рисунках 6.42, 6.43 приведены графики распределения потенциалов при аварийных режимах работы печи. Очень высокий уровень шихты в пределах 2400-3100 мм и низкий уровень расплава шлака 1100-1300 мм. При этом хорошо видно, что в шихте высокое напряжение 90-150 В. А напряжение в расплаве шлака низкая, 10-40 В, было нехарактерно для безаварийного режима работы печи.

В этом режиме работы печи измерения сопровождались неожиданными газовыми выбросами при прохождении зондом шихты и верхней части расплава шлака, происходило покрытия поверхности части измерительного зонда, находящегося в подсводовом пространстве, белесым налетом Si.

На графиках видно, что шихта неоднородна по своему составу: в некоторых местах она имеет высокую проводимость (наличие большого количества восстановителя), а в других - остается непроводящих (резкие изменения напряжения в области шихты от максимального до 0 В).

Рисунок 6.40 – График напряжений с пояснительными надписями

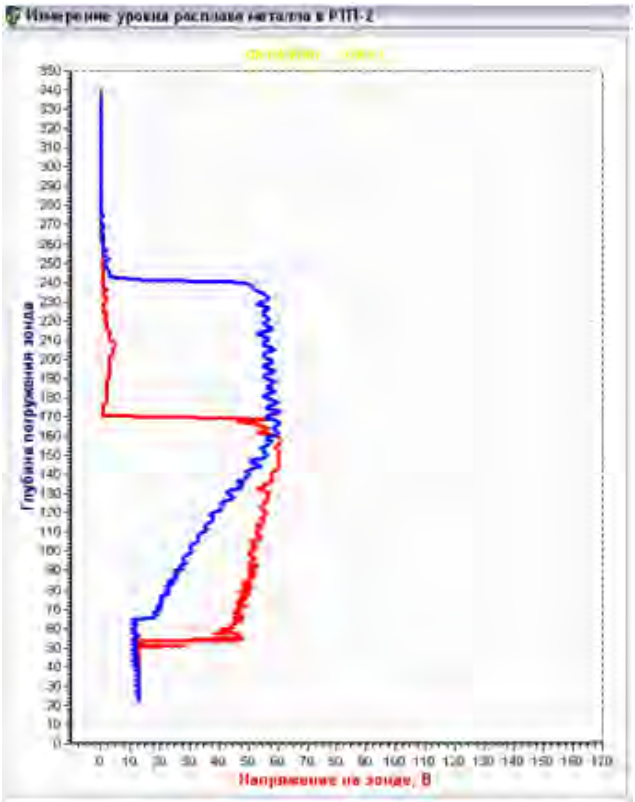


Рисунок 6.41 – Пример измерения при нормальных условиях работы печи

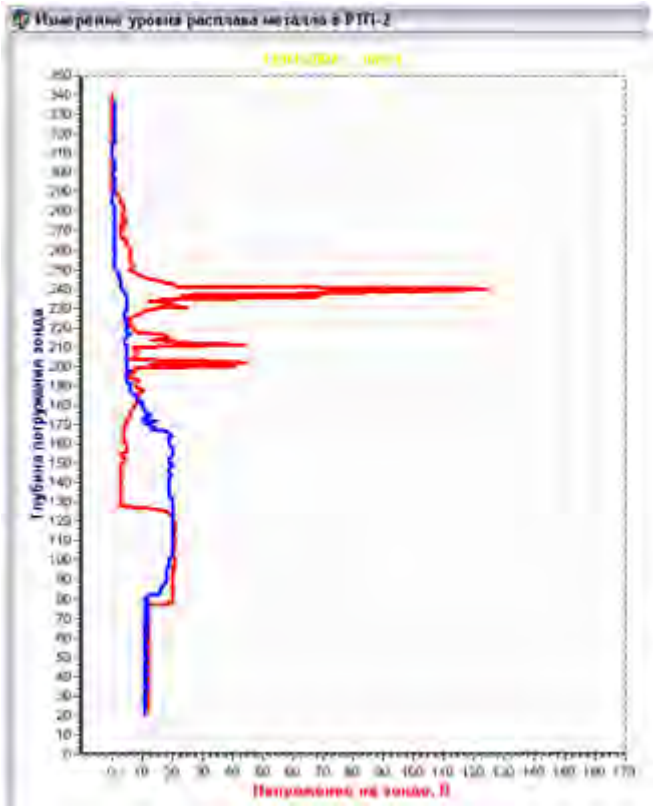


Рисунок 6.42 – Аварийный режим работы печи - высокое напряжение в шихте, низкая - в расплаве шлака

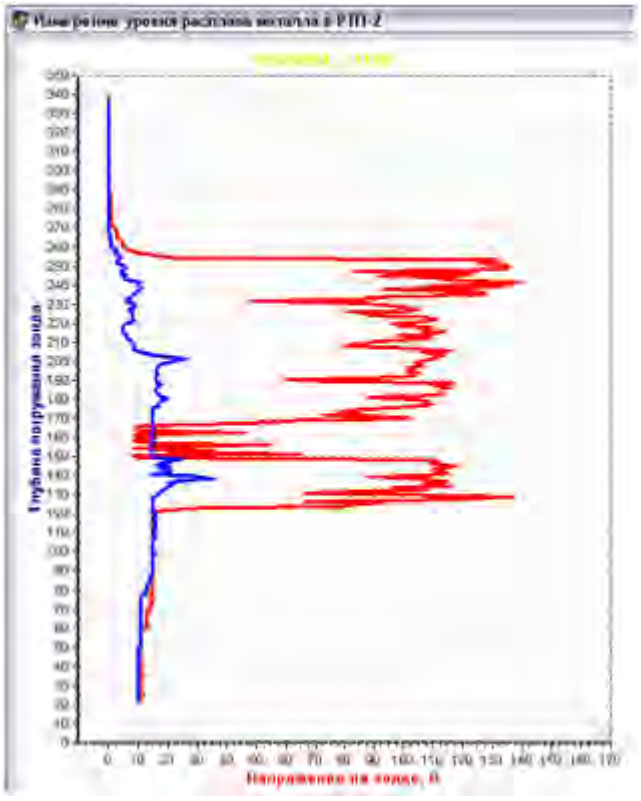


Рисунок 6.43 – Аварийный режим работы печи - слишком высокое напряжение в шихте

6.8 Использование эволюционирующих нейронных сетей в самообучающихся интеллектуальных системах прогнозирования

Интеллектуальные информационные системы (ИИС) основаны на концепции использования базы знаний для генерации алгоритмов решения прикладных задач различных классов в зависимости от конкретных информационных потребностей пользователей [320-323].

Самообучающиеся интеллектуальные информационные системы основаны на методах автоматической классификации ситуаций из реальной практики или на методах обучения на примерах реальных ситуаций, образующих обучающую выборку. Элементы обучающей выборки описываются множеством классификационных признаков. При «обучении с учителем» специалист (учитель) задает для каждого примера значения признаков, определяющих его принадлежность к некоторому классу ситуаций. При обучении «без учителя» система должна самостоятельно выделять классы ситуаций по степени близости значений классификационных признаков.

Среди самообучающихся ИИС [321], включающих индуктивные системы, искусственные нейронные сети (ИНС), системы, основанные на прецедентах и информационные хранилища, существенный интерес представляют ИНС.

Разрабатываемая самообучающаяся интеллектуальная информационная система [324] предназначена для решения задач прогнозирования важнейших технико-экономических показателей деятельности предприятия. Трудности прогнозирования временных рядов, порожденных экономическими процессами, связаны в первую очередь с их нестационарностью, обусловленной изменением с течением времени не только характеристик исследуемого процесса прогнозирования, но и их состава и взаимосвязей. Последнее приводит к необходимости изменения структуры используемой математической модели объекта прогнозирования и пересчету всех ее

параметров. Трудности получения качественного и точного прогноза существенно возрастают, если рассматривается задача краткосрочного прогнозирования.

Среди новых методов и алгоритмов, делающих возможным получение адекватных прогнозов, одними из наиболее перспективных представляются нейросетевые, которые могут служить основой создания ИИС прогнозирования временных рядов, требующих при своем функционировании минимального участия человека. Считается, что эти методы позволят увеличить глубину прогноза за счет выявления скрытых закономерностей взаимосвязей исследуемого процесса.

На первоначальном этапе создания ИИС прогнозирования временных рядов решалась задача прогнозирования энергопотребления деревообрабатывающим предприятием Германии с помощью эволюционирующих нейронных сетей [324]. Для построения модели процесса энергопотребления использовалась статистическая выборка за последние 3 года. Результаты прогнозирования на полгода вперед представлены на рис.6.44, а результаты прогнозирования на текущие сутки показаны на рис.6.45. График средней ошибки предсказания показан на рис.6.46. Как следует из результатов моделирования, применение ЭИНС в задачах прогнозирования нестационарных временных рядов является весьма эффективным.

Выводы по разделу 6

1. Имитационное моделирование является необходимым этапом исследования эффективности как выбранной архитектуры ИНС, так и применяемого алгоритма обучения сети. В настоящее время наиболее массовым направлением нейрокомпьютинга является моделирование нейронных сетей на персональных компьютерах. Успехи в моделировании позволяют создавать обучающие программы, дающие пользователю полное представление об ИНС и их

возможностях. Анализ особенностей наиболее популярных и широко используемых в настоящее время программных нейросимуляторов позволил выбрать для исследования ЭИНС прямого распространения расширение пакета *NeurophStudio* и все исследования проводились в среде *NeurophStudio*.

2. Проведено имитационное моделирование разработаных архитектур и методов обучения эволюционирующих ИНС. Показаны их преимущества перед известными нейросетевыми системами обработки информации и методами их обучения по точности, устойчивости и быстродействию в задачах идентификации одно- и многомерных нелинейных нестационарных динамических объектов при наличии различного рода помех в условиях априорной и текущей неопределенности.

3. Разработаны и исследованы новые законы адаптивного прогнозирующего нейро-управления нелинейными нестационарными динамическими объектами, функционирующими в условиях неопределенности. Для коррекции эталонной траектории предложено использовать эволюционный подход. Полученные законы управления не используют градиентные алгоритмы и характеризуются повышенным быстродействием. Дополнительное преимущество обеспечивается использованием результатов работы контура, в котором оцениваются параметры помехи.

4. Разработаны и исследованы новые методы нейросетевого сжатия изображений, основанные на модификации сетей Кохонена, «нейро-газ» и РБФ.

5. Решены практические задачи измерения температурных полей вращающихся трубчатых печей в ООО «Побужзкий ферроникелевый комбинат»; управления процессом травления полосовой стали на металлургических предприятиях Украины (АТ «Співдружність-Т»); разработки АСУ ТП диффузии, дефекосатурации, выпаривания и кристаллизации в ООО «Кириковский сахарный завод», прогнозирования энергопотребления.

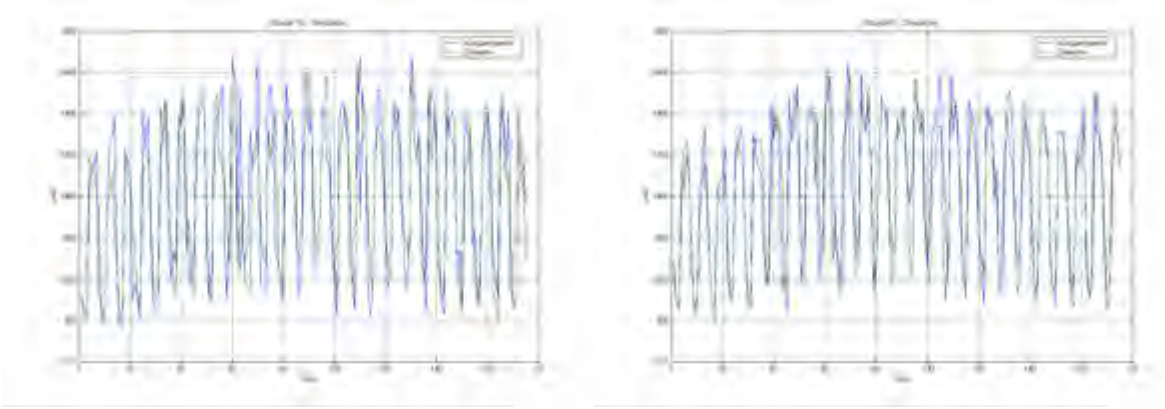


Рисунок 6.44 – Результаты прогнозирования энергопотребления на полгода вперед

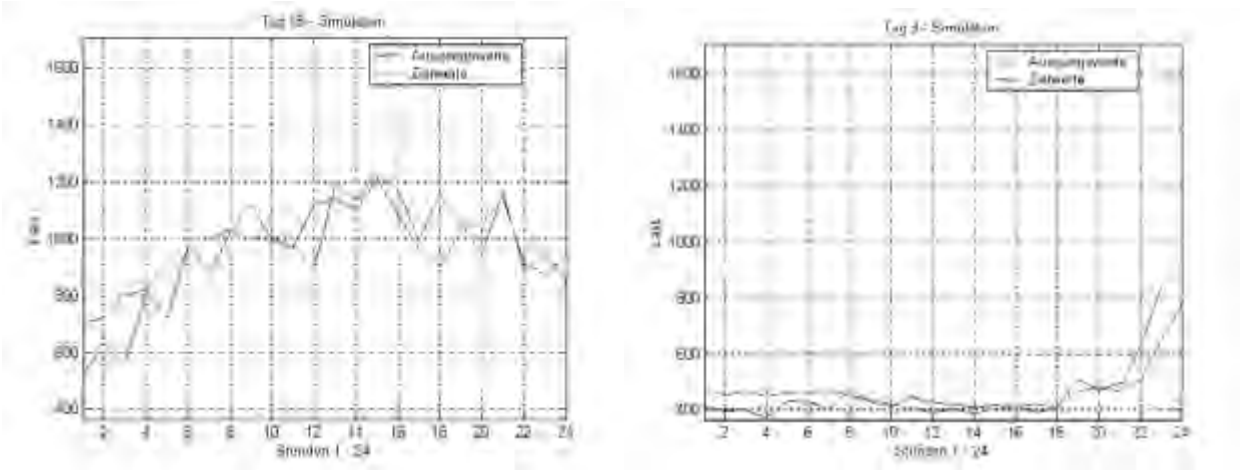


Рисунок 6.45 – Результаты прогнозирования энергопотребления на текущие сутки

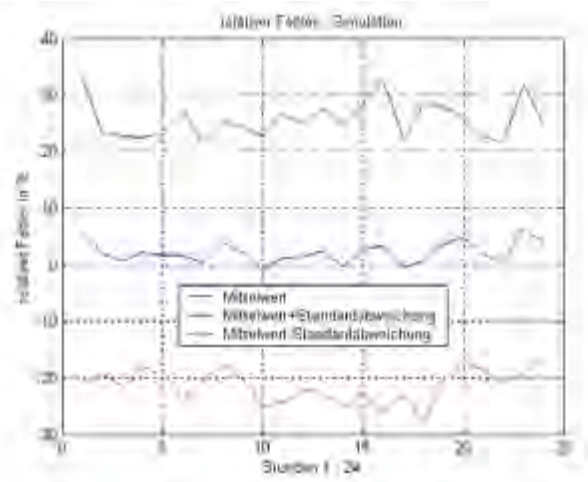


Рисунок 6.46 – Средняя ошибка предсказания

ОБЩИЕ ВЫВОДЫ

В диссертационной работе, посвященной решению важной научно-практической проблемы развития теоретических основ интеллектуального анализа данных и созданию новых эволюционирующих ИНС прямого распространения с целью повышения эффективности обработки информации в условиях априорной и текущей неопределенности, которые имеют важное научное и практическое значение для создания эффективных интеллектуальных систем обработки информации, эмуляции, управления, прогнозирования, идентификации как функционирующих объектов, так и вновь создаваемых

1. Проведен анализ современного состояния проблемы интеллектуальной обработки информации, который показал, что в качестве базиса для синтеза подобных систем обработки целесообразно использовать статические ИНС прямого распространения, а также обоснована необходимость применения для улучшения их свойств методов теории адаптации и эволюции.

2. Разработан новый метод робастной многокритериальной оптимизации (Парето-оптимизации) на основе робастных фитнес-функций и информационных критериев оценки сложности модели, который позволяет определить оптимальную нейросетевую модель при наличии негауссовских помех.

3. Предложены новые методы обучения ИНС, обеспечивающие требуемую точность обучения при наличии ограниченных помех. Данные методы характеризуются вычислительной простотой и повышенным быстродействием за счет использования дополнительных процедур адаптивной коррекции их параметров.

4. Предложена новая процедура М-обучения вейвлет нейросетей первого и второго порядка, в которой для выбора структуры сети применен робастный информационный критерий для оценки параметров – робастный алгоритм Гаусса-Ньютона. Использованные в данной процедуре вместо

вторых производных функционалов значения весовых функций позволяют повысить ее устойчивость.

5. Разработаны новые робастные методы обучения ИНС, дающие возможность эффективно обрабатывать информацию при наличии помех с асиметричными распределениями. Данные методы используют дополнительные процедуры оценивания параметров функционалов и помех, что позволяет осуществлять коррекцию получаемых оценок и устранять их смещение.

6. Разработаны новые процедуры коррекции параметров используемых при обучении функционалов и оценивания параметров помехи, описываемой моделью Тьюки-Хьюбера, что позволяет при отсутствии априорной информации о статистических свойствах помех адаптивно корректировать параметры ИНС.

7. Разработаны новые законы адаптивного прогнозирующего нейро-управления нелинейными нестационарными динамическими объектами, функционирующими в условиях неопределенности. Для коррекции эталонной траектории предложено использовать эволюционный подход. Полученные законы управления не используют градиентные алгоритмы и характеризуются повышенным быстродействием. Дополнительное преимущество обеспечивается использованием результатов работы контура, в котором оцениваются параметры помехи.

8. Предложены новые методы аппроксимации гауссовских базисных функций в РБС нулевого и первого порядков. Использование кусочно-линейной аппроксимации позволяет существенно упростить вычисления, сопутствующие процессам построения нейросетевой модели исследуемых объектов и управления ими.

9. Получил дальнейшее развитие метод гибридного обучения ИНС. Модификация метода состоит в использовании эволюционного алгоритма для грубой настройки параметров сети и разработанной робастной рекуррентной процедуры Левенберга-Марквардта для окончательной тонкой

настройки. Данная модификация позволяет повысить качество получаемой нейросетевой модели и устойчивость процесса обучения в условиях априорной и текущей неопределенности.

10. Получил дальнейшее развитие эволюционный метод устранения влияния помех при определении структур и параметров нейросетевых моделей за счет использования оценок параметров модели помехи Тьюки-Хьюбера в процедуре М-обучения. Такой подход позволяет улучшить фильтрующие свойства используемых процедур, упростить структуру хромосомы и сократить тем самым процесс получения модели ИНС.

11. Получил дальнейшее развитие эволюционный метод многокритериальной оптимизации структуры и параметров ИНС путем выделения общих для эволюционного и имунного подхода операторов. Такая модификация позволяет наиболее эффективно использовать преимущества обоих подходов.

12. Получила дальнейшее развитие архитектура эволюционирующей ИНС прямого распространения, которая помимо эволюции структуры сети и ее параметров учитывает также эволюцию алгоритма обучения и помехи. Такая ИНС обладает существенно улучшенными аппроксимирующими и экстраполирующими свойствами, что дает возможность за счет адаптации структуры, параметров и процедуры обучения эффективно обрабатывать информацию в нестационарных условиях и наличии неопределенности.

13. Получили дальнейшее развитие коэволюционирующие ИНС прямого распространения, в которых для решения задачи многокритериальной оптимизации использован алгоритм кластеризации. Такой подход позволяет упростить архитектуру получаемой нейросетевой модели и повысить ее робастность.

14. Проведено имитационное моделирование разработаных архитектур и методов обучения эволюционирующих ИНС. Показаны их преимущества перед известными нейросетевыми системами обработки информации и методами их обучения по точности, устойчивости и быстродействию в

задачах аппроксимации (эмуляции), идентификации нелинейных нестационарных процессов и интеллектуального управления нелинейными нестационарными динамическими объектами в условиях априорной и текущей неопределенности.

15. Решены практические задачи измерения температурных полей вращающихся трубчатых печей в ООО „Побужзкий фероникелевый комбинат”; управления процессом травления полосовой стали на металлургических предприятиях Украины (АТ «Співдружність-Т»); разработки АСУ ТП диффузии, дефекосатурации, выпаривания и кристаллизации в ООО “Кириковский сахарный завод”.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Хайкин С. Нейронные сети: полный курс / С. Хайкин / Пер. с англ. – М.: Вильямс, 2006. – 1104 с.

2. Осовский С. Нейронные сети для обработки информации / С. Осовский / Пер. с польского. – М.: Финансы и статистика, 2002. – 344 с.

3. Руденко О.Г. Основы теории искусственных нейронных сетей / О.Г. Руденко, Е.В. Бодянский. – Харьков: ТЕЛЕТЕХ, 2002. – 317 с.

4. Yao X. Evolving Artificial Neural Networks / X. Yao // Proc. of the IEEE. – 1999. – V.87. – №9. – P. 1423-1447.

5. Yao X. A new evolutionary system for evolving artificial neural networks / X. Yao, Y. Lin // IEEE Trans. on Neural Networks. – 1997. – v.3. – №3. – P. 694-713.

6. Floreano D. Neuroevolution: from architecture to learning / D. Floreano, P. Durr, C. Mattiussi // Evol. Intel. – 2008. – 1. – P.47-62.

7. Fogel D.B. An introduction to simulated evolutionary optimization / D.B. Fogel // IEEE Trans. on Neural Networks. – 1994. – v.5. – №1. – P. 3-14.

8. Back T. Evolutionary computation: Comments on the history and current state / T. Back, V. Hammel, H.-D. Schwefel // IEEE Trans. on Evolutionary Computation. – 1997. – v.1. – №4. – P. 3-17.

9. Zhang B. Evolutionary Computation and its Applications in Neural and Fuzzy Systems / B. Zhang, Y. Wu, J. Lu, K.-L. Du // Applied Computational Intelligence and Soft Computing. – 2011. Article ID 938240. – 20 p.

10. Whitley D. An overview of evolutionary algorithm: practical issues and common pitfalls / D. Whitley // Inf. and Software Technology. – 2001. – v.43. – P. 817-831.

11. Holland J.H. Adaption in Natural and Artificial Systems. University of Michigan Press, 1975. – 211 p.

12. Holland J.H. Adaptation in Natural and Artificial Systems 2nd edn / J. Holland – Cambridge, MIT Press. – 1992. – 228 p.

13. Goldberg D.E. Genetic Alogorithms in Search, Optimization and Machine Learning / D.E. Goldberg. – Addison Wesley Longman, 1989. – 372 p.

14. Tomassini M. A survey of genetic algorithms / M. Tomassini // Annual Reviews of Computational Physics. – 1995. – Vol.3. – P. 87-118.

15. Dumitrescu D. Evolutionary Computation / D. Dumitrescu, B. Lazzerini, L.C. Jain, A. Dumitrescu // CRC Press, 2000.

16. Michaliwicz Z. Genetic Algorithms + Data Structurs = Evolution Programs / Z. Michaliwicz – Springer-Verlag Berlin Heidelberg New York, 3. edition, 1996.

17. Goldberg D.E. Genetic algorithms and machine learning / D.E. Goldberg, J.H. Holland // Mach. Learn. – 1988. – v.3(2). – P. 95–99.

18. Mitchell T.M. Machine learning / T.M. Mitchell. – McGraw-Hill, Boston. – 1997. – 267 p.

19. Forrest S. Genetic algorithms: Principles of natural selection applied to computation / S. Forrest // Science. – 1993. – v. 261(5123). – P. 872–878.

20. Grefenstette J.J. Optimization of control parameters for genetic algorithms / J.J. Grefenstette // IEEE Transactions on Systems Man and Cybernetics.– 1986. – v. 16(1). – P. 122–128.

21. Mitchell M. An Introduction to Genetic Algorithms (Complex Adaptive Systems) / M. Mitchell. – MIT Press, Cambridge, 1998. – 221 p.

