БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
Факультет радиофизики и электроники
Кафедра системного анализа

РАЗРАБОТКА АЛГОРИТМОВ И ПРОГРАММНЫХ СРЕДСТВ АНАЛИЗА АТОМАРНЫХ СПЕКТРОВ С ИСПОЛЬЗОВАНИЕМ НЕЙРОННЫХ СЕТЕЙ

Магистерская диссертационная
работа студента магистратуры
Назарова Петра Владимировича

Руководитель:
заведующий кафедрой системного анализа
доктор физ.-мат. наук, профессор Апанасович
Владимир Владимирович

Рецензент:
канд. физ.-мат. наук, доцент
Семенчик Владимир Григорьевич

“Допустить к защите”

Заведующий кафедрой системного анализа

профессор В.В. Апанасович

“___” _____ 2001 г.

МИНСК 2001

РЕФЕРАТ

Диссертационная работа 38 страниц, 19 рисунков, 16 формул, 7 источников.

Ключевые слова: *нейронная сеть, обучение, абсорбционный спектр, внутрирезонаторная лазерная спектроскопия.*

Объектом исследований являются искусственные нейронные сети.

Основная цель работы состоит в разработке и исследовании нейросетевого метода анализа атомно-абсорбционных спектров. Нейросетевые методы имеют ряд преимуществ по сравнению с классическими вариационными, особенно при наличии различных искажений в анализируемых данных. Применение нейронных сетей требует решения ряда специфических проблем на этапе обучения. Некоторые из таких проблем были решены в настоящей работе.

В ходе выполнения диссертационной работы построен алгоритм прямого определения концентраций элементов на основании их абсорбционных спектров. Создан программный продукт, позволяющий производить обработку спектров. Результат работы нейросетевого метода по точности в 1.5 раза превосходит результат работы вариационного метода, основанного на аппроксимации спектров некоторой аналитической функцией. Кроме того, разработана стратегия обучения нейронной сети, позволяющая снизить затраты на создание экспериментального обучающего множества.

СОДЕРЖАНИЕ

СПИСОК ИСПОЛЬЗОВАННЫХ СОКРАЩЕНИЙ

ВРЛС — внутрирезонаторная лазерная спектроскопия;

ВРЛ… — внутрирезонаторный лазерный …;

ИНС — искусственная нейронная сеть;

МСП — многослойный персептрон;

ЛК — лазер на красителях;

СВР — спектрограф высокого разрешения;

ЭСРС — электронная система регистрации спектров;

ЭТА — электротермический атомизатор.

ВВЕДЕНИЕ

В последние десятилетия в мире успешно развивается новая прикладная область математики, специализирующаяся на искусственных нейронных сетях (ИНС). Актуальность исследований в этом направлении подтверждается массой различных применений ИНС. Это автоматизация процессов распознавания образов, адаптивное управление, аппроксимация функционалов, прогнозирование, создание экспертных систем, организация ассоциативной памяти.

Широкий круг задач, в которых работают ИНС, не позволяет в настоящее время создавать универсальные, мощные сети, вынуждая разрабатывать специализированные ИНС, функционирующие по различным алгоритмам.

Целью данной работы являлось создание нейросетевого алгоритма анализа атомно-абсорбционных спектров с целью определения предельно малых концентраций элементов в пробах различного состава. Исходные спектры снимались с помощью внутрирезонаторного лазерного (ВРЛ) спектрометра.

В первой главе работы рассмотрены основы нейросетевого подхода к решению различных задач. Приведена структура используемой ИНС, описан алгоритм её обучения. Также в этой главе описаны проблемы, возникающие перед исследователем при применении нейросетевого подхода к обработке информации.

Вторая глава работы посвящена вопросам внутрирезонаторной лазерной спектроскопии (ВРЛС), являющейся одним из наиболее чувствительных методов детектирования следовых количеств элементов. В этой главе представлен разработанный на основании вариационных подходов алгоритм обработки спектров.

Третья глава целиком посвящена применению ИНС для анализа экспериментальных ВРЛ спектров. В ней приводятся результаты работы ИНС с экспериментальными данными, а также анализируются структура сети и стратегии её обучения.