22. Coley D.A. An introduction to Genetic Algorithms for Scientists and Engineers / D.A. Coley – World Scientific Publishing, Singapore, 1999. – 244 p.

23. Rechenberg I. Evolutionsstrategie / I. Rechenberg. – Friedrich Frommann Verlag, 1973. – 217 p.

24. Bäck T. A survey of evolution strategies / T. Bäck, F. Hoffmeister, H.P. Schwefel // In: Proceedings of the 4th International Conference on Genetic Algorithms. – 1991. – P. 2-9.

25. Ohkura K. Robust evolution strategies / K. Ohkura, Y. Matsumura, K. Ueda // Applied Intelligence. – 2001. – v.15(3). – P. 153–169.

26. Huhse J. Evolution Strategy with Neighborhood Attraction–A Robust Evolution Strategy / J. Huhse, A. Zell // In: Proceedings of the Genetic and Evolutionary Computation Conference, GECCO 2001. – 2001. – P. 1026–1033.

27. Auger A. Theory of Evolution Strategies: A new perspective. / A. Auger, N. Hansen // In: Auger A., Doerr B. (eds.), Theory of Randomized Search Heuristics: Foundations and Recent Developments. – World Scientific Publishing, Singapore. – 2011. – P. 289–325.

28. Beyer H.-G. Evolution strategies – A comprehensive introduction / H.-G. Beyer, H.-P. Schwefel // Natural Computing. – 2002. – v.1(1). – P. 3–52.

29. Beyer H.-G. The theory of evolution strategies / H.-G. Beyer – Springer, Heidelberg. – 2001. – 380 p.

30. Koza J.R. Genetic Programming: On the Programming of Computers by means of Natural Selection / J.R. Koza. – The MIT Press, 1992. – 840 p.

31. Koza J.R. Genetic programming as a means for programming computers by natural selection / J.R. Koza // Statistics and Computing. – 1994. – v.4(2). – P. 87–112.

32. Ferreira C. Gene Expression Programming: a New Adaptive Algorithm for Solving Problems / C. Ferreira // Complex Systems. – 2001. – v.13(№ 2). – P. 87-129.

33. Руденко О.Г. Программирование с экспрессией генов: способы кодирования / О.Г. Руденко, С.В. Мирошниченко, А.А. Бессонов // Управляющие системы и машины. – 2015. – №3(257). – С. 82–92.

34. Руденко О.Г. Программирование с экспрессией генов:

генетические операторы / О.Г. Руденко, С.В. Мирошниченко, А.А. Бессонов // Управляющие системы и машины. – 2015. – №4(258). – С. 72–82.

35. Ferreira C. Gene Expression Programming in Problem Solving / C. Ferreira // Soft Computing and Industry Recent Applications. – Springer-Verlag, 2001. – P. 635-654.

36. Руденко О.Г. Программирование с экспрессией генов: модификации эволюционного процесса / О.Г. Руденко, С.В. Мирошниченко, А.А. Бессонов // Управляющие системы и машины. – 2015. – №5(259). – С.73-78.

37. Cybenko G. Approximation by superposition of a sigmoidal function / G. Cybenko // Math. of controls, Signals & Systems. – 1989. – 2. – P.303-314.

38. Бодянский Е.В. Искусственные нейронные сети: архитектура, обучение, применение / Е.В. Бодянский, О.Г. Руденко. – Харьков: ТЕЛЕТЕХ, 2004. – 372 с.

39. Уоссермен Ф. Нейрокомпьютерная техника / Ф. Уоссермен. – М.: Мир, 1992. – 184 с.

40. Ham F.M. Principles of Neurocomputing for Science and Engineering/ F.M. Ham, I.Kostanic. – N.Y.: Mc Graw-Hill Inc., 2001. – 468p.

41. Patterson D. Artifical Neural Networks, Theory and Application / Patterson D. – Singapur: Prenice Hall Inc., 1996. – 497p.

42. Hornik K. Multilayer feedforward networks are universal approximators / K. Hornik, M. Stinchombe, H. White // Neural Networks. – 1989. – N2. – P.359-366.

43. Billings S.A. Properties of neural networks with applications to modelling non-linear dynamical Systems / S.A. Billings, H.B. Jamaluddin, S. Chen // Int.J.Control. – 1992. – Vol.55. – N1. – P.193-224.

44. Zhu Q.M. Fast orthogonal identification of nonlinear stochastic models and radial function neural networks / Q.M. Zhu, S.A. Billings // Int.J.Control. – 1996.

– Vol.64. – №5. – P.871-886.

45. Li Y. Analysis of minimal radial basis function network algorithm for real-time identification of nonlinear dynamic systems / Y. Li, N. Sundararajan, P. Saratchandran // IEEE Proc. – Control Theory Appl. – 2000. – v.147. – №4. – P. 476 - 484.

46. Specht D.F. A general regression neural network / D.F. Specht // IEEE Trans. on Neural Networks. – 1991. – Vol.2. – №6. – P.568-576.

47. Benaim M. On functional approximation with normalised gaussian units / M. Benaim // Neural Computation. – 1989. – Vol.1. – P.281-294.

48. Shorten R. On normalizing Radial Basis function networks / R. Shorten, R. Murray-Smith // Int. Joint Conf. on Neural Networks. Nagoya, Japan, 1993. – Vol.1. – P.29-34.

49. Pham D.T. Neural Network for Identification, Prediction and Control / D.T. Pham, X. Liu. – London: Springer – Verlag, 1997. – 238p.

50. Albus J.S. A new approach to manipulator control: the cerebellar model articulation controller (CMAC) / J.S. Albus // ASME Trans., J. Dynamic Systems, Measurement and Control, 1975. – Vol. 97. – №3. – P. 220-227.

51. Albus J.S. Data storage in cerebellar model articulation controller (CMAC) / J.S. Albus // ASME Trans. J. Dynamic Systems, Measurement and Control, 1975. – Vol. 97. – №3. – P. 228–233.

52. Дремин И.М. Вейвлеты и их использование / И.М. Дремин, О.В. Иванов, В.А. Нечитайло // Успехи физических наук. – 2001. – Т.171. – №5. – С. 465-501.

53. Daugmann J. Complete discrete 2-D Gabor transforms by neural networks for image analysis and compression / J. Daugmann // IEEE Trans. Acoustic, Speech, Signal Proc. – 1988. – 36. – P. 1169-1179.

54. Pati Y.C. Discrete affine wavelet transforms for analysis and synthesis of feedforward neural networks / Y.C. Pati, P.S. Krishnaprasad // Adv. in Neural inf. Proc. Systems. – 1991. – 3. – P. 743-749.

55. Zhang Q. Wavelet networks / Q. Zhang, A. Benveniste // IEEE Trans. on Neural Networks. – 1992. – 3. – P. 889-898.

56. Руденко О.Г. Робастное обучение вейвлет-нейросетей / О.Г. Руденко, А.А. Бессонов // Междунар. научн.-техн. журн. «Проблемы управления и информатики». – 2010. – № 5.- С. 66-79.

57. Pati Y.C. Analysis and synthesis of feedforward neural networks using discrete affine wavelet transformations / Y.C. Pati, P.S. Krishnaprasad // IEEE Trans. on Neural Networks. – 1992. – 4. – P. 73-85.

58. Szu H. Neural Network adaptive wavelets for signal representation and classification / H. Szu, B. Teffer, S. Kadambe // Opt. Eng. – 1992. – 31. – P. 1907-1916.

59. Zhang J. Wavelet Neural Networks for function learning / J. Zhang, G.G. Walter, Y. Malo, W.N. Wayne Lee // IEEE Trans. On Signal Processing. – 1995. – Vol. 43(6). – P. 1485-1497.

60. Бессонов А.А. Сравнительный анализ эффективности радиально-базисных сетей и вэйвлет-нейросетей в задаче идентификации нелинейных динамических объектов / А.А. Бессонов, С.О. Руденко // Сучасні напрями розвитку інформаційно-комунікаційних технологій та засобів управління : міжнар. наук.-техн. конф : тези доп. – Харків-Київ, 13-14 грудня 2010. – С. 71.

61. Бессонов А.А. Управление сложными технологическими процессами с помощью робастных вэйвлет-нейронных сетей / А.А. Бессонов, С.О. Руденко // Інформаційні технології в навігації і управлінні: стан та перспективи розвитку : міжнар. наук.-техн. конф. : тези доп. – Київ, 5-6 липня 2010. – С. 61.

62. Петухов О.П. Введение в теорию базисов всплесков / О.П. Петухов. – С.-Петербург: Издательство СПбГТУ, 1999. – 131 с.

63. Грибунин В.Г. Теория и практика вейвлет-преобразования / В.Г. Грибунин, В.И. Воробьев. – С.-Петербург: ВУС, 1999. – 204 с.

64. Micchelli C.A. On a family of filters arising in wavelet construction / C.A. Micchelli // Applied and Computational Harmonic Analysis. – 1997. – Vol.4. – P. 38-50.

65. Zhang Q. Using Wavelet Network in Nonparametric Estimation / Q. Zhang // IEEE Trans. Neural Network. – 1997. – Vol. 8. – P.227-236.

66. Степашко В.С. Методы и критерии решения задач структурной идентификации / В.С. Степашко, Ю.Л. Кочерга // Автоматика. – 1985. – №5. – С. 29-37.

67. Лисицин Д.В. Выбор структуры многоотклико вой регрессионной модели / Д.В. Лисицин // Научный вестник НГТУ. – Новосибирск. – 2006. – №1(22). – С. 17-32.

68. Al-Subaihi A.A. Variable selection in multivariable regression using SAS/IML / A.A. Al-Subaihi // Journal of Statistical Software. – 2002. – Vol. 7. – Issue 12. –20p.

69. Bedrick E.J. Model selection for multivariate regression in small samples / E.J. Bedrick, C.C. Tsai // Biometrics. – 1994. – Vol.50. – P. 226-231.

70. Bearse P.M. Subset selection in vector autoregressive models using the genetic algorithm with informational complexity as the fitness function / P.M. Bearse, H. Bozdogan // System analysis, modeling and simulation. – 1998. – Vol. 31. – P. 61-91.

71. Cavanaugh J.E. A large-sample model selection criterion based on Kull-back's symmetric divergence / J.E. Cavanaugh // Statistics and Probability Letters. – 1999. – Vol.44. – P. 333-344.

72. Hannan E.J. The determination of the order of an autregression / E.J. Hannan, B.G. Quinn // Journal of the Royal Stastical Society, ser. B. – 1979. – Vol.41. – P. 190-195.

73. Konishi S. Generalised information criteria in model selection / S. Konishi, G. Kitagawa // Biometrika. – 1996. – Vol. 83. – P. 875-890.

74. Sugiura N. Further analysis of the data by Akaike's information criterion and the finite corrections / N. Sugiura // Communications in Statistics – Theory and Methods. – 1978. – Vol. 7. – P. 13-26.

75. Akaike H. A new look at the statistical model identification / H. Akaike // IEEE Trans. on Automatic Control. – 1974. – Vol.19. – P. 716-723.

76. Руденко О.Г. Об особенностях использования неквадратичных критериев в задаче обучения искусственных нейронных сетей / О.Г. Руденко, А.А. Бессонов, С.О. Руденко // Автоматизация: проблемы, идеи, решения : междунар. науч.-техн. конф. : тезисы докл. – Севастополь, 6-10 сентября 2010. – С. 9–11.

77. Fahlman S. Fast learning variations on back-propagation: an empirical study / Proc. of the 1988 Connectionist Models Summers School / Touretzky D.S., Hinton G.E., Sejnowski T. (Eds.) – Pittsburg.: Morgan-Kauffmann, 1988. – P. 38-51.

78. Fahlman S. The cascade-correlation learning architecture / S. Fahlman, C. Lebiere // Advances in Neural Information Processing Systems 2, D.S. Touretzky (ed.). – Los Altos CA: Morgan-Kaufmann, 1990. – P. 524-532.

79. Riley M.J. Improving the Performance of Cascade Correlation Neural Networks on Multimodal Functions / M.J. Riley, Ch.P. Thompson, K.W. Jenkins // Proc. of the World Congress of Engineering. – London, 2010. – V.III. – 7 p.

80. Усков А.А. Интеллектуальные системы управления на основе нечеткой логики / А.А. Усков, В.В. Круглов. – Смоленск: Смоленская городская типография, 2003, – 177с.

81. Руденко О.Г. Идентификация нелинейных нестационарных объектов в реальном времени с помощью радиально-базисных сетей / О.Г. Руденко, А.А. Бессонов // Кибернетика и системный анализ. – 2003. – №6. – С.177-185.

82. Руденко О.Г. Адаптивное управление многомерными нелинейными объектами на основе радиально-базисных сетей / О.Г. Руденко, А.А. Бессонов // Кибернетика и системный анализ, 2005. – №2. – С. 168-176.

83. Rudenko O.G. Echtzeit-Identifikation nichtlinearer instationärer Systeme mit RBF- Netzwerken / O.G. Rudenko, A.A. Bessonov, P. Otto, J. Wernstedt // at - Automatisierungstechnik. – №5. – Vol. 52. – 2004. – P. 209-217.

84. Rissanen J. A universal prior for integers and estimation by minimum decision length / J. Rissanen // Ann. Statist. – 1983. – 11. – P. 416-431.

85. Rissanen J. Order estimation by accumulated prediction errors / J. Rissanen // In Essay in Time Series and Allied Processes (Gani J. and Priestley H.B., eds) , J. Appl. Probab. – 1989. – 23 A. – P. 55-61.

86. Qian G. On model selection in robust linear regression / G. Qian, H.R. Künsch // Research Report 80, ETH Zentrum, Zurich, 1996. – 33 p.

87. Qian G. Computing minimum description length for robust linear regression model selection / G. Qian // Pacific Symposium of Biocomputing, 1999: Big Island of Hawaii, USA. – 1999. – 4. – P.314-325.

88. Schwarz G. Estimating the dimension of model / G. Schwarz // Ann. Statist. – 1978. – 6. – P.461-464.

89. Ronchetti E. Robustness aspects of model choice / E. Ronchetti // Statistica Sinica. – 1997. – 7. – P.327-338.

90. Ronchetti E. Robust model selection in regression / E. Ronchetti // Statist. Probab. Lett. – 1985. – 3. – P.21-23.

91. Machado J.A.F. Robust model selection and M-estimation / J.A.F. Machado // Econometrics Theory. – 1993. – 9. – P.478-493.

92. Бессонов А.А. Использование асимметричных функционалов для обучения радиально-базисных сетей / А.А. Бессонов, О.Г. Руденко, С.О. Руденко // Автоматизация: проблемы, идеи, решения : междунар. науч.-техн. конф. : тезисы докл. – Севастополь, 5-9 сентября 2011. – С. 8–9.

93. Изерман Р. Цифровые системы управления / Р. Изерман; [пер. с англ.]. – М. : Мир, 1984. – 541 с.

94. Patterson D. Artifical Neural Networks, Theory and Application / D. Patterson – Singapur: Prenice Hall Inc., 1996. – 497 p.

95. Kaczmarz S. Angenäherle Auflösung von Systemen linearer Gleichungen / S. Kaczmarz // Bull. Int. Acad. Polon. Sci. Lett., C 1, Sci. Math. Nat., Ser. A, 1937. – S. 355-357.

96. Widrow B. Adaptive switching circuits / B. Widrow, M.E. Hoff // IRE WESCON Convention Record. – N.Y., IRE, 1960. – P.96-104.

97. Руденко О.Г. Устойчивые алгоритмы обучения радиально-базисных сетей / О.Г. Руденко, А.А. Бессонов // Інтелектуальні системи прийняття рішень та проблеми обчислювального інтелекту : міжнар. наук. конф. : тези доп. – Євпаторія, 18-22 травня 2009. – С. 421–422.

98. Руденко О.Г. Робастное обучение радиально-базисных сетей / О.Г. Руденко, А.А. Бессонов // Автоматика-2009 : междунар. конф. по автоматическому управлению : тезисы докл. – Черновцы, 22-25 сентября 2009. – С. 368.

99. Shepherd A.J. Second-order methods for neural networks: Fast and reliable training methods for multi-layer perceptrons / A.J. Shepherd. – New York: Springer, 1997. – 146 p.

100. Поляк Б.Т. Введение в оптимизацию / Б.Т. Поляк. – М.: Наука, 1983. – 384с.

101. Whitley D. Lamarkian Evolution, The Baldwin Effect and Function Optimization / D. Whitley, Y.S. Gordon, K. Mathias // Proc. of the Parallel Problem Solving from Nature. – Berlin: Springer-Verl., 1994. – P.6-15.

102. Girand-Carrier Ch. Unifying Learning with Evolution Through Baldwinian Evolution and Lamarkism: A Case Study / Ch. Girand-Carrier // Proc.

Symposium on Comp. Intelligence and Learning (CoIL-2000). – Chios, Greece. – 2000. – P.36-41.

103. Darwin C. On the origin of species by means of natural selection, or the preservation of favoured races in the struggle for life / C. Darwin – John Murray, London, 1859. – 440 p.

104. Waddington C.H. Epigenetics and Evolution / C.H. Waddington // Symposia of the Society for Experimental Biology. – 1953. – v.7. – P. 186–199.

105. Jablonka E. Epigenetic inheritance and evolution: The Lamarckian dimension / E. Jablonka. – Oxford University Press, Oxford, 1999. – 360 p.

106. Tapscott D. Grown Up Digital: How the Net Generation is Changing Your World / D. Tapscott. – HC. Mcgraw-Hill, 2008. – 288 p.

107. Bird A. Perceptions of epigenetics / A. Bird // Nature. – 2007. – v. 447(7143). – P. 396–398.

108. Handel A. Is Lamarckian evolution relevant to medicine? / A. Handel, S. Ramagopalan // BMC Medical Genetics. – 2010. – v.11(1). – 73 p.

109. Braun H. ENZO-M – a hybrid approach for optimizing neural networks by evolution and learning / H. Braun, P. Zagorski // Parallel Problem, Solving from Nature. - Berlin: Springer-Verlag. – 1994. – v.3. – P. 440-451.

110. Hinton G.E. How learning can guide evolution / G.E. Hinton, S.J. Nowlan // In. Complex Systems. – 1987. – v.1. – P.495-502.

111. Ackley D.H. Interactions between learning and evolution / D.H. Ackley, M.L. Littman // In. Artificial Life. – v. 2. – P. 487-509.

112. French R.M. Genes, phenes arid the Baldwin effect: Learning and evolution in a simulated population / R. M. French, A. Messinger // In. Artificial Life. - Cambridge, MA: MIT Press, 1994. – v.4. – P. 277-282.

113. Keesing R. Evolution and learning in neural networks: the number and distribution of learning trial affect the rate of evolution / R. Keesing, D.G. Stork // Advances in Neural Information Processing Systems. – San Mateo, CA: Morgan

Kaufmann, 1991. – v.3. – P. 804-810.

114. Whitley D. Lamarckian evolution, the Baldwin effect and function optimization / D. Whitley, V.S. Gordon, K. Mathias // Parallel Problem Solving from Nature. - Berlin: Springer-Verlag, 1994 . – v. 3. – P. 6–15.

115. Federici D. Culture and the Baldwin effect / D. Federici // Lecture Notes in Computer Science. – 2003. – v. 2801. – P. 309–318.

116. Turney P. How to shift bias: lessons from the Baldwin effect / P. Turney // In. Evolutionary Computation. – 1997. – v. 4 (3). – P. 271–295.

117. Hinton G.E. How learning can guide evolution / G.E. Hinton, S.J. Nowlan // Complex Systems. – 1987. – v.1. – P. 495–502.

118. Руденко О.Г. Робастная нейроэволюционная идентификация нелинейных нестационарных объектов / О.Г. Руденко, А.А. Бессонов // Кибернетика и системный анализ. – 2014. – № 1. С.21-36.

119. Schwefel H.P. Numerical optimization of computer models / H.P. Schwefel – Chichister. – N.Y.: John Wiley & Sons, 1981. – 389 p.

120. Bäck T. An overview of evolution algorithms for parameter optimization / T. Bäck, H.-P. Schwefel // Evolutionary Computation. – 1993. – 1(1). – P. 1-23.

121. Koza J.R., Rice J.P. Genetic generation of both the weights and architecture for neural networks / J.R. Koza, J.P. Rice // Proc. IEEE Int. Joint Conf. on Neural Networks. – Seattle, WA. IEEE Press. – 1991. – v.2. – P. 71-76.

122. Lefort V. Simultaneous Optimization of Weights and Structure of an RBF Neural Network / V. Lefort, K. Knibbe, G. Beslon, J. Fevrel // In. Artificial Evolution. – Berlin: Springer-Verl., 2006. – P.49-60.

123. Ninton G.E. How learning can guide evolution / G.E. Ninton, S.J. Nowlan // Complex Systems. – 1987. – 1. – P. 495-502.

124. Chalmers D.J. The evolution of learning: an experiment in genetic connectionism / D.J. Chalmers // Proc. of the 1990 Connectionist Models Summer

School / Touretzky D.S., Elman J.L, Hinton G.E. (Eds). – San Mateo, CA: Morgan-Kauffmann, 1990. – P. 81-90.

125. Baxter J. The evolution of learning algorithms for artificial neural networks / J. Baxter // In “Complex Systems”. – Green D., Bosso-Majer T. (Eds). – Amsterdam: IOS Press, 1992. – P.313-326.

126. Kinnebrock W. Accelerating the standart backpropagation method using a genetic approach / W. Kinnebrock // Neurocomputation. – 1994. – Vol. 6. – №5-6. – P. 583-588.

127. Ku K.W.C. Approaches of Combining Local and Evolutionary Search for Training Neural Networks: A Review and Some New Results / K.W.C. Ku, M.W. Mak, W.C. Siu // In “Advances in evolutionary computing: Theory and applications”. – Berlin: Springer-Verl., 2003. – P. 615–641.

128. Ding S. Studies of Optimization Algorithms for Some Artificial Neural Networks Based on Genetic Algorithm (GA) / S. Ding, X. Xu, H. Zhu // J. of Computers. – 2011. – v.6. – №5. – P. 939-946.

129. Braun H. ENZO-M – a hybrid approach for optimazing neural networks by evolution and learning / H. Braun, P. Zagorski // Proc. of the parallel Problem Solving from Nature. - Berlin: Springer-Verl., 1994. – P.440-451.

130. Guo L., Huang D.-S., and Zhao W. Combining genetic optimization with hybrid learning algorithm for radial basis function neural networks / L. Guo, D.-S. Huang, and W. Zhao // Electron. Lett. – 2003. – vol. 39. – № 22. – P. 1600–1601.

131. Harpham C. A review of genetic algorithms applied to training radial basis function networks / C. Harpham, C.W. Dawson, M.R. Brown // Neural Comp. and Appl. – 2004. – 13(3). – P. 193-201.

132. Chun Lu. Hybrid BP-GA for multilayer feedforward neural networks / Lu Chun, Shi Bingxue, Chen Lu. // Electronics, Circuits and Systems. The 7th IEEE International Conference. – 2000. – V.2. – P. 958-961.

133. Castillo P.A. G-Prop: global optimization of multilayer perceptrons using GAs / P.A. Castillo, J.J. Merelo, V. Rivas, G. Romero, A. Prieto // Neurocomputing.– 2000.– v.35. – P. 149-163.

134. Stanley K.O. Evolving neural networks through augmenting topologies / K.O. Stanley, R. Miikkulainen // Evol. Comp. – 2002. – V.10. – №2. – P. 99-127.

135. Leung F.H.F. Tuning of the structure and parameters of a neural network using an improved genetic algorithm / F.H.F. Leung, H.K. Lam, S.H. Ling, P.K.S. Tam // IEEE Trans. on Neural Networks. – v.14. – №1. – 2003. – P. 79-88.

136. Montana D.J. Training feedforward networks using genetic algorithms / D.J. Montana, L. Davis // Proceedings of the 11th Int. Joint Conf. on Artificial Intelligence. – Detroit, Mich: Morgan-Kaufmann, 1989. – P. 762-767.

137. Neruda R. Parameter Genetic Learning of Perceptron Networks / R. Neruda, S. Slusnu // Proc. of the 10th WSEAS Int. Conf. of SYSTEMS, Athens, Greece. – 2006. – P. 92-97.

138. Gonzalez J. Multiobjective Evolutionary Optimization of the Size, Shape, and Position Parameters of Radial basis Function Networks for the Function Approximation / J. Gonzalez, I. Rojas, J. Ortega, H. Pomares, F.J. Fernandez, A.F. Diaz // IEEE Trans. on Neural Networks. – 2003. – V.14. - №6. – P. 1478-1495.

139. Wu A.S. Empirical studies of the genetic algorithm with non coding segments / A.S. Wu, R.K. Lindsay // Evolutionary Computation, 1995. – 3(2). – P. 121-147.

140. Castellano J.G. Scrapping or recycling: the role of chromosome length-altering operators in Genetic Algorithms / J.G. Castellano, P.A. Castillo, J.J. Merelo // Technical Report. GeNeura Group. Department of Architecture and Computer Technology. University of Granada. – 2001. – 19 p.

141. Руденко О.Г. Использование нормализующей компоненты при нейросетевом сжатии изображений / О.Г. Руденко, А.А. Бессонов, Р.В. Бобнев // Управляющие системы и машины. – 2013. – №5. – С.27-31.

142. Руденко О.Г. Аппроксимация гауссовских базисных функций в задаче адаптивного управления нелинейными объектами / О.Г. Руденко, А.А. Бессонов, А.С. Ляшенко, Р.А. Сунна // Кибернетика и системный анализ. – 2011. – №1. – С.3-13.

143. Руденко О.Г. Нейросетевое управление на основе кусочно-линейной аппроксимации базисных функций / О.Г. Руденко, А.А. Бессонов, А.С. Ляшенко // Материалы международной научно-технической конференции. Автоматизация: проблемы, идеи решения. – Севастополь. – 7-12 сентября 2009. – С. 26-28.

144. Бессонов А.А. Идентификация нелинейных нестационарных объектов с помощью эволюционного многослойного персептрона / А.А. Бессонов, С.О. Руденко // Вестник ХНТУ. – 2012. – №1(44). – С. 130-135.

145. Руденко О.Г. Идентификация нелинейных нестационарных объектов с помощью эволюционирующей радиально-базисной сети / О.Г. Руденко, А.А. Бессонов // Проблемы управления и информатики. – 2012. – №4. – С.5-14.

146. Castillo P.A. Evolving Multilayer Perceptrons / P.A. Castillo, J. Carpio, J.J. Merelo, A. Prieto, V. Rivas, G. Romero // Neural Processing Letters.– 2000. – v.12. – P. 115-127.

147. Buchtala O. Evolutionary optimization of radial basis function classifiers for data mining applications / O. Buchtala, M. Klimek, B. Sick // IEEE Trans. on Systems, Man, and Cybernetics, Part B. – 2005. – P. 928-947.

148. Maillard E.P. RBF neural network, basis functions and genetic algorithm / E.P. Maillard, D. Gueriot // Proc. Int. Conf. Neural Networks, Houston, TX. – 1997. – v.4. – P. 2187-2192.

149. Chen S. Combined genetic algorithm optimization and regularized orthogonal least squares learning for radial basis function networks / S. Chen, Y. Wu, B.L. Luk // IEEE Trans. on Neural Networks. – 1999. – v.10. - №5. – P. 1239-1243.

150. Jin Y. Evolutionary Optimization in Uncertain Environments – A Survey / Y. Jin, J. Branke // IEEE Tr. Evolutionary Computation. – 2005. – v.5. – №3. – P.303-317.

151. Хьюбер П. Робастность в статистике / П. Хьюбер – М.: Мир, 1984, 304 с.

152. Цыпкин Я.З. Основы информационной теории идентификации/ Я.З. Цыпкин – М.: Наука, 1984. – 320 с.

153. Цыпкин Я.З., Поляк Б.Т. Огрубленный метод максимального правдоподобия / Я.З. Цыпкин, Б.Т. Поляк // В кн. «Динамика систем». – Горький, 1977. – Вып. 12. – С. 22-46.

154. Lee C.C. Robust Radial Basis Function Networks / C.C. Lee, P.C. Chung, J.R. Tsai, C.I. Chang // IEEE Trans. on Syst., Man and Cybernetics – 1999. – v.29. – №6. – P. 674-684.

155. Руденко О.Г. Робастное обучение радиально-базисных сетей / О.Г. Руденко, А.А. Бессонов // Кибернетика и системный анализ. – 2011. – №6. – С. 38-46.

156. Руденко О.Г. М-обучение радиально-базисных сетей с использованием асимметричных функций влияния / О.Г. Руденко, А.А. Бессонов // Междунар. научн.-техн. журн. «Проблемы управления и информатики». – 2012. – № 1. – С. 79-93.

157. Rudenko O. Function Approximation Using Robust Radial Basis Function Networks / O. Rudenko, O. Bezsonov // Journal of Intelligent Learning Systems and Applications. – 2011.– №3. – P. 17-25.

158. Бессонов А.А. Обучение радиально-базисных сетей с помощью генетических алгоритмов с адаптивной мутацией / А.А. Бессонов // Системи обробки інформації. – 2012. – №3(101). – С. 177-180.

159. Hsieh J.-G. Preliminary Study on Wilcoxon Learning Machines / J.-G. Hsieh, Y.-L. Lin, J.-H. Jeng // IEEE Trans. on Neural Networks.– 2008.– V.19.– №2.– Pp. 201-211.

160. Еремеев А.В. Генетические алгоритмы и оптимизация / А.В. Еремеев // Учебное пособие. – Омск, 2008. – 34 с.

161. Рутковская Д. Нейронные сети, генетические алгоритмах и нечеткие системах / Д. Рутковская, М. Пилинвский, Л. Рутковский – М.: Горячая линия – Телеком, 2006.

162. Misevicius A. Comparison of crossover operators for the quadratic assignment problem / A. Misevicius, B. Kilda // Inf. Technol. Control. – 2005. – v.34.– P.109–119.

163. D.M. Tate A genetic approach to the quadratic assignment problem / D.M. Tate, A.E. Smith. // Computers & Operations Research. – 1995. – v.1. – P. 73–83.

164. Goldberg D.E. Alleles, Loci and the Traveling Salesman Problem / D.E. Goldberg, R. Lingle // In: Proceedings of International Conference on Genetic Algorithms and Their Applications, J.J. Grefenstette (ed.), Lawrence Erlbaum Associates, Hillsdale, NJ, 1985. – P. 154-159.

165. Migkikh V.V. Combined genetic and local search algorithm for the quadratic assignment problem / V.V. Migkikh, A.A. Topchy, V.M. Kureichik, A.Y. Tetelbaum // Proceedings of the First International Conference on Evolutionary Computation and its Applications (EVCA'96), Presidium of the Russian Academy of Sciences, Moscow, 1996. – P. 335−341.

166. Merz P. A genetic local search approach for the quadratic assignment problem / P. Merz, B. Freisleben // In T.Bäck (ed.), Proceedings of the Seventh International Conference on Genetic Algorithms, Morgan Kaufmann, East Lansing, 1997. – P. 465–472.

167. Freisleben B. A genetic local search algorithm for solving symmetric and asymmetric traveling salesman problems / B. Freisleben, P. Merz. // Proceedings of the 1996 IEEE International Conference on Evolutionary Computation, Nagoya, Japan, 1996, 616−621.

168. Merz P. A comparison of memetic algorithms, tabu search, and ant colonies for the quadratic assignment problem / P. Merz, B. Freisleben // In P.Angeline (ed.), Proceedings of 1999 Congress on Evolutionary Com-putation (CEC'99), IEEE Press, New York, 1999. – P. 2063–2070.

169. Merz P. Fitness landscape analysis and memetic algorithms for the quadratic assignment problem / P. Merz, B. Freisleben // IEEE Transactions on Evolutionary Computation. – 2000. – v.4. – P. 337−352.

170. Ahuja R.K. A greedy genetic algorithm for the quadratic assignment problem / R.K. Ahuja, J.B. Orlin, A. Tiwari // Com-puters & Operations Research, 2000. – v.27. – P. 917–934.

171. Davis L. Order-based genetic algorithms and the graph coloring problem / L. Davis // In L.Davis (ed.), Handbook of Genetic Algorithms, Van Nostrand, New York, 1991. – P. 72−90.

172. Vázquez M. A hybrid genetic algorithm for the quadratic assignment problem / M. Vázquez, L.D. Whitley // In L.D. Whitley, D.E. Goldberg, E. Cantú-Paz et al. (eds.), Proceedings of the Genetic and Evolutionary Computation Conference (GECCO'00), Morgan Kaufmann, San Francisco, 2000. – P. 135–142.

173. Drezner Z. A new genetic algorithm for the quadratic assignment problem / Z. Drezner // INFORMS Journal on Computing. – 2003. – v.15. – P. 320−330.

174. Misevicius A. Performance of hybrid genetic algorithm for the grey pattern problem / A. Misevicius, D. Rubliauskas // Information Technology and Control. – 2005. – v.34(1). – P. 15−24.

175. Sivanandam S.N. Introduction to Genetic Algorithms / S.N. Sivanandam, S.N. Deepa // Springer, 2007. – 442 p.

176. Yang J.-M. A genetic algorithm with adaptive mutations and family competition for training neural networks / J.-M. Yang, J.-T. Horng, C.-Y. Kao // International Journal of Neural Systems. – 2000. – 10(5). – P. 333-352.

177. Hong T.-P. A dynamic mutation genetic algorithm / T.-P. Hong, H.-S. Wang // Systems, Man, and Cybernetics. – 1996, IEEE International Conference. – v. 3. – P. 2000-2005.

178. Li F. Study on Genetic Algorithm Based on Schema Mutation and Its Performance Analysis / F. Li, T. Zhang // Electronic Commerce and Security. ISECS'09. Second International Symposium on. – v.1. – IEEE, 2009.

179. Deutsch S. The case for large-size mutations / S. Deutsch - IEEE Transactions on Biomedical Engineering. – 2001. – v.48, Issue: 1. – P. 124-127.

180. Haupt R.L. Practical Genetic Algorithms / R.L. Haupt, S. Haupt // A Willey-Interscience Publication, USA, 1998.

181. Cheng H. Hyper-mutation Based Genetic Algorithms for Dynamic Multicast Routing Problem in Mobile Ad Hoc Networks / H. Cheng, S. Yang // Trust, Security and Privacy in Computing and Communications (TrustCom), 2012 IEEE 11th International Conference on. IEEE, 2012.

182. Райбман Н.С. Адаптивные модели в системах управления / Н.С. Райбман, В.М. Чадеев– М.: Сов. Радио. – 1966. – 156 с.

183. Аведьян Э.Д. Модифицированные алгоритмы Качмажа для оценки параметров линейных объектов // Автоматика и телемеханика. – 1978. – №5.– С. 64-72.

184. Вазан М. Стохастическая аппроксимация / М. Вазан. – М.: Мир, 1972. – 295 с.

185. Widrow B. et al. Stationary and nonstationary learning characteristics of the LMS adaptive filter / B. Widrow // Proc. IEEE. – 1976. – V.64(8). – P.1151-1162.

186. Haykin S. Adaptive Filter Theory. – Pearson Education, 2014. – 889 p.

187. Fener A. Convergence analysis of LMS filters with uncorrelated Gaussian data / A. Fener, E. Weinstein // IEEE Trans. Acoust. Speech Signal Process., 1988. – V.33. – №1. – P. 222-230.

188. Diniz P.S.R. Adaptive Filtering: Algorithms and Practical Implementation / P.S.R. Diniz– N.Y.: Springer. – 2008. – 627 p.

189. Goodwin G.C. A Globally Convergant Adaptive Predictor / G.C. Goodwin, P.J. Ramage, P.E. Caines // Automatica. – 1981. – Vol. 17. – №1. – P.135-140.

190. Руденко О.Г. Оценка скорости сходимости одношаговых устойчивых алгоритмов идентификации / О.Г. Руденко // Доклады АН УССР. – Сер. А. Физ-мат и техн. науки. – 1982. – №1. – С.64-66.

191. Benesty J. Advances in Network and Acoustic Echo Cancellation / J. Benesty, T. Gaensler, D.R. Morgan et. al. – Berlin: Springer-Verlag, 2001. – 222 p.

192. Nagumo I. A learning method for system identification / I. Nagumo, A. Noda // IEEE Trans. Autom. Control, 1967. – AC-12. - №3. – P.282-287.

193. Benesty J. On regularization in adaptive filtering / J. Benesty, C. Palelogy, S. Ciochina // IEEE Trans. on Audio, Speech, and Language Proc. – 2011. – V.19. - №6. – P.1734-1742.

194. Benesty J. An improved PNLMS algorithm / J. Benesty, S.L. Gay // Proc. IEEE ICASSP, 2002. – V.2. – P.1881-1884.

195. Фомин В.Н. Математическая теория обучаемых опознающих систем / В.Н. Фомин – Л.: Изд. ЛГУ, 1976. – 235 с.

196. Деревицкий Д.П. Прикладная теория дискретных адаптивных систем управления / Д.П. Деревицкий, А.Л. Фрадков – М.: Наука, 1982. – 216 с.

197. Ядыкин И. Б. Адаптивное управление непрерывными технологическими процессами / И. Б. Ядыкин, В. М. Шумский, Ф. А. Овсепян –

М.: Энергоатомиздат, 1985. – 240 с.

198. Демиденко Е.З. Линейная и нелинейная регрессии / Е.З. Демиденко. – М.: Финансы и статистика, 1981. -302с.

199. Hoerl A. Ridge regression: biased estimation for nonorthogonal problems / A. Hoerl, R. Kennard // Technometrics.– 1970. – Vol.12. –N3. – P.55-67.

200. Thisted R. A critique of some ridge regression methods: comment / R. Thisted // J. Amer. Stat. Assoc. – 1980. – Vol.75. – P.81-86.

201. Агаджанов С.Г. Алгоритмы идентификации с адаптивно настраиваемой зоной нечувствительности / С.Г. Агаджанов, В.В. Роговенко, И.Д. Теренковский, В.А Тимофеев // АСУ и приборы автоматики. – 1998. – Вып. 107. – С. 86-95.

202. Бедельбаева А. А. Релейные алгоритмы оценивания / А.А. Бедельбаева // Автоматика и телемеханика. – 1978. – №1. – С. 87-95.

203. Gharieb W. Fuzzy retiming PI controller / W. Gharieb, M.A. Sheiran // Proc. IFAC-IFIP -IMACS Conf. –Belfort, France, 1997. –P. 167-171.

204. Справочник по теории автоматического управления / Под ред. А.А. Красовского. – М.: Наука, 1987. – 712 с.

205. Жуковский Е.Л. О рекуррентном способе вычисления нормальных решений линейных алгебраических уравнений / Е.Л. Жуковский, Р.Ш. Липцер // Журнал вычислительной математики и математической физики. – 1972. – Т.12. – № 4. – С.843-857.

206. Цыпкин Я.З. Дискретные адаптивные системы управления детерминированными объектами / Я.З. Цыпкин, Э.Д. Аведьян // Итоги науки и техники: техническая кибернетика. – Т. 18. – С. 45-48. Москва, ВИНИТИ, 1985.

207. Лоусон Ч. Численное решение задач метода наименьших квадратов/ Ч. Лоусон, Р. Хенсон. – М.: Наука, 1986. – 232 с.

208. Fogel E. On the value of information in system identification - bounded noise case / E. Fogel, Y.F. Huang // Automatica. – 1982. –

18. – №2. – P. 229- 238.

209. Canudas de Wit C.C. A modified EW-RLS algorithm for systems with bounded disturbances / C.C. Canudas de Wit, J. Carrilo // Automatica. – 1990. – 26. – P. 599-606.

210. Тимофеев В.А. Модифицированный рекуррентный метод наименьших квадратов с супремальными свойствами / В.А. Тимофеев // Автоматика, автоматизация, электротехнические комплексы и системы. – 2003. – №2(12). – С. 38-45.

211. Тимофеев В.А. Модифицированный алгоритм идентификации динамических объектов, возмущаемых ограниченными шумами / В.А. Тимофеев, Е.Г. Кочуев, Е.В. Тимофеева, И.В. Гурина // Вестник СумГУ, Сумы. – 2003. – №1(57). – С. 31-35.

212. Lozano-Leal R. Independent tracking and regulation adaptive control with forgetting factors / R. Lozano-Leal // Automatica. – 1983. – 19. – №1. – P. 95-97.

213. Forescue T.R. Implementation of self-tuting regulators with variable forgetting factors / T.R. Forescue, L.S. Kershenbaum, B.E. Ydstie // Automatica, 1981. –17. – №6. – P. 831-835.

214. Бессонов А.А. Нейросетевые методы построения устойчивых моделей нелинейных объектов / А.А. Бессонов // Сучасні методи, інформаційне, програмне та технічне забезпечення систем управління організаційно-технологічними комплексами : міжнар. наук.-техн. конф : тези доп. – Київ, 26-27 листопада 2009. – С. 24.

215. Руденко О.Г. Аппроксимация нелинейных функций с помощью робастных радиально-базисных сетей / О.Г. Руденко, А.А. Бессонов // Проблеми інформаційних технологій. – 2009. – №2. – С. 27-36.

216. Ljung L. Theory and practice of recursive identification / L. Ljung, T. Söderström // Cambridge, MA: MIT Press, 1983. – 529 p.

217. Ngia L.S.H. Efficient training of neural nets for nonlinear adaptive filtering using a recursive Levenberg-Marquardt algorithm / L.S.H. Ngia, J. Sjöberg // IEEE Trans. Signal Proc. – 2000. – 48(№7). – P.1915-1927.

218. Руденко О.Г. Исследование методов обучения многослойного персептрона / О.Г. Руденко, А.А. Шамраев, К.А. Лавриненко // Автоматизированные системы управления и приборы автоматики. – Харьков: ХНУРЭ. – 2002. – №121. – С.4-9.

219. Шамраев А.А. Исследование методов обучения многослойного персептрона / А.А. Шамраев, К.А. Лавриненко // 8-я Международная конф. “Теория и техника передачи, приема и обработки информ. ” (“Интегрированные информ. системы, сети и технологии”) “ИИСТ-2002”: Сб. науч. тр. – Харьков: ХНУРЭ. – 2002. – С.514-515.

220. Медведев В.С. Нейронные сети. MATLAB 6 / В.С. Медведев, В.Г. Потемкин. – М.: ДИАЛОГ-МИФИ, 2002. – 496с.

221. Rousseeuw P.J. Least Median of Squares regression / P.J. Rousseeuw // Resaerch Report No. 178, Centre for Statistics and Operations research, VUB Brussels, 1982.

222. Руденко О.Г. Робастная нейросетевая идентификация нелинейных динамических объектов / О.Г. Руденко, А.А. Бессонов // Автоматика-2010 : междунар. конф. по автоматическому управлению : тезисы докл. – Харьков, 27-29 сентября 2010. – С. 153–154.

223. Руденко О.Г. Робастная идентификация нелинейных объектов с помощью эволюционного многослойного персептрона / О.Г. Руденко, А.А. Бессонов, С.О. Руденко // Информатика, математическое моделирование, экономика : междунар. науч.-практ. конф : тезисы докл. – Смоленск, 20 апреля 2012. – С. 7–13.

224. Руденко О.Г. Робастная идентификация нелинейных объектов с помощью эволюционирующей радиально-базисной сети / О.Г. Руденко,

А.А. Бессонов, С.О. Руденко // Кибернетика и системный анализ. – 2013. – №2.– С. 15-26.

225. Мудров В.И. Методы обработки измерений. (Квазиподобные оценки) / В.И. Мудров, В.Л. Кушко – М.: Сов. радио, 1976. – 192 с.

226. Смоляк С.А. Устойчивые методы оценивания: статистическая обработка неоднородных совокупностей / С.А. Смоляк, Б.П. Титаренко – М.: Статистика, 1980. – 208 с.

227. Hampel F.R. The influence curve and its role in robust estimation / F.R. Hampel // J. Amer. Statist. Assoc. – 1974. – June. – 69. – P. 383-393.

228. Hampel F.R. Robust Statistics. The Approach Based on Influence Functions / F.R. Hampel, E.M. Ronchetti, P.J.Rousseeuw, W.A. Stahel– N.Y.: John Wiley and Sons, 1986. – 526 p.

229. Andrews D.F. A robust method for multiple linear regression / D.F. Andrews // Technometries. – 1974. – 16. – P.523-531.

230. Mosteller F. Data analysis and regression: a second course in statistics / F. Mosteller, J.W. Tukey– Addison Wesley, 1977. – 588 p.

231. Deng G. Sequential and adaptive learning algorithms for M-estimation / G. Deng // EURASIP J. Adv. in Signal Proc., 2008, ID 459586.

232. Хогг Р.В. Введение в помехоустойчивое оценивание / Р.В. Хогг // В кн. «Устойчивые стохастические методы оценки данных» – М.: Машиностроение, 1984. – С.12-25.

233. Хьюбер П.Дж. Помехоустойчивое сглаживание / П.Дж. Хьюбер // В кн. «Устойчивые стохастические методы оценки данных» – М.: Машиностроение, 1984. – С. 36-46.

234. Руденко О.Г. Робастная идентификация нелинейных объектов / О.Г. Руденко, А.А. Бессонов, С.О. Руденко // Проблеми інформаційних технологій. – №1(17) . – 2015. – С.83-86.

235. Chambers J. A Robust Mixed-Norm Adaptive Filter Algorithm / J. Chambers, A. Avlonitis // IEEE Signal Processing Letters. – 1997. – Vol. 4. – №2. – P.46-48.

236. Бессонов А.А. Применение робастных фитнесс-функций при оценивании параметров нелинейных объектов с помощью эволюционирующей радиально-базисной сети / А.А. Бессонов // Системи управління, навігації та зв'язку. – 2012. – №3(23). – С. С.51-56.

237. Allende H. M-estimators with asymmetrical influence functions: the Γ_A^0 distribution case / H. Allende, A.C. Freery, J. Galbiatis, L. Pizarro // J. Statist. Comp. and Simulation, 2006. –76, №11. – P. 941-956.

238. Ищенко Л.А. Проекционные алгоритмы идентификации линейных объектов / Л.А. Ищенко, Б.Д. Либероль, О.Г. Руденко // Докл. АН УССР. Сер. А. – 1985. – №7. – С. 62 - 64.

239. Либероль Б.Д. Выбор ширины окна в алгоритме текущего регрессионного анализа / Б.Д. Либероль, О.Г. Руденко // Доповіді НАН України. – 1996. – №3. – С. 69 - 73.

240. Либероль Б.Д. Модифицированный алгоритм Качмажа для оценивания параметров нестационарных объектов / Б.Д. Либероль, О.Г. Руденко, В.А. Тимофеев // Проблемы управления и информатики. – 1995. – №4. – С. 81-89.

241. Либероль Б.Д. О влиянии помех измерений на свойства проекционных алгоритмов идентификации / Б.Д. Либероль, О.Г. Руденко // Доповіді НАН України. – 1995. – №3. – С. 28 - 30.

242. Бессонов А.А. Оценивание параметра масштаба при робастном обучении искусственных нейронных сетей / А.А. Бессонов, О.Г. Руденко, С.О. Руденко // Проблеми інформаційних технологій. – 2010. – №2 (008). – С.81-84.

243. Hogg R.V. Statistical robustness: one view of its use in applications / R.V. Hogg // Amer. Statist. 1979. – 33. – Pp.108-111.

244. Мартин Р.Д. Устойчивый авторегрессионный анализ временных рядов / Р.Д. Мартин // с. 121-146.

245. Martin R.D. Determining the character of time series outliers / R.D. Martin, J.E. Zeh // Proc. Amer. Statist. Assoc. “Business and Economics” Section. – 1977.

246. Lee K.-M. Robust adaptive segmentation of range images / K.-M. Lee, P. Meer, R.-H. Park // IEEE Tr. Pattern Analysis and Machine Intelligence. – 1998. – 20. - №2. – Pp. 200-205.

247. Martin R.D. Robust estimation of signal amplitude / R.D. Martin // IEEE Tr. Inf. Theory. – 1972. –18. – P. 596-606.

248. Martin R.D. Robust estimation via stochastic approximation / R.D. Martin, C.J. Masreliez // IEEE Tr. Inf. Theory. – 1975. –21. – P. 263-271.