Работа выполнена с использованием опытного образца прибора, созданного в лаборатории лазерной диагностики плазмы института Молекулярной и атомной физики НАН РБ. Автор выражает признательность старшему научному сотруднику лаборатории ЛДП ИМАФ НАН РБ, кандидату физ.-мат. наук П. Я. Мисакову за предоставленную информацию и помощь в выполнении работы.

ГЛАВА 1. ИСКУССТВЕННЫЕ НЕЙРОННЫЕ СЕТИ

В этой главе будут рассмотрены основы нейросетевого подхода к решению задач анализа данных, описаны структуры и методы обучения искусственных нейронных сетей (ИНС), использованных автором для анализа атомно-абсорбционных спектров.

Нейросетевые методы обработки данных характеризуются высокой гибкостью и устойчивостью к различного рода искажениям информации. Это особенно чётко проявляется при решении обратных задач, каковой является, по сути, задача определения концентраций элементов, на основании их абсорбционных (и эмиссионных) спектров. Применением стандартных методов, основанных на вариационных подходах, зачастую трудно добиться достаточной устойчивости результата к различным искажениям исходных данных [1].

Ситуация кардинально меняется, если обратная задача решается с использованием нейросетевого подхода. Нейросетевые методы принципиально отличаются от вариационных, поскольку они принимают во внимание множество свойств входных данных и, выбирая наиболее важные, снижают ошибку определения искомых параметров [1].

Следует помнить, что практическое применение ИНС требует решения ряда дополнительных проблем, связанных с выбором структуры и построением обучения сети. Эти проблемы решаются в Гл. 3.

1.1. Принцип функционирования и структура ИНС

В самом общем случае ИНС состоит из совокупности простейших вычислительных ячеек – искусственных нейронов, соединённых связями с различными весовыми коэффициентами. Рассмотрим строение и функционирование искусственного нейрона.

1.1.1. Искусственный нейрон

Формальный нейрон состоит из набора межсоединений, сумматора и нелинейного оператора. Значение каждого входа X_i умножается на свой вес W_i, затем суммируется и из суммы вычитается пороговое значение θ. После этого на результат

действует некоторая активационная функция F и преобразованная таким образом информация подаётся на выход нейрона Y (рис. 1.1).

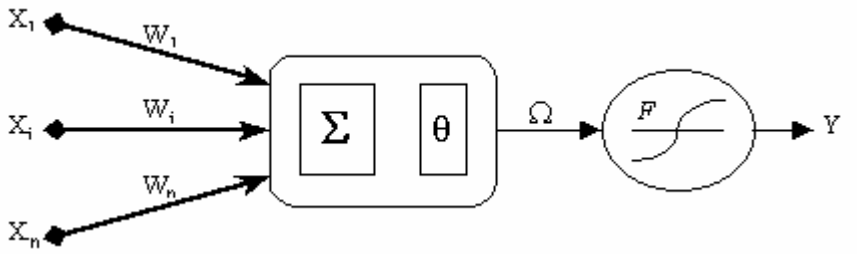


Рис. 1.1. Модель искусственного нейрона.

Таким образом, выход каждого нейрона вычисляется по формуле:

$$Y = F\left\{\sum_{i=1}^{n}\left(X_i W_i\right) - \theta\right\} \tag{1.1}$$

В дальнейшем условимся обозначать сумму произведений $X_i W_i$ через переменную Ω.

Был проведен ряд опытов, результаты которых показали целесообразность использования в выходной функции порога θ, чтобы избавить её от нечётности и симметричности вообще, так как и то и другое влечёт за собой симметричность откликов ИНС и значительно ухудшает её работу.

1.1.2. Влияние активационных функций

В качестве выходных функций нейронов принято использовать ограниченные на множестве значений нелинейные функции, такие как: пороговая (случай сигнума) (рис. 1.2*а*), комбинации пороговой и линейной функций, сигмоидальная функция (рис. 1.2*б*), однотипная с ней функция гиперболического тангенса (рис. 1.2*в*), а также радиальные базисные функции (рис. 1.2*г*).