249. Ванде-Линде В.Д. Метод помехоустойчивого оценивания в задачах передачи сообщений / В.Д. Ванде-Линде // В кн. «Устойчивые стохастические методы оценки данных» - М.: Машиностроение, 1984. – С. 147-164.

250. Бессонов А.А. Устойчивое обучение радиально-базисных сетей с ограниченной точностью / А.А. Бессонов, О.Г. Руденко, С.О. Руденко // Вестник ХНТУ. – 2010. - №2(38). – С.130-134.

251. de Castro L.N., Von Zuben F.J. Learning and optimization using clonal selection principle // IEEE Tr. Evol. Comp. – 2002. – 6, №3. – P.239-251.

252. Руденко О.Г. Многокритериальная оптимизация эволюционирующих сетей прямого распространения / О.Г. Руденко, А.А. Бессонов // Проблемы управления и информатики. – 2014 . – №6. – С.29-41.

253. Кини Р.Л. Принятие решений при многих критериях: предпочтения и замещения / Р.Л. Кини, Х. Райфа // М.: Радио и связь, 1981. – 560 с.

254. Ногин В.Д. Принятие решений в многокритериальной среде: количественный подход / В.Д. Ногин // М.: ФИЗМАТЛИТ, 2002. – 144 с.

255. Подиновский В.В. Парето-оптимальные решения многокритериальных задач / В.В. Подиновский, В.Д. Ногин // М.: ФИЗМАТЛИТ, 2007. – 256 с.

256. Карпенко А.П. Популяционные методы аппроксимации множества Парето в задаче многокритериальной оптимизации. Обзор. / А.П. Карпенко, А.С. Семенихин, Е.В. Митина // Наука и образование: электронное научно-техническое издание, 2012, № 4. (http://technomag.bmstu.ru/doc/363023.html).

257. Соболь И.М. Выбор оптимальных параметров в задачах со многими критериями: Учебное пособие для ВУЗов / И.М. Соболь, Р.Б. Статников – М.: Дрофа, 2006. – 175 с.

258. Бессонов А.А. Многокритериальная нейроэволюционная оптимизация нелинейных функций / А.А. Бессонов // Системи обробки інформації. – 2012. – №9(107) . – С. 5-9.

259. Zitzler, E. An evolutionary algorithm for multiobjective optimization: The strength pareto approach / E. Zitzler, L. Thiele // Technical Report 43, Computer Engineering and Networks Laboratory (TIK), Swiss Federal Institute of Technology (ETH) Zurich, Gloriastrasse 35, CH-8092 Zurich, Switzerland, 1998. – 40 p.

260. Zitzler E. Multiobjective evolutionary algorithms: a comparative case study and the strength Pareto approach / E. Zitzler, L. Thiele // IEEE Trans. on Evolutionary Computation. – V. 3. -1999, N 4. – P. 257–271.

261. Coello C.A. Multi-objective optimization of trusses using genetic algorithms / C.A. Coello, A.D. Christiansen // Computers & Structures. – 2000. – V.75. – P. 647–660.

262. Eickhoff R. Pareto-optimal Noise and Approximation Properties of RBF Networks / R. Eickhoff, U. Rückert // In Proceedings of ICANN, 2006. – 1. – P. 993-1002.

263. Schaffer J.D. Multiple objective optimization with vector evaluated genetic algorithms / J.D. Schaffer // Proceedings of the 1st International Conference on Genetic Algorithms. – 1985. - P. 93–100.

264. Fonseca C. M. Genetic algorithm for multiobjective optimization, formulation, discussion and generalization / C. M. Fonseca, P.J. Fleming, // Genetic Algorithms: Proceeding of the Fifth International Conference. CA .- 1993. - pp. 416-423.

265. Horn J. N. A niched Pareto genetic algorithm for multiobjective optimization / J. N. Horn, A. L. Nafpliotis, D. E. Goldberg // Proceedings of the First IEEE Conference on Evolutionary Computation, IEEE World Congress on Computational Intelligence, pp. 82–87, IEEE Service Center, Piscataway, NJ, USA, Jun 1994.

266. Deb K. A Fast Elitist Multiobjective Genetic Algorithm: NSGA-II / K. Deb, A. Pratap, S. Agarwal, T. Meyarivan // IEEE Transactions on Evolutionary Computation. – v. 6. – 2002. - №. 2. – P. 182 - 197.

267. Руденко О.Г. Робастная многокритериальная идентификация нелинейных объектов на основе эволюционирующих радиально-базисных сетей / О.Г. Руденко, А.А. Бессонов // Проблемы управления и информатики. – 2013. – №5. – С. 22-32.

268. Бессонов А.А. Робастная многокритериальная идентификация нелинейных объектов с помощью сетей прямого распространения / А.А. Бессонов, С.О. Руденко // Вестник ХНТУ. – 2013. –№1(46) . – С.142-145.

269. Castillo P.A. Artificial Neural Networks Design using Evolutionary Algorithms / P.A. Castillo, M.G. Arenas, J.J. Castillo-Valdivieso, J.J. Merelo, A. Prieto, G. Romero // Advances in Soft Computing. – Springer Verlag. – 2003. – P. 43-52.

270. Tulai A. F. Combining competitive and cooperative coevolution for training cascade neural networks / A. F. Tulai, F. Oppacher // In Proc. of the Genetic

and Evolutionary Computation Conf. – New York, 2002. – P. 618–625.

271. Lewis D. Symbiosis and mutualism / D. Lewis // The Biology of Mutualism: Ecology and Evolution. – Groom Helm, 1985. – P. 29-39.

272. Margnlis L. Symbiogenesis and symbionticism / L. Margnlis, R. Fester// Symbiosis as a Source of Evolutionary Innovation. – The MIT Press, 1991. – P.1-14.

273. Зинченко Л.А. Коэволюция в методах вычислительного интеллекта / Л.А. Зинченко, С.Н. Сорокин // Двенадцатая национальная конференция по искусственному интеллекту с международным участием КИИ-2010 (20 - 24 сентября 2010 г., г. Тверь, Россия): Труды конференции. Т. 4. – М.: Физматлит, 2010. – С. 97-104.

274. Smith R. E. Cooperative versus competitive system elements in coevolutionary systems / R. E. Smith, H. Brown Cribbs III // In Proc. of the Fourth International Conference on Simulation of Adaptive Behavior, North Falmouth, MA (USA). – MIT Press, Cambridge, MA. – 1996. – P. 9-13.

275. Chand S. Cooperative coevolution of feed forward neural networks for financial time series problem / S. Chand, R. Chandra // In Int. Joint Conf. on Neural Networks (IJCNN), Beijing, China. – 2014. – P. 202–209.

276. Potter M. A. Cooperative coevolution: An architecture for evolving coadapted subcomponents / M. A. Potter, K. A. De Jong // Evol. Comput. - 2000. - Vol. 8. - № 1. – P. 1–29.

277. Chandra R. On the issue of separability for problem decomposition in cooperative neuro-evolution / R. Chandra, M. Frean, M. Zhang // Neurocomputing. – 2012. – Vol. 87. – P. 33–40.

278. Gomez F. Accelerated neural evolution through cooperatively coevolved synapses / F. Gomez, J. Schmidhuber, R. Miikkulainen // J. Mach. Learn. Res. – 2008. – Vol. 9. – P. 937–965.

279. Chandra R. An encoding scheme for cooperative coevolutionary feedforward neural networks / R. Chandra, M. Frean, M. Zhang // In AI 2010:

Advances in Artificial Intelligence, ser. Lecture Notes in Computer Science. – Springer Berlin. – Heidelberg. – 2010. – Vol. 6464. – P. 253–262.

280. Potter M. A. A cooperative coevolutionary approach to function optimization / M. A. Potter, K. A. De Jong // In PPSN III: Proc. of the Int. Conf. on Evolutionary Computation. The Third Conf. on Parallel Problem Solving from Nature. London, UK. – Springer-Verlag. – 1994. – P. 249–257.

281. Dorronsoro B. Achieving superlinear performance in parallel multi-objective evolutionary algorithms by means of cooperative coevolution / B. Dorronsoro, G. Danoy, A. Nebro, B. Bouvry // Computers and Operations Research. – 2013. – Vol. 40. – P. 1552–1563.

282. Goh C.K. A competitive and cooperative co-evolutionary approach to multi-objective particle swarm optimization algorithm design / C.K. Goh, K.C. Tan, D.S. Liu, S.C. Chiam // European J. of Operational Research. – 2010. – Vol. 202. – P. 42–54.

283. Garcia-Pedrajas N. Covnet: A cooperative coevolutionary model for evolving artificial neural networks / N. Garcia-Pedrajas, C. Hervas-Martinez, J. Munoz-Perez // IEEE Transactions on Neural Networks. – 2003. – 14(3). – P. 575–596.

284. Chand S. Multi-objective cooperative coevolution of neural networks for time series prediction / S. Chand, R. Chandra // In Int. Joint Conf. on Neural Networks (IJCNN), Beijing, China. – 2014. - P. 190-197.

285. Wiegand R.P. Robustness in cooperative coevolution / R.P. Wiegand, M.A. Potter // Proc. of the 8th annual conf. on Genetic and evolutionary computation, Seattle, Washington, USA. – 2006.

286. Perez-Godoy M.D. CO2RBFN: an evolutionary cooperative–competitive RBFN design algorithm for classification problems / M.D. Perez-Godoy, A.J. Rivera, F.J. Berlanga, M.J. Del Jesus // Soft Computing. – 2010. – №14. – P.953–971.

287. Kohonen T. Associative Memory: A System Theoretic Approach / T. Kohonen – Berlin: Springer, 1977. – 178 p.

288. Kohonen T. LVQ-PAK: The Learning Vector Quantization Program Package / T. Kohonen, J. Kangas, J. Laaksonen, K. Torkkola – Helsinki Univ. Techn., Finland, Tech. Rep. TR A30. – 1996. – 28 p.

289. Liang J.J. Novel Composition Test Functions for Numerical Global Optimization / J.J. Liang, P.N. Suganthan, K. Deb // Proc. IEEE Swarm Intelligence Symposium. - Pasadena, USA, 2005. – P. 68-75.

290. Руденко О.Г. Робастная многокритериальная идентификация нелинейных динамических объектов с помощью эволюционирующей РБС / О.Г. Руденко, А.А. Бессонов // Современные направления развития информационно-коммуникационных технологий и средств управления : междунар. науч.-техн. конф. : тезисы докл. – Полтава-Белгород-Харьков-Киев-Кировоград, 11-12 апреля 2013. – С. 47.

291. Бессонов А.А. Многокритериальная оптимизация искусственных нейронных сетей прямого распространения / А.А. Бессонов, С.О. Руденко // Стратегия качества в промышленности и образовании : междунар. конф. : тезисы докл. – Варна (Болгария), 2013. – С.443-445.

292. Бессонов А.А. Применение эволюционирующей радиально-базисной сети в системах управления с прогнозирующей моделью / А.А. Бессонов // Современные направления развития информационно-коммуникационных технологий и средств управления : междунар. науч.-техн. конф. : тезисы докл. – Полтава-Баку-Белгород-Кировоград-Харьков, 4-5 декабря 2014. – С. 22.

293. Бессонов А.А. Решение задачи управления с прогнозирующей моделью на основе эволюционирующего многослойного персептрона / А.А. Бессонов // Системи обробки інформації . – №1(126) . – 2015. – С.7-11.

294. Бессонов А.А. Идентификация нелинейных объектов с помощью эволюционной радиально-базисной сети / А.А. Бессонов, С.О. Руденко // Сучасні напрями розвитку інформаційно-комунікаційних технологій та засобів управління : міжнар. наук.-техн. конф. : тези доп. – Київ-Харків, 15-16 грудня 2011. – С. 55.

295. Руденко О.Г. Прогнозирующее управление нелинейными объектами на основе эволюционирующих нейронных сетей прямого распространения / О.Г. Руденко, А.А. Бессонов // Проблемы управления и информатики. – №6. – 2015. – С.46-55.

296. Бессонов А.А. Нейроуправление с прогнозирующей моделью / А.А. Бессонов // Информатика, математическое моделирование, экономика : междунар. науч.-практ. конф. : тезисы докл. – Смоленск, 11-15 мая 2015. – Т. 1. – С. 18–22.

297. Qin S. J. A survey of industrial model predictive control technology / S.J. Qin, T. Badgwell // Control Engineering Practice, 2003. – v.11. – №7. – P. 733-764.

298. Yu D. L. Neural model adaptation and predictive control of a chemical rig. / D. L. Yu, D. W. Yu, J. B. Gomm // IEEE Transaction on Control Systems Technology, 2006. – v.14. – № 5. – P. 828-840.

299. Lawrynczuk M. An efficient nonlinear predictive control algorithm with neural models based on multipoint on-line linearization / M. Lawrynczuk // EUROCON 2007. The International Conference on “Computer as a Tool. Warsaw. September 9-12. P. 777-784.

300. Rusnak A. Generalized predictive control based on neural networks / A. Rusnak, M. Fikar, K. Najim, A. Meszaros // Neural Process. Lett. – 1996. – № 4. – P. 107–112.

301. Бессонов А.А. Определение структуры радиально-базисной сети с помощью генетических алгоритмов / А.А. Бессонов // Інформаційні технології в

навігації і управлінні: стан та перспективи розвитку : міжнар. наук.-техн. конф. : тези доп. – Київ, 16-17 липня 2011. – С. 36.

302. Бессонов А.А. Обобщенный алгоритм обучения эволюционирующей радиально-базисной сети / А.А. Бессонов // Системи обробки інформації. – № 10(135) . – 2015. – С.163-166.

303. Kohonen T. Self-Organising Maps / T. Kohonen. – Berlin: Springer Verlag, 1995. – 362 p.

304. Martinetz T.M. A “neural-gas” network learns topologies / T.M. Martinetz, K.J. Schulten // In “Artificial Neural Networks”. – North-Holland: Elsevier Science Publishers, 1991. – P.397-402.

305. Martinetz T.M. “Neural-gas” network for vector quantization and its application to time series prediction / T.M. Martinetz, S.G. Berkovich, K.J. Schulten// IEEE Trans. on Neural Networks. – 1993. – 4(4). – P. 558-569.

306. Fritzke B. A growing neural gas network learns topologies / B. Fritzke// In Advances in Neural Information Processing Systems. – Cambridge MA, MIT Press, 1995. – v.7. – P.625-632.

307. Бахмаш М.І. Буряковий цукор технології виробництва / М.І. Бахмаш, М.І. Ігнатьев, І.А. Вітивський - Кам’янець – Подільський: Абетка – НОВА, 2004. – 372с.

308. Силин П.М. Технология сахара / П.М.Силин – Москва: Второе издание. Пищевая промышленность, 1967. – 624с.

309. Сапронов А.Р. Технология сахарного производства: Учебник и учеб. пособие для студентов высших учебных заведений / А.Р. Сапронов – М.: Колос, 1998. – 495с.

310. Бессонов А.А. Управление технологическими процессами при производстве сахара с помощью нейроэволюционных моделей / А.А. Бессонов, С.О. Руденко // Сб. научн. статей по итогам 2-ой междунар. Науч.-практ. конф.

«Информатика, математическое моделирование, экономика». Смоленск, 20 апреля 2012. – 3. – С.7-13.

311. Бессонов А.А. Построение робастных моделей технологических процессов производства сахара с помощью эволюционирующих радиально-базисных сетей / А.А. Бессонов, С.О. Руденко // Автоматизация: проблемы, идеи, решения : междунар. науч.-техн. конф. : тезисы докл. – Севастополь, 2013. – С. 170–172.

312. Негода Ф.В. Управление наклонным диффузионным аппаратом / Ф.В. Негода, А.П. Ладанюк, В.М. Лысянський. – М.: ЦНИИТЭПищпром, 1983, сер. 3. – 14 с.

313. Ладанюк А.П. Анализ динамических характеристик наклонного шнекового экстрактора как объекта управления. / А.П. Ладанюк, Ф.В. Негода, Ю.А. Янченко. – К., 1985, – 32. – Деп. в УкрНИИНТИ 18.02.85, № 357-УК-85.

314. Ляшенко С.А. Анализ эксплуатационных параметров оборудования диффузионного отделения сахарного завода / С.А. Ляшенко, А.М. Фесенко, И.С. Беляева, А.С. Ляшенко // Вісник Харківського національного технічного університету сільського господарства імені Петра Василенка «Технічний сервіс АПК, техніка та технології у сільськогосподарському машинобудуванні». – Харків, 2012. – Вип. 131. – С. 98–106.

315. Ляшенко С.А. Усовершенствование автоматизированной системы управления диффузионного отделения сахарного завода с помощью нейросетевого подхода / С. А. Ляшенко, А. С. Ляшенко // Motrol. Motorization and rower industry in agriculture. – 2009. – Vol. 11A. – Simferopol-Lublin. – P. 207–209.

316. Ляшенко С.А. Синтез нейросетевых подходов управления сложными динамическими процессами в сахарном производстве / С.А. Ляшенко // Вісник НТУ «ХПІ». – Збірник наукових праць. Серія: Системний аналіз, управління та інформаційні технології. – Х. : НТУ «ХПІ». – 2014. – № 61(1103).

– С. 30–39.

317. Беняковский М.А. Технология прокатного производства [Текст]. Т.2: справочник / М.А. Беняковский, К.Н. Богоявленский, А.И. Виткин. – М.: Металлургия, 1991. – 423с.

318. Илюнин О.О. Система нечеткого управления травлением стали с компараторной идентификацией дефектов проката [Текст] / О.О. Илюнин, А.А. Шамраев, С.Г. Удовенко, А.И. Лазарев // Системні технології. – 2011. – №3 (86). – С.151-159.

319. Бессонов А.А. Нечеткий регулятор скорости травления стали / А.А. Бессонов, О.О. Илюнин, А.В. Илюнин, // Современные направления развития информационно-коммуникационных технологий и средств управления : междунар. науч.-техн. конф. : тезисы докл. – Полтава-Баку-Кировоград-Харьков, 21-22 апреля 2016. – С. 25.

320. Козлов А.Н. Интеллектуальные информационные системы: учебник / А.Н. Козлов – Пермь: Изд-во ФГБОУ ВПО Пермская ГСХА, 2013.– 278 с.

321. Громов Ю.Ю. Интеллектуальные информационные системы и технологии: учебное пособие / Ю.Ю. Громов, О.Г. Иванова, В.В. Алексеев и др. – Тамбов: Изд-во ФГБОУ ВПО «ТГТУ», 2013. – 244 с.

322. Макаренко С.И. Интеллектуальные информационные системы: учебное пособие / С.И. Макаренко – Ставрополь: СФ МГГУ им. М.А. Шолохова, 2009.– 206 с.

323. Мигас С.С. Интеллектуальные информационные системы: конспект лекций / С.С. Мигас– Санкт-Петербург:СПбГИЭУ, 2009. – 160 с.

324. Руденко О.Г. Прогнозирование экономических процессов с помощью эволюционирующих искусственных нейронных сетей / А.А. Бессонов, О.Г. Руденко // Коллективная монография «Информационные технологи в управлении, образовании, науке и промышленности», Харьков: Издатель Рожко С.Г., 2016. – 566 с.

«ПРИЛОЖЕНИЯ»

Содержание приложений

А. Основные операторы мутаций и стратегии их применения 389
Б. Исследование вопросов сходимости процедуры (3.84)-(3.85) 403
В. Вывод рекуррентных форм многошаговых алгоритмов 405
Г. Исследование сходимости РМНК с зоной нечувствительности 412
Д. Получение нелинейных процедур обучения 416
Е. Исследование сходимости робастной процедуры (4.29) 418
Ж. Получение рекуррентной формы модифицированной многошаговой процедуры с зоной нечувствительности 423
З. Исследование сходимости процедуры обучения (4.52)-(4.53) 427
И. Математические модели ДА 429
К. Принцип работы системы АВРПП 432
Л. Акты внедрения результатов диссертационной работы 436

Приложение А

Основные операторы мутаций и стратегии их применения

При двоичном кодировании гена активизации нейрона необходимо использовать **инверсную мутацию** (рис.А.1), при которой некоторые биты хромосомы могут изменить свое значение с 1 на 0 или с 0 на 1.

Рисунок А.1 – Инверсная мутация

В случае применения **мутации обмена** (рис. А.2) случайным образом выбираются два гена хромосомы и происходит обмен их значениями.

Рисунок А.2 – Мутация обмена

Также для хромосом с бинарным кодированием применяется **реверсная мутация** (рис. А.3). При этом случайным образом выбирается ген и все последующие биты записываются в обратном порядке, формируя хромосому особи-потомка.

Рисунок А.3 – Реверсная мутация

При вещественном же кодировании параметров БФ и весовых параметров возможно использование различных типов мутации. Наиболее часто используются следующие типы мутаций.

Мутация вставкой (рис. А.4). В данном типе мутации вначале случайным образом в хромосоме выбираются два гена. Затем, второй ген перемещается таким образом, чтобы он последовал первому случайно выбранному гену, а все последующие смещаются сохраняя порядок следования.

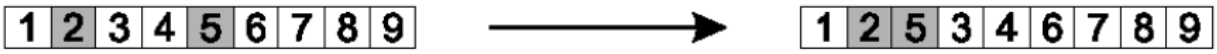


Рисунок А.4 – Мутация вставкой

Инверсная мутация. В случае инверсной мутации (рис. А.5) случайным образом выбираются два гена с позициями в хромосоме i и j, такими, что $i<j$, а затем все гены, находящиеся между ними, переставляются в хромосоме потомка в обратном порядке.

Рисунок А.5 – Инверсная мутация

Центральная инверсная мутация (рис. А.6). При таком типе мутации хромосома разбивается на две секции. При формировании хромосомы потомка все гены каждой секции инверсивно располагаются в тех же самых секциях.

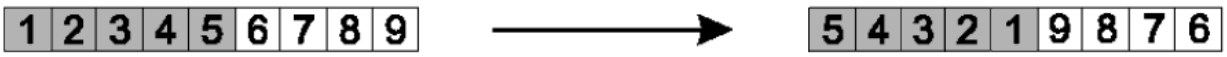


Рисунок А.6 – Центральная инверсная мутация

Мутация скремблированием (рис. А.7). При такой мутации случайным образом выбирается последовательность генов, после чего значения внутри последовательности переставляются случайным образом.

Рисунок А.7 – Мутация скремблированием

Мутация перестановкой (рис. А.8). При такой мутации случайным образом выбираются два гена и происходит обмен их значениями.

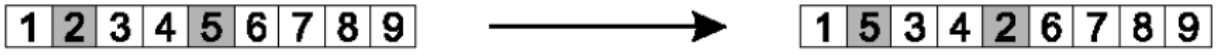


Рисунок А.8 – Мутация перестановкой

Мутация трех генов (рис. А.9). При такой мутации случайным образом выбираются три гена, которые должны занимать различные позиции в хромосоме, такие, что $i < j < k$. После этого, ген, находящийся на позиции i в хромосоме родителя перемещается на позицию j хромосомы потомка, с позиции j - на позицию k, и, наконец, ген родителя с позиции k перемещается в хромосоме потомка на позицию i.

Рисунок А.9 – Мутация трех генов

Неравномерная мутация. При неравномерной мутации ген h'_{ij} из гена $h_{ij} \in [h_{\min}, h_{\max}]$, где $h_{\min}$ и $h_{\max}$ минимально и максимально допустимые значения для данного гена, получается следующим образом:

$$h'_{ij} = \begin{cases} h_{ij} + \Delta(k, h_{\max} - h_{ij}), \text{если } \tau = 0; \\ h_{ij} - \Delta(k, h_{ij} - h_{\min}), \text{если } \tau = 1, \end{cases}$$

где τ - случайная бинарная величина с равномерным распределением. Значение функции Δ вычисляется с помощью формулы

$$\Delta(k, y) = y\left(1 - \alpha^{\left(1 - \frac{k}{T}\right)^b}\right),$$

где α - случайная величина, равномерно распределенная в интервале $[0,1]$, T - максимальное число итераций алгоритма, k – текущая итерация, b – параметр, определяющий степень неравномерности распределения.

Оператор $\Delta(k, y)$ может принимать значения из диапазона $[0, y]$, причем вероятность того, что это значение будет близко к нулю возрастает с

увеличением k. Таким образом, на начальной стадии работы ГА неравномерная мутация позволяет существенно изменять значение мутирующего гена, а на более поздних стадиях осуществляются лишь небольшие уточняющие мутации, позволяющие увеличить точность уже полученного решения.

Равномерная мутация. Данный оператор мутации заменяет значение случайно выбранного гена некоторой случайной величиной, равномерно распределенной между нижней и верхней границами допустимых значений для данного гена [175].

Следует отметить, что метод неравномерной мутации используется лишь для генов, кодирующих параметры сети. Для генов, отвечающих за активацию нейронов и вид базисной функции, используется случайная замена

$$h_{ij} = \mathrm{rand}[h_{\min}, h_{\max}],$$

где $\mathrm{rand}[x, y]$ - случайное целое число в интервале $[x, y]$, распределенное по равномерному закону. Так, для гена, отвечающего за активацию нейрона x = 0, y = 1, а для гена, определяющего вид базисной функции, x = 1, $y = M$, где M – количество используемых базисних функций.

Адаптивная мутация. Адаптивная мутация Гаусса, предложенная в работе [176] и исследованная в [158], вначале осуществляет коррекцию шага мутации, а затем производит непосредственно изменение значения мутирующего в хромосоме гена. Эту процедуру можно описать следующим образом:

$$\nu_j(k) = \nu_j(k-1)\exp[lN(0,1) + l'N(0,1)], \qquad \text{(A. 1)}$$

$$h'_{ij} = h_{ij} + \nu_j(k)N(0,1), \qquad \text{(A. 2)}$$

где $N(0,1)$ - случайная величина, распределенная по нормальному закону с нулевым математическим ожиданием и единичной дисперсией; l и l'

некоторые параметры, для которых в работе [120] были получены следующие оптимальные значения: $l = (2N)^{-0.5}$ и $l' = \left(2\sqrt{N}\right)^{-0.5}$ соответственно.

В качестве альтернативы гауссовской мутации в последнее время все чаще используется адаптивная мутация Лапласа, ставшая особенно популярной в задачах минимизации многоэкстремальных функции. Настройка шага мутации осуществляется по правилу (А. 1), т.е. как и при использовании адаптивной мутации Гаусса, а для мутации гена используется соотношение

$$h'_{ij} = h_{ij} + \nu_j(k)L(\alpha), \qquad \text{(A. 3)}$$

где L - случайная величина, распределенная по закону Лапласа с параметром α, плотность вероятности которой имеет вид:

$$f(x) = \begin{cases} \frac{1}{2}e^{\alpha x}, \text{если } x \le 0; \\ 1 - \frac{1}{2}e^{-\alpha x}, \text{если } x > 0. \end{cases} \qquad \text{(A. 4)}$$

Наконец, при использовании уменьшающейся мутации Коши, изменение шага мутации происходит следующим образом:

$$\nu_j(k) = \gamma \nu_j(k-1), \qquad \text{(A. 5)}$$

где $\gamma \in [0,1]$ - некоторая константа (в работе [176] предлагается использовать $\gamma = 0.95$).

Изменение же значения гена происходит по следующему правилу:

$$h'_{ij} = h_{ij} + \nu_j(k)C(k,t), \qquad \text{(A. 6)}$$

где $C(k,t)$ - случайная величина, распределенная по закону Коши

$$f(x) = \frac{1}{\pi}\left(\frac{\tau}{x^2 + t^2}\right) \qquad \text{(A. 7)}$$

с параметром t.

На рис.А.10 представлены плотности вероятности распределения Лапласа с $\alpha = 2$ (пунктирная линия), Гаусса с единичной дисперсией (сплошная линия), Коши с $\tau = 1$ (линия с кружками).

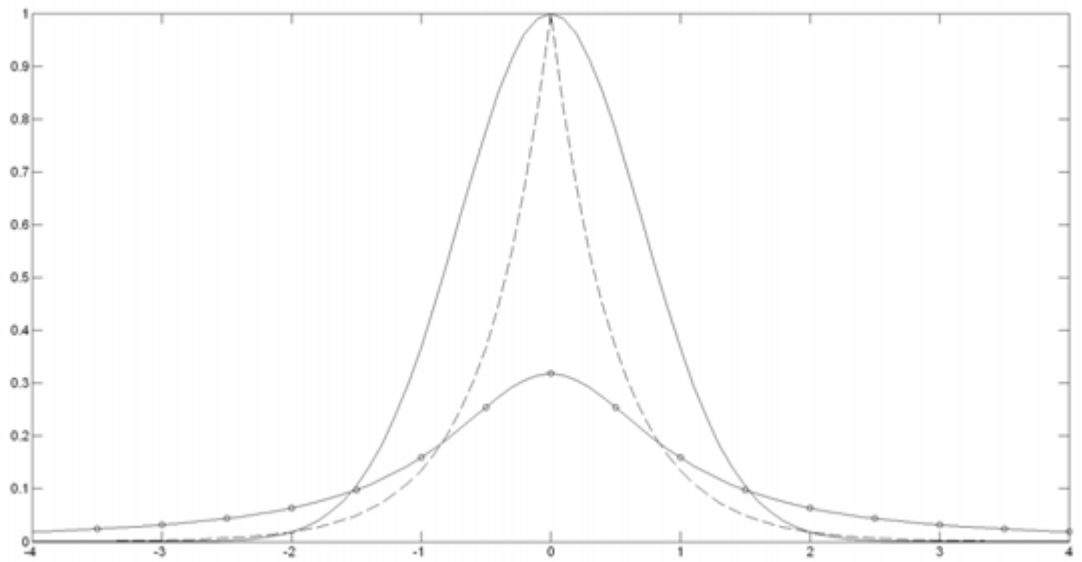


Рисунок А.10 – Плотности вероятности распределения Лапласа, Гаусса и Коши

Рассмотрим некторые стратегии применений операторов мутаций.

ГА с динамической мутацией (DMGA) [177]. В связи с тем, что различные операторы мутаций подходят для различных стадий работы ГА, данный подход позволяет автоматически выбирать соответствующий оператор мутации. ГА с динамической мутацией использует несколько операторов одновременно для генерации следующего поколения. Первоначально вероятности применения всех доступных операторов мутации равны и установлены в *l*/*N* от общего коэффициента мутации для решаемой задачи. Каждый оператор мутации затем применяется в соответствии с его заданной

вероятностью и производится оценка пригодности полученного потомства. Затем вероятности применения операторов мутаций, которые приводят к увеличению средних значений приспособленности особи, увеличиваются, а вероятности применения операторов мутаций, которые приводят к снижению средних значений приспособленности - уменьшаются. Впоследствии наиболее подходящий оператор мутации побеждает и контролирует почти все мутации, происходящие в популяции. Описать данный алгоритм можно следующим образом.

Генетический алгоритм с динамической мутацией:

Шаг 1: Определение подходящего представления решаемой задачи.

Шаг 2: Создание начальной популяции, состоящей из N особей.

Шаг 3: Определение подходящей фитнес-функции F для оценки приспособленности особей.

Шаг 4: Применение оператора кроссовера для генерации возможного потомства.

Шаг 5: Определение нескольких возможных операторов мутаций и присвоение каждому некоторой начальной вероятности его применения.

Шаг 6: Выполнение генетических операций для создания возможного потомства (каждый оператор мутации применяется со своей вероятностью).

Шаг 7: Оценка приспособленности каждой особи в новой популяции.

Шаг 8: Расчет среднего роста приспособленности, сгенерированного каждым оператором мутации.

Шаг 9: Выбор наилучших N особей в соответствии с их значениями фитнес-функции.

Шаг 10: Настройка вероятностей каждого возможного оператора мутации в соответствии со средним значением роста фитнес-функции.

Шаг 11: Если критерий останова не удовлетворен, переход к шагу 4, в противном случае - остановка алгоритма.

Когда критерий останова удовлетворяется, особь с наивысшим значением фитнес-функции является наилучшим решением.

ГА с использованием схемы мутации (ГАСМ). С целью избегания застревания алгоритма в локальном оптимуме, оператор мутации ГА может быть рассмотрен как некоторая стратегия, целью которой является постоянная генерация особей с все лучшей приспособленностью. Обычно при таком подходе применяются следующие операторы мутации [178]:

1) Классический оператор мутации (изменение каждого отдельного значения гена происходит с определенной вероятностью).

2) Неравномерная мутация (вероятность мутации выше в начале эволюции, затем в ходе эволюции вероятность мутации соответственно уменьшается).

3) Адаптивная мутация (у особей с большей приспособленностью вероятность мутации ниже, у особей же с низкой приспособленностью вероятность мутаций намного выше). Этот метод эффективно сохраняет особи с высокой приспособленностью, но с высокой вероятностью «застревает» в локальных минимумах.

В [178] предлагается стратегия мутации, основанная на мутации по схеме, с целью защиты существующих особей с высокой приспособленностью. Такой подход способствует генерации более приспособленных особей.

Шаг 1: Определение общих особенностей особей с высокой приспособленностью в текущей популяции, т.е. определение их общих генов со сходными значениями с последующим построением относительной квази-оптимальной схемы H.

Шаг 2: В процессе создания следующего поколения мутации в хромосомах особей, соответствующих схеме H, происходят с меньшей вероятностью $P_c^{(1)}$, мутации же в хромосомах особей не принадлежащих к схеме Н будут происходить с большей вероятность $P_c^{(2)}$.

Очевидно, что при $P_c^{(1)} = P_c^{(2)}$ данный алгоритм становится классическим ГА. Поэтому данный подход можно рассматривать как дальнейшее развитие классического ГА для конкретных практических задач. В качестве остальных

генетических операторов в данном алгоритме используется модифицированный пропорциональной отбор и модифицированный одноточечный кроссовер.

Генетический алгоритм с комбинированной мутацией (ГАКМ). Как уже отмечалось, операция мутации является важной процедурой, применяемой для решения проблемы «застревания» алгоритма обучения в локальных оптимумах, которая обычно направлена на постепенную генерацию особей с все лучшей приспособленностью, что приводит, в свою очередь, к эволюции всей популяции. Таким образом, основные задачи алгоритма обучения можно сформировать следующим образом:

1) Поиск наилучшего решения в текущей популяции;

2) Поддержание разнообразия текущей популяции, а также обеспечение ее дальнейшей эволюции, избегая при этом локальные сходимости.

В процессе мутации различные операторы мутаций имеют различные поисковые особенности. Так мутации с меньшей интенсивностью подходят для локального поиска, а с большей - для глобального. Для хромосом, использующих вещественное кодирования наиболее часто используются операторы равномерной мутации и мутация Гаусса.

В данном алгоритме также предлагается использовать комплексный подход к мутации с целью сохранения существующих особей с лучшей приспособленностью. Основная идея данного подхода заключается в том, что особи, представляющие собой оптимальные решения, защищены с помощью небольшой вероятности мутации в их хромосомах, в то время как суб-оптимальные решения имеют большую вероятность мутации. Такой метод позволяет значительно расширить пространство поиска и увеличить разнообразие популяции. Алгоритм можно записать следующим образом:

Шаг 1: Функция критерия мутации

Функция критерия мутации это метод, который отражает процесс биологической мутации. Пусть $m(x)$ будет функция критерия мутации и X = (x_1,

$x_2,\ldots, x_n$) обозначает хромосому мутирующей с вероятностью p_m особи. Затем, каждому гену применяется следующая процедура,

1) случайным образом число $r \in [0,1]$;

2) Если $r \geq pm$, то ген x_i не мутирует.

Шаг 2: Комбинированная стратегия мутации

На этом шаге на основе функции критерия мутации осуществляется комбинированная стратегия мутации. С помощью сортировки особей в соответствии со значением фитнесс-функции в каждом поколении, определяются особи с наиболее высоким значением целевой функции из которых составляется оптимальная под-популяция. Для особей оптимальной под-популяции соответствующим образом уменьшается интенсивность мутации с помощью изменения функции критерия мутации. Для особей же, которые не вошли в оптимальную под-популяцию, интенсивность мутаций возрастает. Такая стратегия мутаций позволяет сохранить лучшие особи каждого поколения, а также содействует поиску глобального решения.

Генетический алгоритм с кластерной адаптивной мутацией. Производительность ГА всегда чувствительна к определению таких параметров, как темпы скрещивания и вероятность мутации, численность популяции и т.д. Таким образом, многие исследователи сосредоточили свои усилия на исследованиях робастных ГА, например, бинарного ГА (BGA) и адаптивного ГА (AGA), в которых вероятность мутаций изменяется в зависимости от значений фитнес-функции решения. Решения с высоким фитнесом защищены, в то время как решения со значением фитнес-функции ниже среднего - отклоняются. В [179], авторы предположили, что мутация большого размера, при которой диапазон мутации и ее частота уменьшаются экспоненциально, позволяет получить лучшую производительность ГА. Кроме того, для ГА с вещественным кодированием, Хаупт [180] предложил использовать популяции небольшого размера с относительно большой вероятностью мутаций, что является значительно эффективнее применения

больших популяций с низкой вероятностью мутаций. К сожалению, эта адаптивная схема мутации не поддерживает подходящую скорость сходимости и надежность алгоритма одновременно. Более того, проблема выживаемости мутировавщего потомства часто игнорируется, так что большее количество мутаций приводит к избыточным вычислениям.

Для многих реальных приложений, однако, BGA, использующий алфавит символов или чисел для формирования хромосом, что является более предпочтительным. Также был разработан RGA (Real-GA), который использует вещественное или целочисленное кодирование, а также порядко-зависимое кодирование, при котором порядок генов в хромосоме играет важную роль. Проведенные экспериментальные сравнения показали, что вещественное кодирование обеспечивает лучшую производительность, чем двоичное. BGA требует большего времени вычислений для непрерывного кодирования и декодирования хромосомы в каждом поколении при переходах от генотипа к фенотипу при оценке пригодности особи. С точки зрения разнообразия хромосом ГА, известно, что разнообразие уменьшается в следствие применения оператора кроссовера и повышается благодаря мутациям. Из-за выживания наиболее приспособленных особей в процессе отбора, конвергентная сила всегда является более мощной. В связи с этим было введено понятие степени разнообразия популяции, а также было предложено количественно охарактеризовать и проанализировать теоретически проблему преждевременной сходимости в ГА с помощью цепей Маркова. Исследования показали, что степень разнообразия популяции стремится к нулю с вероятностью равной единице, что снижает способность поиска в ГА и приводит к возникновению преждевременной сходимости.

Также была рассмотрена концепция для наблюдения за ситуацией сходимости с помощью кластеризации и коэффициента выживаемости потомства для исследования причин преждевременной сходимости в ГА. Здесь под выжившим потомством понимается особь текущего поколения, которая выбирается для операции скрещивания с целью генерации нового поколения. В

результате моделирования были обнаружены некоторые основные недостатки, возникающие в процессе эволюции.

Согласно анализу, упомянутому выше, главной целью данного алгоритма является улучшение выживаемости мутировавшего потомства. Плотность популяции, рассматриваемая как принципиальная особенность описываемого алгоритма, оценивается с помощью метода кластеризации ближайших соседей, который имеет низкую вычислительную стоимость и не требует предопределения количества кластеров. После кластеризации, избыточные и повторяющиеся родительские особи отбрасываются с целью повышения конкурентоспособности потомства при операции отбора. Кроме того, мутировавшее потомство заменяет удаленные родительские особи, таким образом, что количество мутаций настраивается адаптивно и автоматически в зависимости от степени разнообразия популяции.

Сам алгоритм можно записать следующим образом.

Шаг 1: Определение параметров для генерации начальной популяции и оценка ее значения пригодности.

Шаг 2: Генерация потомства с помощью выбранного оператора кроссовера.

Шаг 3: Кластеризация родительских хромосом и устранение избыточности.

Шаг 4: Замена избыточных родительских особей мутировавшим потомством.

Шаг 5: Формирование нового поколения, состоящего из мутировавших потомков и замененных родительских хромосом.

Шаг 6: проверка критерия останова.

Генетический алгоритм с использованием гипер-мутации. В стационарных условиях от ГА ожидается сходимость с определенной скоростью для нахождения оптимальных решений многих задач оптимизации. Тем не менее, для динамической задачи оптимизации сходимость, как правило,

становится большой проблемой для ГА так как изменения условий среды обычно требуют от ГА поддержки определенного уровня разнообразия популяции для обеспечения ее способности к адаптации. Гипер-мутация является одним из основных методов, позволяющих ГА справиться с динамически изменяющейся средой. В [181] разработаны эффективные ГА на основе гипер-мутации для решения задачи динамической многоадресной маршрутизации в MANET. В традиционном ГА вероятность мутаций является фиксированной в течение всего процесса эволюции. В ГА же на основе гипер-мутаций основная идея заключается в адаптивной настройке вероятности мутаций. Так как мутации помогают генерировать новые решения, очевидно, что при изменениях в окружающей среде вероятность мутаций должна достигать высокого уровня для сохранения разнообразия популяции. Если же среда, в которой работает ГА, находится в стационарном состоянии, то вероятность мутаций должна быть снижена до некоторого низкого уровня с целью надежного исследования локального пространства поиска и улучшения уже имеющегося суб-оптимального решения.

Таким образом можно определить два типа гипер-мутаций:

1. high-low гипер-мутация;
2. постепенная гипер-мутация.

В high-low гипер-мутации предлагается новая концепция интервала изменений. Интервал изменений определяет число поколений между двумя последовательными изменениями в окружающей среде. Для первой половины интервала изменений вероятность мутаций устанавливается на заранее заданном высоком уровне, а для второй половины - на низком. При постепенной гипер-мутации в случае изменений в окружающей среде, частота мутаций увеличивается до заранее заданного высокого уровня. Затем в каждом последующем поколении частота мутаций постепенно уменьшается пока не достигнет предопределенного низкого уровня либо до следующего изменения в среде функционирования ГА. Таким образом, в случае изменений среды функционирования ГА, текущее поколение сталкивается с серьезной

проблемой сходимости и высокая вероятность мутаций позволяет ему «выпрыгнуть» из локального оптимума. После этого, способность населения к выживанию в новой среде становится все сильнее и сильнее, в связи с чем, вероятность мутаций должна становится все меньше и меньше.

Сравнительная характеристика описанных выше стратегий мутации приведена в таблице А.1.

Таблица А.1 – Сравнительная характеристика стратегий мутации

№	Название стратегии	Особенности	Преимущества	Недостатки
1.	ГА с динамической мутацией (DMGA)	Использует несколько операторов мутаций	Автоматически выбирает подходящий оператор мутации	Проблема преждевременной сходимости
2.	ГА с использованием схемы мутации (SM-GA)	Решает проблему преждевременной сходимости ГА	Превосходит обычный ГА в вычислительной эффективности и стабильности сходимости.	Проблема локального оптимума
3.	ГА с комбинированной мутацией (BCM-GA)	Специфический подход для борьбы с «застреванием» ГА в локальном оптимуме.	1) Обеспечивает поиск наилучшего решения в текущей популяции 2) Поддерживает разнообразие ущей популяции, а также обеспечивает е дальнейшую эволюцию, избегая при этом локальные сходимости.	Невозможно определить оптимальное количество мутаций
4.	ГА с кластерной адаптивной мутацией (CBAM)	Адаптивно генерирует оптимальное количество мутаций и гарантирует более высокий уровень выживания потомства	Выбирает подходящее число мутаций для уменьшения времени вычислений и поддержания разнообразия популяции для предотвращения преждевременной сходимости.	Проблемы со случайным поиском
5.	ГА с использованием гипер-мутации (HMDA)	Адаптивно настраивает вероятность мутации в процессе работы ГА. Поддерживает разнообразие популяции, позволяющее адаптироваться к изменениям среды.	Использует два различных типа гипер-мутаций для повышения производительности ГА в динамической среде.	Для ряда практических задач недопустимой является ситуация, при которой вероятность мутации постепенно возрастает.

Приложение Б

Исследование вопросов сходимости процедуры (3.84)-(3.85)

Для определения требований, которым должен удовлетворять параметр $\gamma(k)$ с целью обеспечения сходимости алгоритма, рассмотрим функцию Ляпунова вида

$$V(k) = \tilde{\Theta}^T(k) P^{-1}(k) \tilde{\Theta}(k). \qquad \text{(Б. 1)}$$

Записав (3.82) относительно ошибок оценивания $\tilde{\Theta}(i) = \hat{\Theta}(i) - \Theta$ и подставив полученное соотношение в (Б.1), имеем

$$\begin{aligned} V(k) &= (\tilde{\Theta}(k-1) - \gamma(k) P(k) \psi(k-1) e(k))^T P^{-1}(k) (\tilde{\Theta}(k-1) - \\ &- \gamma(k) P(k) \psi(k-1) e(k)) = \tilde{\Theta}^T(k-1) P^{-1}(k) \tilde{\Theta}(k-1) - \\ &- \gamma(k) \tilde{\Theta}^T(k-1) \psi(k-1) e(k) + \gamma^2(k) \psi^T(k-1) P(k) \psi(k-1) e^2(k) \quad . \end{aligned} \qquad \text{(Б. 2)}$$

Принимая во внимание, что $e(k) = \tilde{\Theta}^T(k-1)\psi(k-1) + \xi(k)$ и учитывая полученные соотношения для матриц, после несложных преобразований получаем

$$V(k) = V(k-1) + \gamma(k) w^2 k) - \gamma(k) \frac{e^2(k)}{1 + \gamma(k) \psi^T(k-1) P(k-1) \psi(k-1)}. \qquad \text{(Б. 3)}$$

Если $|\xi(k)| \le \delta$, то

$$V(k) \le V(k-1) + \gamma(k) \delta^2 - \gamma(k) \frac{e^2(k)}{1 + \gamma(k) \Psi^T(k-1) P(k-1) \Psi(k-1)}$$

и для сходимости процедуры, т.е. для выполнения условия (3.4) необходимо, чтобы

$$\gamma(k)\delta^2 - \gamma(k)\frac{e^2(k)}{1+\gamma(k)\psi^T(k-1)P(k-1)\psi(k-1)} \le 0. \qquad \text{(Б. 4)}$$

В этом случае функция Ляпунова $V(k)$ является неотрицательной и ограниченной. Ее ограничивает $V(0) = \tilde{\Theta}^T(0)P^{-1}(-1)\tilde{\Theta}(0)$. А так как $V(k) \ge 0$, последовательность $V(k)$ сходится.