При использовании пороговой функции (1.2) нейрон остаётся неактивным до тех пор, пока суммарный сигнал с его входов не достигает порогового значения. Когда этот порог достигнут, нейрон возбуждается и даёт на выходе 1.

$$Y = \begin{cases} 0, & \Omega \le \theta \\ 1, & \Omega > \theta \end{cases} \tag{1.2}$$

Пороговая функция наиболее точно моделирует нелинейную передаточную характеристику биологического нейрона, однако вследствие того, что она не дифференцируема, сложные сети довольно редко строятся на её основе.

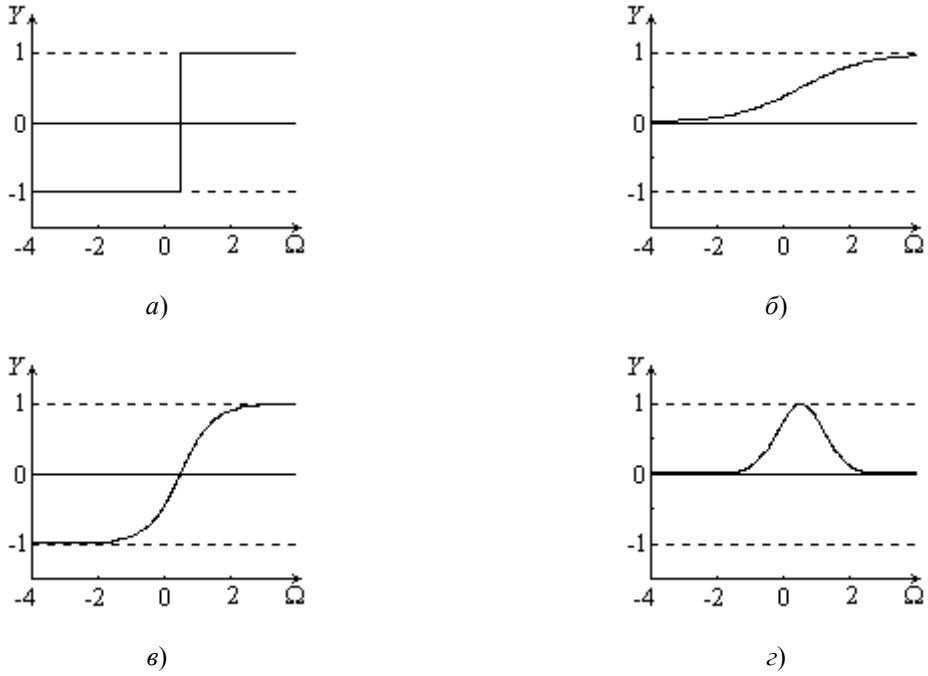


Рис 1.2. Наиболее распространённые виды активационных функций (порог равен 0.5);
а) пороговая; *б*) сигмоидальная; *в*) гиперболический тангенса; *г*) радиальная базисная.

При использовании сигмоидальной (иначе – логистической) функции (1.3) диапазон изменений величины X сужается таким образом, что при любых значениях X значения Y принадлежат некоторому конечному интервалу. Малое изменение $(\Omega - \theta)$ в области нуля приводит к значительному изменению значения функции, тогда как при больших $(\Omega - \theta)$ значение функция практически не зависит от изменения аргумента. Это позволяет избежать негативных ситуаций, связанных с насыщением сети, когда весовые коэффициенты принимают очень большие по модулю значению и процесс обучения становится неустойчивым. К тому же эта функция удобно дифференцируема, что немаловажно в алгоритмах обучения.

$$Y = \frac{1}{1 + e^{-(\Omega - \theta)}}. \tag{1.3}$$

Функция гиперболического тангенса является расширением логистической функции на интервал (-1,1) и имеет вид

$$Y = 1 - \frac{2}{1 + e^{2(\Omega - \theta)}}. \tag{1.4}$$

Весьма интересным и перспективным на сегодняшний день (особенно в вопросах интерполяции и аппроксимации) является класс т.н. *радиальных базисных функций* (РБФ), к которым принято относить функции, имеющие глобальный