Отсюда следует

$$\lim_{k \to \infty} \frac{\delta^2(1+\gamma(k)\psi^T(k-1)P(k-1)\psi(k-1)) - e^2(k)}{1+\gamma(k)\psi^T(k-1)P(k-1)\psi(k-1)} = 0. \qquad \text{(Б. 5)}$$

Таким образом, установление сходимости модификаций РМНК при различном выборе параметра $\gamma(k)$ сводится к проверке выполнения неравенства

$$\delta^2(1+\gamma(k)\psi^T(k-1)P(k-1)\psi(k-1)) - e^2(k) \le 0. \qquad \text{(Б. 6)}$$

Приложение В

Вывод рекуррентных форм многошаговых алгоритмов

Пусть на (n-1)-м шаге по L наблюдениям получена оценка $\theta(k)$

$$\theta(k-1)=\left(X_L(k-1)X_L^T(k-1)\right)^{-1}X_L(k-1)Y_L(k-1). \qquad \text{(В. 1)}$$

Обозначим

$$S_L^{-1}(k-1)=X_L(k-1)X_L^T(k-1). \qquad \text{(В. 2)}$$

При поступлении нового (n)-го наблюдения строим по L+1 наблюдениям вспомогательную оценку

$$\hat{\theta}(k)=U_{L+1}(k)X_{L+1}(k)Y_{L+1}(k), \qquad \text{(В. 3)}$$

где

$$\begin{aligned} U_{L+1}(k)&=\left(X_{L+1}(k)X_{L+1}^T(k)\right)^{-1}=\left(X_L(k-1)X_L^T(k-1)+x(k)x^T(k)\right)^{-1}= \\ &=\left(S_L^{-1}(k-1)+x(k)x^T(k)\right)^{-1} \end{aligned} \qquad \text{(В. 4)}$$

Применение к (В.4) леммы об обращении матриц при условии, что матрица $S_L^{-1}(k-1)$ – неособенная, дает

$$U_{L+1}(k)=S_L(k-1)-\frac{S_L(k-1)x(k)x^T(k)S_L(k-1)}{1+x^T(k)S_L(k-1)x(k)}.$$

Подставляя данное соотношение в (В.3) и учитывая (В.1), после несложных преобразований получаем

$$\hat{\theta}(k)=\hat{\theta}(k-1)+\frac{S_L(k-1)x(k)}{1+x^T(k)S_L(k-1)x(k)}\left(y(k)-c^T(k-1)x(k)\right).$$

А так как для получения МНК оценки с окном *L=const* необходимо исключить (*n-L*+1)-е наблюдение, можно записать

$$\hat{\theta}(k)=\hat{\theta}(k-1)+S_L(k-1)X_L(k)Y_L(k), \qquad \text{(В. 5)}$$

где

$$S_L(k)=\left(X_{L+1}(k)X_{L+1}^T(k)-x(k-L+1)x^T(k-L+1)\right)^{-1};$$
$$X_L(k)Y_L(k)=X_{L+1}(k)Y_{L+1}(k)-\mathrm{x}(k-L+1)y(k-L+1).$$

С учетом введенных обозначений и при тех же условиях, что и выше, получаем

$$S_L(k)=U_L(k-1)-\frac{U_{L+1}(k)x(k-L+1)x^T(k-L+1)U_{L+1}(k)}{1-x^T(k-L+1)U_{L+1}(k)x(k-L+1)}.$$

Подстановка данного выражения в (В.5) приводит к рекуррентной процедуре (3.86). Собирая все входящие в данную процедуру соотношения, можно окончательно записать процедуру обучения в виде (3.86)-(3.89).

Для получения рекуррентной формы процедуры (3.77), (3.79) поступим следующим образом.

Введем обозначения

$A_s(i)=X_s^T(i)X_s(i)D_s(i)=A_s^{-1}(i)$.

Представим матрицы $A_{s-1}(k)$ и $A_s(k)$ в блочном виде, т.е.

$$A_s(k-1)=\left(\begin{array}{c|c} D_{s-1}^{-1}(k-1) & X_{s-1}^T(k-1)x(k-s) \\ \hline x^T(k-s)X_{s-1}(k-1) & \|x(k-s)\|^2 \end{array}\right);$$

$$A_s(k)=\left(\begin{array}{c|c} \|x(k)\|^2 & x^T(k)X_{s-1}(k-1) \\ \hline X_{s-1}^T(k-1)x(k) & D_{s-1}^{-1}(k-1) \end{array}\right).$$

Матрицы $D_{s-1}(k-1)$ и $D_s(k)$ будем искать в виде

$$D_{s-1}(k-1)=\left(\begin{array}{c|c} F_{s-1}(k-1) & a(k-1) \\ \hline b^T(k-1) & d(k-1) \end{array}\right), \quad D_s(k)=\left(\begin{array}{c|c} \hat{d}(k) & \hat{a}^T(k) \\ \hline \hat{b}(k) & \hat{F}_{s-1}(k-1) \end{array}\right). \qquad \text{(В. 6)}$$

Использование свойства $A_i(j)D_i(j)=I$ позволяет получить систему уравнений для определения элементов матрицы $D_i(j)$. После несложных преобразований рекуррентная форма многошагового проекционного алгоритма может быть представлена следующим образом:

$$\theta(k)=\theta(k-1)+\gamma(k)X_s(k)D_s(k)E_s(k), \qquad \text{(В. 7)}$$

где матрица $D_s(k)$, имеющая вид (В.6), вычисляется по формулам

$$\hat{F}_{s-1}(k-1)=D_{s-1}(k-1)+\frac{D_{s-1}(k-1)X_{s-1}^T(k-s)x(k)x^T(k)X_{s-1}(k-1)D_{s-1}(k-1)}{x^T(k)\left(I-X_{s-1}(k-1)D_{s-1}(k-1)X_{s-1}^T(k-1)\right)x(k)}; \qquad \text{(В. 8)}$$

$$D_{s-1}(k-1)=F_{s-1}(k-1)-\frac{F_{s-1}(k-1)X_{s-1}^T(k-1)x(k-s)x^T(k-s)X_{s-1}(k-1)F_{s-1}(k-1)}{x^T(k-s)\left(I+X_{s-1}(k-1)F_{s-1}(k-1)X_{s-1}^T(k-1)\right)x(k-s)};$$

$$\hat{b}(k)=-D_{s-1}(k-1)X_{s-1}^T(k-1)x(k); \qquad \text{(В. 9)}$$

$$\hat{a}(k) = -x^T(k) X_{s-1}^T(k-1) D_{s-1}(k-1); \qquad \text{(B. 10)}$$

$$\hat{d}(k) = \left(x^T(k) \left(I - X_{s-1}(k-1) D_{s-1}(k-1) X_{s-1}^T(k-1) \right) x(k) \right)^{-1}. \qquad \text{(B. 11)}$$

Элементы $a(k-1), b(k-1), d(k-1)$ матрицы $D_{s-1}(k-1)$ вычисляются по формулам

$$a(k-1) = -F_{s-1}(k-1) X_{s-1}^T(k-1) x(k-s); \qquad \text{(B. 12)}$$

$$b(k-1) = -x^T(k-s) X_{s-1}(k-1) F_{s-1}(k-1); \qquad \text{(B. 13)}$$

$$d(k-1) = \left(x^T(k-s) \left(I + X_{s-1}(k-1) F_{s-1}(k-1) X_{s-1}^T(k-1) \right) x(k-s) \right)^{-1}. \qquad \text{(B. 14)}$$

Блочное представление матриц наблюдения позволяет получить и другую рекуррентную форму проекционного алгоритма. Так как матрица $X_s(k)$ может быть записана в виде (3.84), а обратная матрица $\left(X_s^T(k-1) X_s(k-1) \right)^{-1}$ имеет вид (B.6), то подставив полученные выше выражения для $D_{s-1}(k-1), \hat{F}_{s-1}(k-1), \hat{a}(k), \hat{b}(k), d(k)$ в (B.2) и произведя умножение (с учетом того, что $E_s^T(k) = \left(e(k) \mid E_{s-1}^T(k-1) \right)$), получим

$$\begin{aligned} \theta(k) = \theta(k-1) + \gamma(k) \frac{R_{s-1}(k-1)}{x^T(k) R_{s-1}(k-1) x(k)} e(k) + \\ \gamma(k) \left(I - \frac{R_{s-1}(k-1) x(k) x^T(k)}{x^T(k) R_{s-1}(k-1) x(k)} \right) X_{s-1}(k-1) \left(X_{s-1}^T(k-1) X_{s-1}(k-1) \right)^{-1} E_{s-1}(k-1), \end{aligned} \qquad \text{(B. 15)}$$

где

$$R_{s-1}(k-1) = I - X_{s-1}(k-1)\left(X_{s-1}^T(k-1)X_{s-1}(k-1)\right)^{-1} X_{s-1}^T(k-1),$$
$$e(k) = y(k) - \theta^T(k-1)x(k).$$

Матрица $R_{s-1}(k-1)$ в свою очередь может быть вычислена рекуррентно. Для этого разобьем матрицу $X_{s-1}(k-1)$ на блоки: $X_{s-1}(k-1) = \left(x(k-1) \mid X_{s-2}(k-2)\right)$. После несложных преобразований получаем рекуррентную процедуру (3.92) вычисления $R_{s-1}(k-1)$.

Еще одна форма рекуррентной процедуры обучения рассмотренная в [207], использует псевдообратные матрицы. В этом случае матрица $R_s(k)$ может быть записана следующим образом:

$$R_s(k) = I - X_s^+(k)X_s^T(k). \qquad \text{(B. 16)}$$

Этот же подход позволяет получить более удобную запись рекуррентной формы проекционной процедуры (3.77), (3.79). Действительно, используя блочное представление матрицы и тот факт, что для псевдообратных матриц справедливо

$$\hat{U}_s(k) = \left(X_s^T(k)X_s(k)\right)^+ = X_s^+(k)\left(X_s^T(k)\right)^+,$$

для получения соотношения между $\hat{U}_s(k)$ и $\hat{U}_{s-1}(k-1)$ можно воспользоваться теоремой Гревиля. В этом случае после несложных преобразований имеем

$$\hat{U}_s(k) = \hat{U}_{s-1}(k-1) - \frac{\hat{U}_{s-1}(k-1)x(k)x^T(k)\hat{U}_{s-1}(k-1)}{x^T(k)\hat{U}_{s-1}(k-1)x(k)} - \frac{R_{s-1}(k-1)x(k)x^T(k)\hat{U}_{s-1}(k-1)}{x^T(k)\hat{U}_{s-1}(k-1)x(k)} +$$
$$+ \frac{R_{s-1}(k-1)x(k)x^T(k)R_{s-1}(k-1)}{x^T(k)\hat{U}_{s-1}(k-1)x(k)} \frac{R_{s-1}(k-1)x(k)x^T(k)\hat{U}_{s-1}(k-1)x(k)x^T(k)R_{s-1}(k-1)}{\left\|R_{s-1}(k-1)x(k)\right\|^2},$$

(B. 17)

где $R_{s-1}(k-1) = I - X_{s-1}^{+}(k-1)X_{s-1}^{T}(k-1)$ вычисляется в соответствии с (3.96).

Использование результатов работы [205] позволяет изменить процедуру вычисления матрицы $\left(X_{s-1}^{T}(k-1)X_{s-1}(k-1)\right)^{-1}$ при поступлении нового входного вектора $x(k+1)$. В этом случае становится возможным не пересчет матрицы s раз на каждом шаге процесса идентификации, как это было реализовано в предыдущей рекуррентной форме, (3.95)-(3.97), а расширение матрицы путем добавления (n+1)-го наблюдения и последующего исключения из нее (n-s+1)-го наблюдения. Для реализации процедуры сброса (n-s+1)-го наблюдения воспользуемся следующим соотношением между матрицами $X^{+}(k)$ и $X^{+}(k+1)$:

$$\left(X^{+}(k)\,|\,0\right) = X^{+}(k+1)\left(I - gK^{T}(k)\right), \qquad \text{(В. 18)}$$

где для рассматриваемого случая $s<N$

$$K(k) = \frac{\left(I - X(k+1)X^{+}(k+1)\right)g}{g^{T}\left(I - X(k+1)X^{+}(k+1)\right)g}.$$

Здесь $g = (0,\ldots,1)^{T}$ — вектор (s+1)×1.

Используя соотношение (В.18), легко определить рекуррентную форму вычисления матрицы $\hat{U}(k)$ при сбросе (n-s+1)-го наблюдения. Принимая во внимание, что в нашем случае

$$X^{+} = X\left(X^{T}X\right)^{-1}$$

и тот факт, что

$$X_{s+1}^T(k+1)\,\mathrm{g} = x(k-s+1),$$

после подстановки выражения для $K(k)$ в (В.18) и несложных преобразований получаем

$$\hat{U}_s(k+1) = \hat{U}_{s-1}(k+1) -$$

$$-\frac{\hat{U}_{s+1}(k+1)\hat{x}(k-s+1)\hat{x}^T(k-s+1) + \hat{x}(k-s+1)\hat{x}^T(k-s+1)\hat{U}_{s-1}(k+1)}{\left\|x(k-s+1)\right\|^2} +$$

$$+\frac{\hat{x}^T(k-s+1)\hat{U}_{s+1}(k+1)\hat{x}(k-s+1)}{\left\|\hat{x}(k-s+1)\right\|^4}\hat{x}(k-s+1)\hat{x}^T(k-s+1). \qquad \text{(В. 19)}$$

Здесь $\hat{x}(k-s+1) = \hat{U}_{s+1}(k+1)x(k-s+1)$.

Аналогично

$$R_s(k+1) = R_{s+1}(k+1) + \frac{\hat{x}(k-s+1)\hat{x}^T(k-s+1)}{\left\|x(k-s+1)\right\|^2}. \qquad \text{(В. 20)}$$

Таким образом, процедуры (3.100), (В.16) - (В.17) соответствуют накоплению информации (добавлению (n+1)-го наблюдения), а (В.18) - (В.19) – сбросу (n-s+1)-го наблюдения.

Приложение Г

Исследование сходимости РМНК с зоной нечувствительности

Остановимся более подробно на выборе зоны нечувствительности для алгоритма РМНК, представленного соотношениями (3.84) - (3.85), т.е. рассмотрим модификации традиционного РМНК.

Запишем выражение (3.85), в котором вместо $e(k) = y(k) - \hat{\Theta}^T(k-1)\psi(k-1)$ используется $f(e(k),\delta)$, относительно ошибок обучения $\tilde{\Theta}(i)$

$$\tilde{\Theta}(k) = \tilde{\Theta}(k-1) - \gamma(k)P(k)\psi(k-1)f(e(k),\delta) \qquad (\Gamma.\ 1)$$

и рассмотрим функцию Ляпунова (Б. 1)

$$V(k) = \tilde{\Theta}^T k)P^{-1}(k)\tilde{\Theta}(k) = (\tilde{\Theta}(k-1) - \gamma(k)P(k)\Psi(k-1)f(e(k),\delta))^T \times$$
$$\times P^{-1}(k)(\tilde{\Theta}(k-1) - \gamma(k)P(k)\Psi(k-1)f(e(k),\delta)).$$

С учетом (3.83), (3.84) имеем

$$\begin{aligned} V(k) &= V(k-1) + \gamma(k)(\tilde{\Theta}^T(k-1)\Psi(k-1))^2 - 2\gamma(k)(\tilde{\Theta}^T(k-1)\Psi(k-1)) + \\ &+\gamma^2(k)\frac{\Psi^T(k-1)P^T(k-1)P(k-1)\Psi(k-1)f^2(e(k),\delta)}{1+\gamma(k)\Psi^T(k-1)P(k-1)\Psi(k-1)} = \\ &= V(k-1) + \gamma(k)\left[e(k) - w(k)\right]^2 - 2\gamma(k)\left[e(k) - w(k)\right]f(e(k),\delta) + \\ &+\gamma^2(k)\frac{\Psi^T(k-1)P^T(k-1)P(k-1)\Psi(k-1)f^2(e(k),\delta)}{1+\gamma(k)\Psi^T(k-1)P(k-1)\Psi(k-1)} \quad . \end{aligned} \qquad (\Gamma.\ 2)$$

Для сходимости процедуры (3.84), (3.85) необходимо выполнение условия

$$\gamma(k)\frac{\Psi^T(k-1)P(k-1)\Psi(k-1)f^2(e(k),\delta)}{1+\gamma(k)\Psi^T(k-1)P(k-1)\Psi(k-1)} - 2\gamma(k)\big[e(k)-w(k)\big]f(e(k),\delta) +$$

$$+\big[e(k)-w(k)\big]^2 \le 0 \quad .$$

Решая данное неравенство стандартным способом, получаем, что (3.84), (3.85) сходится, если

$$\big|f(e(k),\delta\big| \le \frac{(1+\gamma(k)\Psi^T(k-1)P(k-1)\Psi(k-1)) \pm \sqrt{1+\gamma(k)\Psi^T(k-1)P(k-1)\Psi(k-1)}}{\gamma(k)\Psi^T(k-1)P(k-1)\Psi(k-1)} \times$$

$$\times(\big|e(k)\big|-\delta)sign\, e(k) .$$

Определим $1+\gamma(k)\psi^T(k)P(k-1)\psi(k-1)$. С учетом (3.111), (3.112) имеем

$$1+\gamma(k)\psi^T(k-1)P(k-1)\psi(k-1) = 1+\alpha\frac{g(e_n,\beta\delta)\psi^T(k-1)P(k-1)\psi(k-1)}{1+\psi^T(k-1)P(k-1)\psi(k-1)} < 1+\alpha \quad (\Gamma.3)$$

вследствие того, что

$$\frac{g(e(k),\beta\delta)\psi^T(k-1)P(k-1)\psi(k-1)}{1+\psi^T(k-1)P(k-1)\psi(k-1)} < 1 .$$

Поэтому неравенство (Б.6) преобразуется так

$$\delta^2(1+\alpha)-e^2(k) = \delta^2\beta^2 - e^2(k) < 0 \quad . \qquad (\Gamma.4)$$

Но так как в (3.112), (3.113) параметр $\gamma(k)$ выбирается в виде (3.111), если $\big|e(k)\big| > \beta\delta$, то такой выбор $\gamma(k)$ обеспечивает сходимость процедуры.

Записав (3.117), (3.118) относительно ошибок обучения $\Theta(i)$ и подставив эти соотношения в (3.4), получим

$$\Delta V(k) \le \frac{\alpha g(e(k),\beta\delta)}{1+\psi^T(k-1)P(k-1)\psi(k-1)}\left(\frac{1+\psi^T(k-1)P(k-1)\psi(k-1)}{1+(1-\alpha g(e(k),\beta\delta)\psi^T(k-1)P(k-1)\psi(k-1)}\delta^2 - e^2(k)\right).$$

Но так как из (3.112) следует, что $0 \le g(e(k),\beta\delta) < 1$, то $\alpha g(e(k),\beta\delta) \le \alpha$, a

$$\begin{aligned}
&1+(1-\alpha g(e(k),\beta\delta))\psi^T(k-1)P(k-1)\psi(k-1) \ge \\
&\ge (1-\alpha g(e(k),\beta\delta)) + (1-\alpha g(e(k),\beta\delta))\psi^T(k-1)P(k-1)\psi(k-1) = \\
&= (1-\alpha g(e_n,\beta\delta))(1+\psi^T(k-1)P(k-1)\psi(k-1)) \ge (1-\alpha)(1+\psi^T(k-1)P(k-1)\psi(k-1)).
\end{aligned} \qquad (\Gamma.\ 5)$$

Это позволяет записать следующее неравенство:

$$\frac{1+\psi^T(k-1)P(k-1)\psi(k-1)}{1+(1-\alpha g(e(k),\beta\delta)\psi^T(k-1)P(k-1)\psi(k-1)} \le$$
$$\le \frac{1+\psi^T(k-1)P(k-1)\psi(k-1)}{(1-\alpha)(1+\psi^T(k-1)P(k-1)\psi(k-1))} = \frac{1}{1-\alpha}.$$

В связи с тем, что из (3.116) следует

$$\frac{1}{1-\alpha} = \beta^2 - \beta_0,$$

для (3.4) можно записать неравенство

$$\Delta V(k) \le \frac{\alpha g(e(k),\beta\delta)}{1+\psi^T(k-1)P(k-1)\psi(k-1)}((\beta^2-\beta_0)\delta^2-e^2(k)) =$$
$$= \frac{\alpha g(e(k),\beta\delta)}{1+\psi^T(k-1)P(k-1)\psi(k-1)}((\beta^2\delta^2-e^2(k))-\beta_0\delta^2). \qquad (Г.\ 6)$$

А так как в процедуре (3.117), (3.118) $\gamma(k)$ выбирается в виде (3.116), если $|e(k)| > \beta\delta$, то $\beta^2\delta^2 - e^2(k) < 0$, а вследствие $\beta\ \delta^2 < 0$, $\Delta V(k) \le 0$, т.е. процедуры (3.117), (3.118) сходятся.

Но как было показано выше, установление факта сходимости процедуры сводится к проверке выполнения неравенства (Г.4). Тогда, воспользовавшись неравенством (Г.5), запишем

$$\delta^2(1+\gamma(k)\psi^T(k-1)P(k-1)\psi(k-1)) - e^2(k) =$$
$$\delta^2\left(1+\frac{\alpha g(e(k),\beta\delta)\psi^T(k-1)P(k-1)\psi(k-1)}{1+(1-\alpha g(e(k),\beta\delta))\psi^T(k-1)P(k-1)\psi(k-1)}\right) - e^2(k) =$$
$$= \delta^2\left(\frac{1-\alpha+\alpha g(e(k),\beta\delta)}{1-\alpha}\right) - e^2(k) \le \frac{\delta^2}{1-\alpha} - e^2(k) =$$
$$= \delta^2(\beta^2-\beta_0) - e^2(k) = (\delta^2\beta^2 - e^2(k))\delta^2\beta_0 < 0.$$

Отсюда следует, что процедуры (3.117), (3.118) сходятся.

Приложение Д

Получение нелинейных процедур обучения

Воспользуемся разложением функционала (3.123) в ряд Тейлора для получения рекуррентных процедур обучения. Предположим, что на (k-1)-м такте получена оценка $\theta(k-1)$, минимизирующая $F\left[e(k-1)\right]$. Для определения процедуры вычисления $\theta(k)$ разложим $F(\theta(k))$ в ряд Тейлора

$$\begin{aligned} F(\theta,k) &= F(\theta(k-1),k-1) + F'(\theta(k-1),k-1)\left[\theta^* - \theta(k-1)\right] + \\ &+ \frac{1}{2}\left[\theta^* - \theta(k-1)\right]^T F''(\theta(k-1),k-1)\left[\theta^* - \theta(k-1)\right] + \\ &+ O\left(\left|\theta^* - \theta(k-1)\right|^2\right). \end{aligned} \qquad (Д.\ 1)$$

Пренебрегая членом $O\left(\left|\theta^* - \hat{\theta}(k-1)\right|^2\right)$, можно записать выражение для оценки $\hat{\theta}(k-1)$, минимизирующей (3.123)

$$\hat{\theta}(k) = \hat{\theta}(k-1) - \left[F''\left(\hat{\theta}(k-1)\right)\right]^{-1} F'\left(\hat{\theta}(k-1)\right)^T. \qquad (Д.\ 2)$$

Дифференцируя (3.123) по θ, имеем

$$F'(\theta,k)^T = -\sum_{i=1}^{k} \lambda^{k-1} \nabla f(i,\theta) e(i,\theta) = \lambda F'(\theta,k-1)^T - \nabla f(k,\theta) e(k,\theta), \qquad (Д.\ 3)$$

где $\nabla f(i,\theta) = \dfrac{\partial f(i,\theta)}{\partial \theta}$.

Аналогично для $F''\left(\theta,k\right)$ получаем

$$F''(\theta,k)=\lambda F''(\theta,k-1)+\nabla f(\theta,k)\nabla f^{T}(\theta,k)+e''(\theta,k)e(\theta,k). \qquad (Д.\ 4)$$

Так как в соответствии с предположением, что оценка $\theta(k-1)$ является оптимальной на $(k-1)$-м такте, то $F'(\theta(k-1),k-1)=0$. Если оценка $\theta(k-1)$ находится вблизи θ^{*}, то значение ошибки $e(\theta(k-1),k)$ мало и является случайной величиной. Если же $e(\theta(k-1),k)$ не коррелированна с $e''\left(\hat{\theta}(k-1),k\right)$, то $e''\left(\hat{\theta}(k-1),k\right)e\left(\hat{\theta}(k-1),k\right)=0$.

После введения обозначений $H(k)=F''(\theta(k))$, процедура (Д.2), (Д.4) может быть представлена как (3.124), (3.125).

Приложение Е

Исследование сходимости робастной процедуры (4.29)

Рассмотрим вопросы сходимости процедуры (4.29).