экстремум и ведущие себя монотонно по мере удаления от него [2]. Например, к радиальным базисным функциям относится функция вида:

$$Y = e^{-\frac{1}{2}(\Omega-\theta)^2}. \quad (1.5)$$

1.2. Модель и обучение искусственной нейронной сети

Информационная ценность нейронной сети заключается в способе соединения нейронов (модели сети) и перенастраиваемых весовых коэффициентах каждого нейрона. Чтобы заставить сеть работать для решения конкретной задачи, веса настраивают определённым образом. Этот процесс называют обучением сети. Алгоритмы обучения принято разделять на два больших класса: обучение с учителем (при известных целевых выходах сети) и без учителя (при неизвестной выходах). В самом общем случае обучение с учителем представимо в виде следующего алгоритма:

1) Создаётся обучающее множество, то есть упорядоченное множество входов и соответствующих им желаемых выходов. Элемент этого множества называют обучающей парой.

2) Входные данные из обучающих пар подаются на входы сети. Сеть обрабатывает их и выдаёт некоторый результат.

3) Результаты работы сети сравниваются с целевыми значениями, и генерируется вектор отклонений. Если суммарная ошибка меньше некоторой наперёд заданной, то алгоритм прекращает свою работу.

4) Веса сети корректируются с использованием вектора отклонений и, в общем случае, входных и выходных значений сети. Алгоритм переходит к п. 2).

При обучении без учителя, как правило, либо корректируют веса таким образом, чтобы не сильно отличающиеся друг от друга входы сети вызывали похожие выходы, либо сеть сама, в процессе функционирования, перестраивает свои веса по какому либо алгоритму.

1.2.1. Многослойный персептрон

Среди всего многообразия моделей нейронных сетей, следует, пожалуй, выделить класс т.н. многослойных персептронов (МСП), как самый распространённый и изученный. Многослойным персептроном (рис. 1.3) называют нейронную сеть, нейроны в которой расположены слоями, причём нейроны каждого слоя связаны только с нейронами предыдущего слоя.

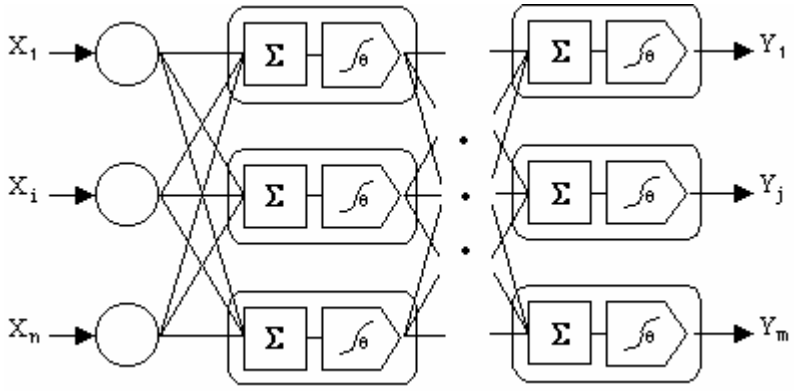


Рис. 1.3. Многослойный персептрон.

Для сетей этого класса существует довольно хороший "классический" метод обучения – т.н. метод обратного распространения ошибки (BPE - *Back Propagation Error*), который относится к разряду методов градиентного спуска. Он сыграл не последнюю роль в пробуждении интереса к нейронным сетям в начале 80-х, а его модификации и по сей день остаются одними из лучших для этого класса сетей.

1.2.2. Метод обратного распространения ошибки

Рассмотрим работу алгоритма обратного распространения ошибки в случае, когда в качестве активационной функции выступает сигмоидальная функция. Путь имеется множество векторов $\{\mathbf{X}\}$, для которых известны желаемые выходы $\{\mathbf{Y}^*\}$ – т.н. целевые вектора. Выход сети, при подаче на её вход вектора **X** обозначим **Y**.

При инициализации сети случайным образом задаются начальные веса и пороги нейронов сети. Алгоритм обучения сети обратного распространения включает в себя следующие шаги:

1. Выбрать очередную обучающую пару $(\mathbf{X},\mathbf{Y}^*)$ из обучающего множества и подать входной вектор **X** на вход сети.
2. Вычислить выход сети **Y**.
3. Вычислить разность между реальным (вычисленным) выходом сети и требуемым выходом (целевым вектором обучающей пары).
4. Подкорректировать веса сети так, чтобы минимизировать ошибку.
5. Повторить шаги с 1 по 4 для каждого вектора обучающего множества до тех пор, пока ошибка на всем множестве не достигнет приемлемой величины.