Запишем (4.29) относительно ошибок обучения $\tilde{\theta}(i)$

$$\tilde{\theta}(k) = \tilde{\theta}(k-1) - \gamma\left[2\lambda e(k) + (1-\lambda)\operatorname{sign} e(k)\right]x(k). \qquad \text{(Е. 1)}$$

Умножив обе части (Е.1) слева на $\tilde{\theta}^T(k)$, с учетом того, что $e(k) = \tilde{\theta}^T(k-1)x(k)$, имеем

$$\begin{aligned}\left\|\tilde{\theta}(k)\right\|^2 &= \left\|\tilde{\theta}(k-1)\right\|^2 - 4\gamma\lambda e^2(k) - 2\gamma(1-\lambda)e(k)\operatorname{sign} e(k) + \\ &+4\gamma^2\lambda^2 e^2(k)\left\|x(k)\right\|^2 + 4\gamma^2\lambda(1-\lambda)e(k)\operatorname{sign} e(k)\left\|x(k)\right\|^2 + \\ &+\gamma^2(1-\lambda)^2\left\|x(k)\right\|^2.\end{aligned} \qquad \text{(Е. 2)}$$

Из (Е.2) видно, что так как $\gamma > 0$, приращение функции Ляпунова

$$\Delta V(k) = \left\|\tilde{\theta}(k)\right\|^2 - \left\|\tilde{\theta}(k-1)\right\|^2$$

будет отрицательным, если

$$2\left(2\lambda e^2(k) + (1-\lambda)|e(k)|\right) > \gamma\left(2\lambda e(k) + (1-\lambda)|e(k)|\right)^2\left\|x(k)\right\|^2. \qquad \text{(Е. 3)}$$

Таким образом, условия сходимости процедуры (Е.1) будет выполняться, если параметр γ удовлетворяет неравенству

$$0 < \gamma < \frac{2e(k)}{\left(\lambda e(k) + (1-\lambda)\operatorname{sign} e(k)\right)\left\|x(k)\right\|^2}. \qquad \text{(Е. 4)}$$

Оптимальное значение параметра γ определяем из уравнения, получаемого путем дифференцирования (Е.2) по γ и приравнивания производной нулю. Таким образом

$$\gamma^{опт} = \frac{e(k)}{\left(\lambda e(k) + (1-\lambda)\operatorname{sign} e(k)\right)\|x(k)\|^2}. \qquad \text{(Е. 5)}$$

Исследуем статистические свойства процедуры обучения (4.29) при наличии помех измерения, т.е. $y^*(k) = \theta^{*T} x(k) + \xi(k), \quad \xi(k) \sim N\left(0, \sigma^2\right)$. Предположим, что помеха не коррелированна с полезными сигналами. Записав (4.29) относительно ошибок обучения, имеем (Е.1).

Рассмотрим математическое ожидание $M\left\{\tilde{\theta}(k)\right\}$. Усредняя обе части (Е.1), получим

$$\begin{aligned} M\left\{\tilde{\theta}(k)\right\} = M\Big\{\tilde{\theta}(k-1) - 2\gamma\lambda\left(\tilde{\theta}^T(k-1)x(k) + \xi(k)\right)x(k) - \\ -\gamma(1-\lambda)\operatorname{sign}\left(\tilde{\theta}^T(k-1)x(k)\right)x(k). \end{aligned} \qquad \text{(Е. 6)}$$

Легко видеть, что

$$\begin{aligned} M\left\{\tilde{\theta}^T(k-1)x(k)x(k)\right\} = M\left\{x(k)x^T(k)\tilde{\theta}(k-1)\right\} = \\ = M\left\{x(k)x^T(k)\right\} M\left\{\tilde{\theta}(k-1)\right\} = R_{xx} M\left\{\tilde{\theta}(k-1)\right\}, \end{aligned} \qquad \text{(Е. 7)}$$

где R_{xx} - корреляционная матрица входного сигнала.

Подробно рассмотрим выражение $M\left\{x(k)\operatorname{sign}\left(\tilde{\theta}^T(k-1)x(k)\right)x(k)\right\}$. В этом случае, если сигнал $x(k) \sim N\left(0, \sigma_x^2\right)$, имеем

$$M\left\{x(k)\operatorname{sign}\left(\tilde{\theta}^T(k-1)x(k)\right)\right\} = M\left\{M\left\{x(k)\operatorname{sign}\left(\tilde{\theta}^T(k-1)x(k)\right)|\,\tilde{\theta}(k-1)\right\}\right\} =$$

$$= M\left\{\sqrt{\frac{2}{\pi}}\frac{1}{\sigma_{e|\theta(k-1)}}M\left\{x(k)\left(\theta^T(k-1)x(k)\right)|\,\theta(k-1)\right\}\right\} =$$

$$= M\left\{\sqrt{\frac{2}{\pi\sigma_e^2}}M\left\{x(k)x^T(k)\theta(k-1)\,|\,\theta(k-1)\right\}\right\} = \qquad \text{(E. 8)}$$

$$= \sqrt{\frac{2}{\pi\sigma_e^2}}M\left\{x(k)x^T(k)\theta(k-1)\right\} = \sqrt{\frac{2}{\pi\sigma_e^2}}R_{xx}M\left\{\theta(k-1)\right\},$$

где σ_e - среднеквадратичное значение рассогласования $e(k)$.

Выражение (E. 3) получено с использованием теоремы Прайса, согласно которой для двух случайных гауссовских величин x и y с нулевыми математическими ожиданиями справедливо

$$M\left\{x\operatorname{sign} y\right\} = \sqrt{\frac{2}{\pi}}\frac{1}{\sigma_y}M\left\{xy\right\},$$

где σ_y - среднеквадратичное значение y.

С учетом свойств помехи $M\left\{\xi(k)x(k)\right\} = 0$ и выражений (E. 2), (E. 3) имеем

$$M\left\{\tilde{\theta}(k)\right\} = \left\{I - 2\gamma\lambda R_{xx} - \gamma\left(1-\lambda\right)\beta R_{xx}\right\}M\left\{\tilde{\theta}(k-1)\right\}, \qquad \text{(E. 9)}$$

откуда следует, что процедура (4.29) будет сходиться в среднем, если параметр γ удовлетворяет неравенству

$$0 < \gamma < \frac{2}{\left(2\lambda + \left(1-\lambda\right)\beta\right)trR_{xx}}. \qquad \text{(E. 10)}$$

Здесь $\beta = \sqrt{\frac{1}{\pi\sigma_e^2}}$; trR_{xx} - след матрицы R_{xx}.

Учитывая свойства входных сигналов, можем записать

$$0 < \gamma < \frac{2}{\left(2\lambda + (1-\lambda)\beta\right) N\sigma_x^2}. \qquad \text{(Е. 11)}$$

Рассмотрим величину $M\left\{\left\|\tilde{\theta}(k)\right\|^2\right\}$.

Взяв от обеих частей (Е. 2) математическое ожидание, имеем

$$\begin{aligned} M\left\{\left\|\tilde{\theta}(k)\right\|^2\right\} &= M\left\{\left\|\tilde{\theta}(k-1)\right\|^2 - 4\gamma\lambda\left(\tilde{\theta}^T(k-1)x(k)\right)^2 - \right. \\ &-2\gamma(1-\lambda)\tilde{\theta}^T(k-1)x(k)\operatorname{sign}\left(\tilde{\theta}^T(k-1)x(k)\right) + 4\gamma^2\lambda^2\left(\tilde{\theta}^T(k-1)x(k)\right)^2\left\|x(k)\right\|^2 + \\ &+4\gamma^2\lambda(1-\lambda)\left(\tilde{\theta}^T(k-1)x(k)\right)\operatorname{sign}\left(\tilde{\theta}^T(k-1)x(k)\right)\left\|x(k)\right\|^2 + \\ &\left.+\gamma^2(1-\lambda)^2\left\|x(k)\right\|^2\right\}. \end{aligned} \qquad \text{(Е. 12)}$$

Рассмотрим каждое слагаемое в правой части (Е.12) с учетом статистических свойств сигналов и помех. В нашем случае

$$M\left\{\left(\tilde{\theta}^T(k-1)x(k)\right)^2\right\} = \sigma_\xi^2 + \sigma_x^2 M\left\{\left\|\theta(k-1)\right\|^2\right\}; \qquad \text{(Е. 13)}$$

$$M\left\{\left(\tilde{\theta}^T(k-1)x(k)\right)^2\left\|x(k)\right\|^2\right\} = 3\sigma_x^4 M\left\{\left\|\tilde{\theta}(k-1)\right\|^2\right\} + N\sigma_x^2\sigma_\xi^4; \qquad \text{(Е. 14)}$$

$$M\left\{\left\|x(k)\right\|^2\right\} = N\sigma_x^2. \qquad \text{(Е. 15)}$$

Выражение для $M\left\{\tilde{\theta}^T(k-1)x(k)\operatorname{sign}\left(\theta^T(k-1)x(k)\right)\right\}$ проанализируем по аналогии с (Е.3).

$$
\begin{aligned}
&M\left\{\tilde{\theta}^T(k-1)x(k)\operatorname{sign}\left(\tilde{\theta}^T(k-1)x(k)\right)\right\}= \\
&= M\left\{M\left\{\tilde{\theta}^T(k-1)x(k)\operatorname{sign}\left(\tilde{\theta}^T(k-1)x(k)\right)\mid\tilde{\theta}(k-1)\right\}\right\}= \\
&= M\left\{\tilde{\theta}^T(k-1)\sqrt{\frac{2}{\pi\sigma_{e|\tilde{\theta}(k-1)}}}M\left\{x(k)x^T(k)\tilde{\theta}(k-1)\mid\tilde{\theta}(k-1)\right\}\right\}\approx \\
&\approx M\left\{\tilde{\theta}^T(k-1)\beta\tilde{\theta}(k-1)\right\}R_{xx}= \\
&= \beta trR_{xx}M\left\{\left\|\tilde{\theta}(k-1)\right\|^2\right\}=\beta\sigma_x^2M\left\{\left\|\tilde{\theta}(k-1)\right\|^2\right\},
\end{aligned}
\qquad \text{(E. 16)}
$$

где $\beta=\sqrt{\dfrac{2}{\pi\sigma_{e(k)}^2}}$.

Аналогично получаем

$$
M\left\{\tilde{\theta}^T(k-1)x(k)\operatorname{sign}\left(\tilde{\theta}^T(k-1)x(k)\right)\|x(k)\|^2\right\}=3\beta\sigma_x^4M\left\{\left\|\tilde{\theta}(k-1)\right\|^2\right\}. \qquad \text{(E. 17)}
$$

Подстановка выражений (E.8), (E.13)-(E.17) в (E.12) и несложные преобразования дают

$$
\begin{aligned}
&M\left\{\left\|\tilde{\theta}(k)\right\|^2\right\}=\left(1-4\gamma\lambda\sigma_x^2+2\gamma(1-\lambda)\beta\sigma_x^2+12\gamma^2\lambda^2\sigma_x^4+12\gamma^2(1-\lambda)\sigma_x^4\right)\times \\
&\times M\left\{\left\|\tilde{\theta}(k)\right\|^2\right\}+\gamma^2(1-\lambda)^2N\sigma_\xi^2\sigma_x^2.
\end{aligned}
\qquad \text{(E. 18)}
$$

Из (E.18) вытекает, что процедура (4.29) будет сходиться в среднеквадратичном при выполнении условия

$$
\left|1-4\gamma\lambda\sigma_x^2+2\gamma(1-\lambda)\beta\sigma_x^2+12\gamma^2\lambda^2\sigma_x^4+12\gamma^2(1-\lambda)\sigma_x^4\right|<1,
$$

откуда вытекают ограничения для параметра γ (4.30).

Приложение Ж

Получение рекуррентной формы модифицированной многошаговой процедуры с зоной нечувствительности

Введём следующее обозначение:

$$\nabla f_s\ (k)\left(\nabla f_s^T(k)\nabla f_s\ (k)\right)^{-1} F_s\ (e,\Delta,k) = r_s\ (k). \qquad (Ж.\ 1)$$

Тогда алгоритм (4.42) принимает вид

$$\theta(k) = \theta(k-1) + \gamma r_s\ (k). \qquad (Ж.\ 2)$$

Неудобством практического применения алгоритма (Ж.2) является необходимость вычисления на каждом такте обратной матрицы $\nabla f_s^T(k)\nabla f_s\ (k)$ размерности SxS. Рассмотрим получение более удобной в вычислительном отношении рекуррентной процедуры, не использующей непосредственно вычисления этой обратной матрицы [238].

Разобьём матрицу $\nabla f_s\ (k)$ и вектор $F_s\ (k)$ на блоки

$$\nabla f_s\ (k) = \left(\nabla f(k) \vdots \nabla f_{s-1}(k-1)\right),\ F_s\ (e,\Delta,\mathrm{k}) = \begin{pmatrix} F(e,\Delta,k) \\ \dots \\ F_{s-1}(e,\Delta,k-1) \end{pmatrix},$$

где $\nabla f_{s-1}(k-1) = \left(\nabla f(k-1),...,\nabla f(k-s+1)\right)$ - матрица Nx(s-1),

$F_{s-1}(e,\Delta,k-1) = \left(f(e,\Delta,k-1),...,f(e,\Delta,k-s+1)\right)^T$ - вектор (s-1)x1.

С учетом этого [240]

$$\left(\nabla f_s^T(k)\nabla f_s\,(k)\right)^{-1}=\begin{pmatrix}\|\nabla f(k)\|^2 & \nabla f^T(k)\nabla f_{s-1}(k-1)\\ \nabla f_{s-1}^T(k-1) & \nabla f_{s-1}^T(k-1)\nabla f_{s-1}(k-1)\end{pmatrix}^{-1}=\begin{pmatrix}d & a^T\\ a & D\end{pmatrix},\qquad (Ж.\ 3)$$

где

$$d=\left[\nabla f^T(k)\left(I-\nabla f_{s-1}(k-1)\left(\nabla f_{s-1}^T(k-1)\nabla f_{s-1}(k-1)\right)^{-1}\nabla f_{s-1}(k-1)\right)\nabla f(k)\right]^{-1};$$

$$a=-d\left(\nabla f_{s-1}^T(k-1)\nabla f_{s-1}(k-1)\right)^{-1}\nabla f_{s-1}(k-1)\nabla f(k);$$

$$\begin{aligned}D=&\left(\nabla f_{s-1}^T(k-1)\nabla f_{s-1}(k-1)\right)^{-1}-\\ &-d\left(\nabla f_{s-1}^T(k-1)\nabla f_{s-1}(k-1)\right)^{-1}\nabla f_{s-1}^T(k-1)\nabla f(k)\nabla f^T(k)\nabla f_{s-1}(k-1)\times\\ &\times\left(\nabla f_{s-1}^T(k-1)\nabla f_{s-1}(k-1)\right)^{-1}.\end{aligned}\qquad (Ж.\ 4)$$

Подставляя полученные выражения в (4.42), имеем

$$\begin{aligned}&\theta(k)=\theta(k-1)+\\ &+d\left[I-\nabla f_{s-1}(k-1)\left(\nabla f_{s-1}^T(k-1)\nabla f_{s-1}(k-1)\right)^{-1}\nabla f_{s-1}^T(k-1)\right]\nabla f(k)f(e,\Delta,k)+\\ &+\nabla f_{s-1}^T(k-1)\left(\nabla f_{s-1}^T(k-1)\nabla f_{s-1}(k-1)\right)^{-1}F_{s-1}(e,\Delta,k-1)-\\ &-d\left[I-\nabla f_{s-1}(k-1)\left(\nabla f_{s-1}^T(k-1)\nabla f_{s-1}(k-1)\right)^{-1}\nabla f_{s-1}^T(k-1)\right]\nabla f(k)\nabla f^T(k)F(e,\delta,k-1)\end{aligned}$$

$$(Ж.\ 5)$$

Обозначим

$$I-\nabla f_{s-1}(k-1)\left(\nabla f_{s-1}^T(k-1)\nabla f_{s-1}(k-1)\right)^{-1}\nabla f_{s-1}^T(k-1)=R_{s-1}(k-1).\qquad (Ж.\ 6)$$

Тогда выражение (Ж.5) можно переписать следующим образом:

$$\begin{aligned}&\theta(k)=\theta(k-1)+\gamma\frac{R_{s-1}(k-1)\nabla f(k)}{\nabla f^T(k)R_{s-1}(k-1)\nabla f(k)}f(e,\Delta,k)+\\ &+\gamma\left(I-\frac{R_{s-1}(k-1)\nabla f(k)\nabla f^T(k)}{\nabla f^T(k)R_{s-1}(k-1)\nabla f(k)}\right)\nabla f_{s-1}(k-1)\left(\nabla f_{s-1}^T(k-1)\nabla f_{s-1}(k-1)\right)^{-1}F_{s-1}(e,\Delta,k-1)\end{aligned}$$

$$(Ж.\ 7)$$

или с учетом введенного обозначения (Ж.1)

$$\theta(k)=\theta(k-1)+\gamma\frac{R_{s-1}(k-1)\nabla f(k)}{\nabla f^T(k)R_{s-1}(k-1)\nabla f(k)}f(e,\Delta,k)+$$
$$+\gamma\left(I-\frac{R_{s-1}(k-1)\nabla f(k)\nabla f^T(k)}{\nabla f^T(k)R_{s-1}(k-1)\nabla f(k)}\right)r_{s-1}(k-1). \qquad (Ж.\ 8)$$

Матрица $R_{s-1}(k-1)$ в свою очередь также может быть вычислена рекуррентно. Для этого разобъём $\nabla f_{s-1}(k-1)$ на блоки:

$$\nabla f_{s-1}(k-1)=\left(\nabla f(k)\vdots\nabla f_{s-2}(k-2)\right).$$

Тогда

$$R_{s-1}(k-1)=I-\nabla f_{s-1}(k-1)\left(\nabla f_{s-1}^T(k-1)\nabla f_{s-1}(k-1)\right)^{-1}\nabla f_{s-1}(k-1)=$$
$$=I-\left(\nabla f(k-1)\vdots\nabla f_{s-2}^T(k-2)\right)\times$$
$$\times\begin{pmatrix}\|\nabla f(k-1)\|^2 & \nabla f^T(k-1)\nabla f_{s-2}(k-2)\\ \nabla f_{s-2}^T(k-2)\nabla f(k-1) & \nabla f_{s-2}^T(k-2)\nabla f_{s-2}(k-2)\end{pmatrix}^{-1}\begin{pmatrix}\nabla f^T(k-1)\\ \cdots\\ \nabla f_{s-2}^T(k-2)\end{pmatrix}.$$

Воспользовавшись выражением для обратной матрицы (Ж.3), элементы которой определяются в соответствии с (Ж.4), после несложных преобразований получим

$$R_{s-1}(k-1)=R_{s-2}(k-2)-\frac{R_{s-2}(k-2)\nabla f(k-1)\nabla f^T(k-1)R_{s-2}(k-2)}{\nabla f^T(k-2)R_{s-2}(k-2)\nabla f(k-1)}. \qquad (Ж.\ 9)$$

Но так как особенностью конечношаговых алгоритмов является использование ограниченного количества измерений (в нашем случае *S*) на каждом такте процесса обучения, то очевидно, что на любом такте после поступления $n \geq S$ измерений матрица $R_{s-1}(k-1)$, вычисляемая в соответствии с (Ж.9), должна содержать (*s*-1) измерение, т.е. на каждом такте процесса обучения матрица $R_{s-1}(k-1)$ должна пересчитываться (*s*-1) раз. Таким образом соответствующая процедура вычисления матрицы $R_{s-1}(k-1)$ будет иметь вид

$$R_i(k-s+i) = R_{i-1}(k-s+i-1) - \\ - \frac{R_{i-1}(k-s+i-1)\nabla f(k-s+i)\nabla f^T(k-s+i)R_{i-1}(k-s+i-1)}{\nabla f^T(k-s+i)R_{i-1}(k-s+i-1)\nabla f(k-s+i)}, \quad \text{(Ж. 10)}$$
$$i = 1, 2, ..., s-1.$$

Легко видеть, что при *s* = 1 данная процедура совпадает с одношаговым алгоритмом Качмажа (3.1). Это означает, что элементы начальных значений матрицы $R_{s-1}(k-1)$ выбираются следующим образом:

$$R_0(k-1) = I.$$

По аналогии с вышеизложенным легко получить рекуррентное соотношение для вычисления $r_s(k)$. Опуская несложные преобразования, имеем

$$r_s(k) = r_{s-1}(k-1) - \frac{R_{s-1}(k-1)\nabla f(k)}{\nabla f^T(k)R_{s-1}(k-1)\nabla f(k)}. \quad \text{(Ж. 11)}$$

Таким образом, окончательно модифицированная многошаговая процедура, содержащая зону нечувствительности, будет иметь вид (Ж.8), (Ж.10), (Ж.11).

Приложение З

Исследование сходимости процедуры обучения (4.52)-(4.53)

Для исследования сходимости процедуры обучения (4.52)-(4.53) поступаем по аналогии с вышеизложенным, т.е. запишем (4.53) относительно ошибок обучения θ, и т.д. Опуская несложные преобразования, получим, что условие сходимости (4.48) выполняется при

$$f^2\left(e,\Delta,k\right) \le \frac{2\theta^T(k-1)R_{s-1}(k-1)\nabla f(k) f(k)\left(e(k),\Delta,k\right)}{\gamma}, \qquad (З.\ 1)$$

а при выборе $\gamma = 1$

$$f^2\left(e,\Delta,k\right) \le 2\theta^T(k-1)R_{s-1}(k-1)\nabla f(k) f\left(e(k),\Delta,k\right).$$

Данное неравенство можно записать следующим образом:

$$\left|f\left(e,\Delta,k\right)\right| \le 2\theta^T(k-1)R_{s-1}(k-1)\nabla f(k)\operatorname{sign} e(k). \qquad (З.\ 2)$$

Однако, как следует из вышеизложенного, практическая проверка условия сходимости (З.2) затруднена, так как невозможно вычислить величину $\theta^T(k-1)R_{s-1}(k-1)\nabla f(k)$.

При рассмотрении нестационарного случая предложим, что

$$\theta^*(k) = \theta^*(k-1) + \Delta\theta^*(k), \qquad (З.\ 3)$$

где $\Delta\theta^*(k)$ - вектор изменения параметров на k-м такте.

Вычитая из обеих частей (З.8) $\theta^*(k)$ и записывая алгоритм относительно ошибок обучения, получим

$$\theta(k)=\theta(k-1)+\Delta\theta^{*}(k)+\gamma\frac{R_{s-1}(k-1)\nabla f(k)}{\nabla f^{T}(k)R_{s-1}(k-1)\nabla f(k)}f(e,\Delta,k)+$$
$$+\gamma\left(I-\frac{R_{s-1}(k-1)\nabla f(k)\nabla f^{T}(k)}{\nabla f^{T}(k)R_{s-1}(k-1)\nabla f(k)}\right)r_{s-1}(k-1). \qquad (3.4)$$

После умножения обеих частей (3. 4) слева на $\theta^{T}(k)$ имеем

$$\|\theta(k)\|^{2}=\|\theta(k-1)\|^{2}+\|\Delta\theta^{*}(k)\|+2\theta^{T}(k-1)\Delta\theta^{*}(k)+$$
$$+2\gamma\frac{R_{s-1}(k-1)\nabla f(k)}{\nabla f^{T}(k)R_{s-1}(k-1)\nabla f(k)}\times$$
$$\times\left(\tilde{\theta}^{T}(k-1)R_{s-1}(k-1)\nabla f(k)-\gamma\nabla f^{T}(k)R_{s-1}(k-1)r_{s-1}(k-1)\right)f(e(k),\Delta,k)+ \qquad (3.5)$$
$$+\frac{\gamma^{2}}{\nabla f^{T}(k)R_{s-1}(k-1)\nabla f(k)}\left(2\nabla f^{T}(k)r_{s-1}(k-1)-1\right)f^{2}(e,\Delta,k)+$$
$$+2\gamma\tilde{\theta}^{T}(k-1)\left(I-\frac{R_{s-1}(k-1)\nabla f(k)\nabla f^{T}(k)}{\nabla f^{T}(k)R_{s-1}(k-1)\nabla f(k)}\right)r_{s-1}(k-1).$$

Здесь

$$\tilde{\theta}(k-1)=\theta(k-1)+\Delta\theta^{*}(k). \qquad (3.6)$$

Поступая по аналогии с вышеизложенным, нетрудно получить неравенство типа (4.48), в котором коэффициенты ***A, B***, определяются в соответствии с (4.49), (4.50) с заменой в ***B*** вектора $\theta(k-1)$ на $\tilde{\theta}(k-1)$, а коэффициент ***C*** будет содержать дополнительные слагаемые, зависящие от $\Delta\theta^{*}(k)$, т.е.

$$C=2\tilde{\theta}^{T}(k-1)\left(\frac{R_{s-1}(k-1)\nabla f(k)\nabla f^{T}(k)}{\nabla f^{T}(k)R_{s-1}(k-1)\nabla f(k)}-I\right)r_{s-1}(k-1)-$$
$$-\left\|\left(I-\frac{R_{s-1}(k-1)\nabla f(k)\nabla f^{T}(k)}{\nabla f^{T}(k)R_{s-1}(k-1)\nabla f(k)}\right)r_{s-1}(k-1)\right\|^{2}- \qquad (3.7)$$
$$-2\theta^{T}(k-1)\Delta\theta^{*}(k)-\|\Delta\theta^{*}(k)\|^{2}.$$

Приложение И

Математические модели ДА

Уравнение теплообмена для любой зоны сокостружечной смеси получают, составляя уравнение теплового баланса. Например, для второй зоны имеем

$$G_c^{12}C_c^{12}\theta_1 + G_{\partial c}^{32}C_{\partial c}^{32}\theta_3 + kF(\theta_{n2} - \theta_2) = G_c^{23}C_c^{23}\theta_2 + G_{\partial c}^{21}C_{\partial c}^{21}\theta_2, \qquad \text{(И. 1)}$$

где G_c^{ij}, $G_{\partial c}^{ij}$, C_c^{ij}, $C_{\partial c}^{ij}$ – потери и теплоемкости свекловичной стружки, диффузионного сока с i-й в j-ту зону соответственно;

θ_i, θ_{nj} – температуры сокостружечной смеси и пара в паровой камере i-ой зоны соответственно;

k, F – коэффициент теплопередачи и площадь нагрева паровой камеры.

При нарушении баланса изменяется количество тепла в емкости со скоростью, зависящей от дисбаланса

$$MH_2L\rho c_2 \frac{d\theta_2}{d\tau} = \Delta(G_c^{12}C_c^{12}\theta_1 + G_{\partial c}^{32}C_{\partial c}^{23}\theta_3 + kF(\theta_{n2} - \theta_2) - \\ -G_c^{23}C_c^{23}\theta_2 - G_{\partial c}^{21}C_{\partial c}^{21}\theta_2), \qquad \text{(И. 2)}$$

где M, H_i, L, ρ, c_i – ширина, высота и длина аппарата, плотность и теплоемкость сокостружечной смеси соответственно.

В отклонениях переменных с учетом линеаризации уравнение принимает вид

$$\frac{d\theta_2}{d\tau} = \frac{1}{MH_2L\rho c_2}(G_{c_0}^{12}C_c^{12}\Delta\theta_1 + C_c^{12}\theta_{1_0}\Delta G_c^{12} + G_{\partial c_0}^{32}C_{\partial c}^{32}\Delta\theta_3 + C_{\partial c}^{32}\theta_{3_0}\Delta G_{\partial c}^{32} + \\ + kF\Delta\theta_{n2} - kF\Delta\theta_2) - G_{c_0}^{23}C_c^{23}\Delta\theta_2 - C_c^{23}\theta_{2_0}\Delta G_c^{23} - G_{\partial c_0}^{21}C_{\partial c}^{21}\Delta\theta_2 - C_{\partial c}^{21}\theta_{2_0}\Delta G_{\partial c}^{21}. \quad \text{(И. 3)}$$

Для других зон сокоструженой смеси уравнения выводятся аналогично.

Уравнения теплообмена для паровой емкости получают следующим образом. Уравнение теплового баланса имеет вид

$$G_{n1}r_1 = kF(\theta_{n1} - \theta_1), \quad \text{(И. 4)}$$

где G_{ni}, r_i – расход пара в паровой камере и теплота парообразования в i-й зоне ДА.

При нарушении баланса изменяется количество тепла в емкости со скоростью, зависящей от дисбаланса

$$\frac{d\theta_{n1}}{d\tau} = \frac{1}{Vc_{n1}}(r_1\Delta G_{n1} - kF\Delta_{n1} + kF\Delta_1), \quad \text{(И. 5)}$$

где V, c_{ni} – объем паровой камеры и теплоемкость пара в i-й зоне соответственно.

Теплоемкость пара зависит от внутренней энергии пара u_n и его плотности ρ''

$$\tilde{n}_{n1} = u_n a + \rho'' b; \quad \rho'' = \overline{\rho''} + a\theta_n; \quad u_n = \overline{u''} + b\theta_n, \quad \text{(И. 6)}$$

где $\overline{\rho''}, \overline{u''_n}, a, b$ – определяются экспериментально [309, 314].

В [309, 314] показано, что концентрация сахара в стружке и соке описывается экспоненциальной зависимостью от текущей длины

диффузионного аппарата, т.е. расходы и теплоемкости стружки и сока по длине ДА описываются соотношениями

$$
\begin{aligned}
&G_c = 31,25e^{-0.00759l},[кг/с],\\
&G_{дс} = 37,5e^{-0.00623l},[кг/с],\\
&C_c = 3,40e^{-0.00759l},[кДж/(кг\ ^0C)],\\
&C_{дс} = 3,61e^{-0.00623l},[кДж/(кг\ ^0C)].
\end{aligned}
\qquad (И.\ 7)
$$

В результате была получена математическая модель теплообменной части ДУ следующего вида (машинное время τ=100t, где τ, с):

$$
\begin{cases}
\dfrac{d\Delta\theta_1}{d\tau} = \dfrac{1}{MLH_1\rho c_1}(G_{c_0}^{01}C_c^{01}\Delta\theta_c + (C_c^{01}\theta_{0_0} - C_c^{12}\theta_{1_0})\Delta G_c + G_{дc_0}^{21}C_{дc}^{21}\Delta\theta_2 + \\
+(C_{дc}^{21}\theta_{2_0} - C_{дc}^{10}\theta_{1_0})\Delta G_{дc} + kF\Delta\theta_{n1} - (kF + G_{c_0}^{12}C_c^{12} + G_{дc_0}^{10}C_{дc}^{10})\Delta\theta_1;\\
\dfrac{d\Delta\theta_2}{d\tau} = \dfrac{1}{MLH_2\rho c_2}(G_{c_0}^{12}C_c^{12}\Delta\theta_1 + (C_c^{12}\theta_{1_0} - C_c^{23}\theta_{2_0})\Delta G_c + G_{дc_0}^{32}C_{дc}^{32}\Delta\theta_3 + \\
+(C_{дc}^{32}\theta_{3_0} - C_{дc}^{21}\theta_{2_0})\Delta G_{дc} + kF\Delta\theta_{n2} - (kF + G_{c_0}^{23}C_c^{23} + G_{дc_0}^{21}C_{дc}^{21})\Delta\theta_2;\\
\dfrac{d\Delta\theta_3}{d\tau} = \dfrac{1}{MLH_3\rho c_3}(G_{c_0}^{23}C_c^{23}\Delta\theta_2 + (C_c^{23}\theta_{2_0} - C_c^{34}\theta_{3_0})\Delta G_c + G_{дc_0}^{43}C_{дc}^{43}\Delta\theta_4 + \\
+(C_{дc}^{43}\theta_{4_0} - C_{дc}^{32}\theta_{3_0})\Delta G_{дc} + kF\Delta\theta_{n3} - (kF + G_{c_0}^{34}C_c^{34} + G_{дc_0}^{32}C_{дc}^{32})\Delta\theta_3;\\
\dfrac{d\Delta\theta_4}{d\tau} = \dfrac{1}{MLH_4\rho c_4}(G_{c_0}^{34}C_c^{34}\Delta\theta_3 + (C_c^{34}\theta_{3_0} - C_c^{4в}\theta_{4_0})\Delta G_c + G_{б_0}C_б\Delta\theta_б + \\
+G_{ж_0}C_ж\Delta\theta_ж - C_{дc}^{43}\theta_{4_0}\Delta G_{дc} + C_б\theta_{б_0}\Delta G_б + C_ж\theta_{ж_0}\Delta G_ж + kF\Delta\theta_{n4} - \\
-(kF + G_{c_0}^{4в}C_c^{4в} + G_{дc_0}^{43}C_{дc}^{43})\Delta\theta_4;\\
\dfrac{d\Delta\theta_{n1}}{d\tau} = \dfrac{1}{Vc_{n1}}(r_1\Delta G_{n1} - kF\Delta\theta_{n1} + kF\Delta\theta_1);\\
\dfrac{d\Delta\theta_{n2}}{d\tau} = \dfrac{1}{Vc_{n2}}(r_2\Delta G_{n2} - kF\Delta\theta_{n2} + kF\Delta\theta_2);\\
\dfrac{d\Delta\theta_{n3}}{d\tau} = \dfrac{1}{Vc_{n3}}(r_3\Delta G_{n3} - kF\Delta\theta_{n3} + kF\Delta\theta_3);\\
\dfrac{d\Delta\theta_{n4}}{d\tau} = \dfrac{1}{Vc_{n4}}(r_4\Delta G_{n4} - kF\Delta\theta_{n4} + kF\Delta\theta_4);
\end{cases}
\qquad (И.\ 8)
$$

Приложение К

Принцип работы системы АВРПП

Принцип работы системы АВРПП заключается в следующем.

На короткий интервал времени проведения измерения в ванну расплавов руднотермической электропечи вводится электрический зонд, по линии перемещения которого измеряются электрические потенциалы, приводятся полями подведенного для плавки тока и специально сформированным и излучаемым измерительным зондом. Одновременно, измеряются текущие значения вертикальной координаты зонда, которые относятся в соответствии со значением измеренных потенциалов, и строятся графики распределения потенциалов по глубине ванны расплавов печи.

Получаемые графики являются важнейшими обобщающими характеристиками эффективности процесса плавки, контроль которых позволяет существенно повышать качество управления плавкой.

Под управлением ПК в отверстие, предназначенное для ручного измерения уровней расплавов металла и шлака традиционным «ломиком», в ванну расплавов шлаков и металла опускается измерительный зонд - антенна, излучающая и воспринимает электрические сигналы, которые обрабатываются в соответствующих блоках комплекса.

По характеру рассчитанных электронно-вычислительной частью комплекса и сложившихся на экране монитора зависимостей изменения электрического потенциала от координаты перемещения конца измерительного зонда:

- Оценивается структура распределения электрического тока в объеме расплавов шлака и металла, напрямую связана с эффективностью использования электрической подведенной мощности, которая дает возможность контролировать эффективность электрических режимов плавки с целью ее ведения с большей скоростью выхода металла при одинаковых затратах электроэнергии;

- Ориентировочно определяется глубина погружения конца электрода, в зоне которого происходит измерение;

- Определяются нормальные электрические режимы РТП и недопустимы так называемой опасности поломки электродов;

- Скачкообразные изменения полученных графиков соответствуют границам разделов расплавов металла и шлака в ванне печи.

Блок измерения формирует необходимые для измерения сигналы и осуществляет первичную аналоговую обработку электрических сигналов, принятых измерительным зондом (ВЗ), что погружают в ванну руднотермической печи. Принятые электрические сигналы при прохождении измерительного зонда через расплавы металла и шлака отличаются, в соответствии с разницей их электрических характеристик.

Устройство задания режимов и управления измерительным зондом, установленный на отметке «17» (ППР «17») служит для переключения режимов работы: автоматического и полуавтоматического.

В автоматическом (штатном) режиме измерения электрических характеристик расплавов металла и шлака и уровня шихты происходит под управлением программы с отображением результатов измерений на ПК и их регистрацией в его долговременной памяти.

Микропроцессорная система сбора данных и управления кареткой (МСЗД) обеспечивает синхронное считывания данных с блока измерения (БВ), измерения текущей координаты конца ВЗ, управление перемещением ВЗ и передачу этих данных по протоколу RS485 в процессор сбора и передачи в ПК цифровой информации (ПЗПЦИ).

С блока ПЗПЦИ через параллельный (LPT) порт данные передаются в ПК, осуществляющий управление микропроцессорной системой МСЗД, отображения полученных данных на мониторе и их регистрацию в своей долговременной памяти. Программное обеспечение отображает полученные данные на экране монитора в виде графиков зависимости величины электрических параметров расплавов от координаты линии перемещения конца

ЗЧВЗ.

Обозначенный на функциональной схеме (рис.6.34) «Специализированный процессор управления быстрым перемещением измерительного зонда» на структурной схеме обозначен как «Устройство задания режимов и управления измерительным зондом», «Приемник / передатчик цифровых сигналов, согласователь интерфейсов» - как «Процессор сбора и передачи в ПК цифровой информации» . Функциональные блоки: «Специализированный процессор быстрой обработки больших информационных массивов» и «спецпроцессоров формирования текущей координаты положения измерительного зонда» входят в структуру «Микропроцессорные системы сбора данных и управления кареткой». А функциональные блоки: «Приемник / Передатчик», «Генератор специальных частот», «Аналого-цифровые преобразователи» и «Фильтры» входят в структуру «Блок измерения».

Измерение распределения электрических потенциалов в ванной руднотермической электропечи производится на работающей электропечи без ее отключения.

Все результаты измерений отображаются на мониторе и регистрируются в долгосрочной памяти персонального компьютера.

Присоединение к концу измерительного зонда специального пробозаборника позволяет оперативно без отключения печи получать (для экспресс-анализа) пробы расплавов металла и шлака с любой заданной глубины ванны.

Замена базовой погружной части измерительного зонда (ЗЧВЗ) на ЗЧВЗ со встроенным термодатчиком позволяет на работающей печи измерять температуру расплавов металла и шлака и формировать на экране монитора графики зависимости этой температуры от координаты вертикали перемещения термодатчика.

Общее время выполнения измерения - ≤ 80 с.

Точность измерения уровней - ≤ 0,9%.

Абсолютная точность определения изменения уровней (при сливе расплавов металла или шлака или добавки шихты) - ≤ ±0,5 см.

Дискретность измерений величины потенциалов в соседних точках при формировании графика потенциалов - 50 мс., что обеспечивает проведение измерений при передвижении измерительного зонда со скоростью 20 см / с в каждой точке, по линии его перемещения, на расстоянии друг от друга 1 см.

На интервалах, где необходимо формировать графики с большей точностью, например, на интервалах переходов металл / шлак, количество дискретных измерений на единицу длины линии перемещения конца ЗЧВЗ увеличивается за счет снижения скорости перемещения конца ВЗ. Например, измерения на конкретных интервалах линии перемещения конца ЗЧВЗ можно делать в соседних точках, находящихся друг от друга на 1 мм.

Измерения проводятся в любое время и ограничены лишь необходимостью выдержки двадцатиминутных интервалов между следующими измерениями с целью предотвращения перегревов ЗЧВЗ.

Приложение Л

Акты внедрения результатов диссертационной работы

ЗАТВЕРДЖУЮ

Перший проректор Харківського національного університету радіоелектроніки

_____ І.І. Клевчиця

_____ 2016р.

АКТ

про використання результатів дисертаційної роботи
Безсонова Олександра Олександровича

Комісія у складі завідувача кафедри електронних обчислювальних машин професора Рубана І. В., професора кафедри електронних обчислювальних машин Корабльова Н.М., професора кафедри електронних обчислювальних машин Аксак Н.Г., підтверджує, що результати дисертаційної роботи Безсонова Олександра Олександровича «Штучні нейронні мережі прямого розповсюдження, що еволюціонують: архітектури, навчання, використання» впроваджені у навчальний процес на кафедрі електронних обчислювальних машин в курсах «Інтелектуальні комп'ютерні системи», «Системи штучного інтелекту», «Методи та засоби обчислювального інтелекту», «Паралельні та розподілені обчислювання» та «Імунні обчислювальні системи», а також у магістерських дипломних роботах та дипломному проектуванні.

Завідувач кафедри ЕОМ _____ І. В. Рубан

Професор кафедри ЕОМ _____ Н.М. Корабльов

Професор кафедри ЕОМ _____ Н.Г. Аксак

ЗАТВЕРДЖУЮ

Директор ТОВ НВП «Ноосфера»

«__» _____ 2016 р.

АКТ

впровадження наукових результатів дисертаційної роботи
Безсонова Олександра Олександровича

Комісія від науково-виробничого підприємства «Ноосфера» у складі: головного директора, кандидата технічних наук, лауреата Державної премії України в галузі науки і техніки, Сотникова О.М. і членів: старшого наукового співробітника Сотникова А.О. та головного бухгалтера Бабушкіної Г.П. підтверджують впровадження практичних результатів дисертаційної роботи Безсонова О.О. у виробничий процес ТОВ «Побузький феронікелевий комбінат» в межах господарських договорів між ТОВ НВП «Ноосфера» та ТОВ «Побузький феронікелевий комбінат» №12-02Т от 20.04.2012 07-01/30 от 21.06.2007 «Створення автоматизованої системи контролю струму електродів руднотермічної електропечі №4» от 29.03.2004 «Створення автоматизованої системи вимірювання температурних полів №№2,3 от 23.02.2006 «Створення автоматизованої системи вимірювання рівня сировини, металу і шлаку у РТП1» а саме:

- у випалювальному цеху ТОВ «ПФК» у виробничому процесі використовується інфрачервона система контролю температурних полів трубчастої печі, що обертається;
- у металургійному цеху ТОВ «ПФК» у виробничому процесі плавки сировини на феронікель;
- до основи обробляючих блоків цієї системи покладені алгоритми розроблені у дисертаційній роботі Безсонова О.О., що дозволяють виконувати цифрову фільтрацію вхідних масивів даних;
- впроваджені нейромережеві структури розробленої системи інтелектуального управління ходом плавки співавтори його наукових праць;
- впровадження відбувалось у рамках госпдоговорів №№ 12-02Т та 07-01/30 між ТОВ НВП «Ноосфера», що виконує розробку системи автоматизації, та ТОВ «ПФК»;
- розроблені системи забезпечують безперервний контроль температурного поля по довжині трубчастої печі, дозволяє ефективно керувати процесом випалювання сировини, подовжує відведений термін експлуатації футеровки та кожухів печі.

Підписи

О.М. Сотников

А.О. Сотников

Г.П. Бабушкіна

"ЗАТВЕРДЖУЮ"

Проректор з наукової роботи
Харківський національний університет
радіоелектроніки

_____ М. І. Єгоренко

"___" _____ 2009 р.

"ЗАТВЕРДЖУЮ"

Директор ТОВ
"Кириківський цукровий завод"

_____ А.О. Таран

"___" _____ 2009 р.

АКТ

впровадження наукових результатів дисертаційної роботи Безсонова Олександра Олександровича

Комісія від Харківського національного університету радіоелектроніки (далі ХНУРЕ) у складі голови – завідувача кафедрою ЕОМ професора Руденка О.Г. та членів – завідувача лабораторією спеціалізованих цифрових обчислювальних систем провідного співробітника каф. ЕОМ Сотнікова О.М., старшого наукового співробітника каф. ЕОМ Єланова С.Г. та головного інженера ТОВ "Кириківський цукровий завод"– Доля А.І. підтверджують факт впровадження результатів дисертаційної роботи Безсонова Олександра Олександровича у виробничий процес цукрового заводу, а саме:

- у ТОВ "Кириківський цукровий завод" розроблена багаторівнева комп'ютеризована система управління технічним процесом в дифузійному відділенні цукрового заводу;
- розроблена у дисертаційній роботі Безсонова О.О. програмна модель була включена в пакет підпрограм, що використовуються у системі управління дифузійною установкою;
- впроваджено нейромережеві структури, які розроблені особисто Безсоновим О.О., з чим згідні співавтори його наукових праць;
- для регулювання та стабілізації температур у дифузійній установці використовувалися методики та алгоритми, які потребують складних обчислень та значних витрат ресурсів управляючих ЕОМ. Розроблені Безсоновим О.О. підходи дозволяють значно спростити обчислення, що забезпечило поліпшення характеристик роботи дифузійної установки.

Акт складений для пред'явлення до спеціалізованої вченої ради із захисту дисертації і не є підставою для фінансових розрахунків.

Від ХНУРЕ:

_____ О.Г. Руденко
_____ О.М. Сотнікова
_____ С.Г. Єланова

Від ТОВ "Кириківський цукровий завод"

_____ А.І. Доля

ЗАТВЕРДЖУЮ
Директор ТОВ „АТ Співдружність - Т"
_____П.О. Капустенко
"___" _____2016 р.

АКТ

впровадження наукових результатів дисертаційної роботи
аспіранта Безсонова Олександра Олександровича

Комісія товариства з обмеженою відповідальністю „АТ Співдружність- Т" (далі ТОВ „АТ Співдружність - Т") у складі директора Капустенка П.О. – проф. каф. ГПЛА Національного технічного університету "Харківський політехнічний інститут", та членів – заступника директора з виробництва Клочка Є.А., начальника ділянки виробництва контрольно-вимірювальних пристроїв і автоматики Добромислова О.С., провідного інженера з автоматичних систем керування Зоренка В.В., підтверджують факт впровадження результатів дисертаційної роботи Безсонова О.О. у виробничий процес ТОВ „АТ Співдружність - Т", а саме:

- в теплообмінних модулях, що виробляє ТОВ „АТ Співдружність - Т" для комунального господарства, хімічних, харчових виробництв та підприємств енергетики, використовується багатоканальний регулятор витрати теплоносія;
- до основи обробляючого блоку регулятора покладені алгоритми та структури, розроблені у дисертаційній роботі Безсонова О.О., що дозволяють керувати витратами теплоносія за принципами реалізації MISO-моделей;
- впроваджені нейромережеві структури розроблені особисто Безсоновим О.О., з чим згодні співавтори його наукових праць;
- впровадження відбувалось у межах господарчої діяльності ТОВ „АТ Співдружність - Т" (замовники: ПрАТ «Теплоенергетичний центр Роганського промвузла», КП «Харківські теплові мережі»);
- розроблена система регулювання забезпечує безперервний контроль температури в контурі споживання тепла, що дозволяє ефективно керувати витратами теплоносія, що надається від зовнішніх теплових джерел.

Капустенко П.О. _____

Клочок Є.А. _____

Добромислов О.С. _____

Зоренко В.В. _____

ДЕРЖАВНИЙ КОМІТЕТ УКРАЇНИ З ПИТАНЬ ТЕХНІЧНОГО РЕГУЛЮВАННЯ
ТА СПОЖИВЧОЇ ПОЛІТИКИ
ДЕРЖАВНА СИСТЕМА СЕРТИФІКАЦІЇ УкрСЕПРО

Серія ВВ

СЕРТИФІКАТ ВІДПОВІДНОСТІ

UA1.066.0017986-11

Зареєстрований в Реєстрі за №
Зареєстрований в Реєстрі

Термін дії з 19 лютого 2011 до 08 лютого 2015
Строк действия с

Продукція Регулятор температури автоматичний РТГ-2
Продукция

31.20.31.790

Відповідає вимогам ТУ У 24341711.002-2000
Соответствует требованиям

Виробник продукції Акціонерне товариство 'Співдружність-Т', 61002, м. Харків, пр. Червонопрапорний, 2, кв. 19, код ЄДРПОУ 23457736, Україна; місце виробництва: НТУ 'ХПІ', кафедра інтегрованих технологій, процесів і апаратів, 61002, м. Харків, вул. Фрунзе, 21, Україна
Изготовитель продукции

Сертифікат видано Акціонерне товариство 'Співдружність-Т', 61002, м. Харків, пр. Червонопрапорний, 2, кв. 19 , ЄДРПОУ 23457736
Сертификат выдан

Додаткова інформація Регулятор температури автоматичний РТГ-2, який виготовляється серійно з 09.02.2011 р. до 08.02.2015 р., з урахуванням гарантійного терміну зберігання, технічний нагляд один раз на рік
Дополнительная информация

Сертифікат видано органом з сертифікації ОС 'Міжнародні стандарти і системи' - юр. адреса: м. Харків, вул. Тобольська, 42, к. 317; факт. адреса: м. Харків, вул. Весніна, 5, тел. (057) 704-33-69 свідоцтво про призначення № UA.P.066 від 21.06.2010 р., свідоцтво про уповноваження № UA.PN.066 від 21.06.2010 р.
Сертификат выдан органом по сертификации

На підставі Протоколу сертифікаційних випробувань № 10.37.15.280 від 17.02.2011 р., виданого ВЛ 'ВСЦ 'Південтест', 49009, м. Дніпропетровськ, вул. Я. Самарського, 2, атестат акредитації № 2Н485 від 25.10.2010 р. до 24.10.2015 р., акт обстеження виробництва № 066-065 5/11-10 від 09.02.2011 р.
На основании

Керівник органу з сертифікації А.М. Сергейчук
Руководитель органа по сертификации
підпис ініціали, прізвище