Шаги 1 и 2 используются как на этапе обучения сети, так и при функционировании уже обученной сети.

Вычисления в сети выполняются послойно. На шаге 3 каждый из выходов сети Y вычитается из соответствующей компоненты целевого вектора с целью получения ошибки. Эта ошибка используется на шаге 4 для коррекции весов сети, причем величина изменений определяются алгоритмом обучения.

Шаги 1 и 2 можно рассматривать как "проход вперед", а 3 и 4 – как проход назад, так как сигнал ошибки распространяется обратно по сети и используется для подстройки весов. Эти два прохода можно выразить математически.

На входе имеем вектор **X**, на основе которого вычисляется выходной вектор **Y**. Вектор **Y** вычитается из целевого вектора $\mathbf{Y}^*$ с целью получения ошибки **ε**:

$$\mathbf{e} = \mathbf{Y}^* - \mathbf{Y}. \tag{1.6}$$

Величина Ω каждого нейрона первого слоя вычисляется как взвешенная сумма входов нейрона. Затем активационная функция F сжимает Ω и дает величину OUT для каждого нейрона в этом слое. Полученное выходное множество **OUT** является входом для следующего слоя. Процесс повторяется слой за слоем, пока не будет получено заключительное множество сети.

На этапе обратного прохода происходит подстройка весов выходного слоя. Так как для каждого нейрона выходного слоя задано целевое значение, то подстройка весов легко осуществляется с помощью дельта правила. Внутренние слои не имеют целевых значений и называются скрытыми слоями.

Рассмотрим процесс подстройки одного веса от нейрона p в скрытом слое j к нейрону q в выходном слое k. Выход OUT слоя k вычитаясь из целевого значения Y^*, дает ошибку, которая умножается на производную сжимающей функции (в нашем случае $OUT(1\text{-}OUT)$), вычисленную для этого нейрона слоя k, давая, таким образом, величину

$$\delta = OUT(1-OUT)(Y^* - OUT) \tag{1.7}$$

Затем δ умножается на величину OUT нейрона j, из которого выходит рассматриваемый вес. Это произведение в свою очередь умножается на коэффициент обучения η (0,01≤η≤1) и результат прибавляется к весу.

$$\Delta w_{pq,k} = \eta \delta_{q,k} \cdot OUT_{p,q}, \tag{1.8}$$

где $\delta_{q,k}$ - величина δ для нейрона q в выходном слое k; $OUT_{p,q}$ - величина выхода для нейрона в скрытом слое j.

$$w_{pq,k}^{(n+1)} = w_{pq,k}^{n} + \Delta w_{pq,k}, \tag{1.9}$$

где $w_{pq,k}^{n}$ – величина веса от нейрона в скрытом слое k к нейрону q в выходном слое на шаге n, $w_{pq,k}^{(n+1)}$ – величина веса на шаге $n+1$ после коррекции. Такая же процедура выполняется для каждого веса от нейрона скрытого слоя к нейрону в выходном слое.

Рассмотрим один нейрон в скрытом слое, предшествующем выходному слою. При проходе вперед этот нейрон передает свой выходной сигнал нейронам в выходном слое через соединяющие их веса. Во время обучения эти веса функционируют в обратном порядке, пропуская величину δ от выходного слоя назад к скрытому слою. Каждый из этих весов умножается на величину δ нейрона, к которому он присоединен в выходном слое. Величина δ, необходимая для нейрона скрытого слоя, получается суммированием всех таких произведений и умножением на производную сжимающей функции:

$$\delta_{p,j} = OUT_{p,j}\left(1 - OUT_{p,j}\right)\sum_{q}\delta_{q,k}w_{pq,k} \tag{1.10}$$

Когда значение δ получено, веса между выходным слоем и скрытым слоем j могут быть скорректированы с помощью формул (3.9) и (3.10), в которых индексы необходимо модифицировать в соответствии со слоем. То есть процесс обучения представляет собой вычисление δ для каждого нейрона в данном слое и коррекцию всех весов данного слоя.

Для ускорения обучения сети вводят нейронное смещение и импульс.

Введение нейронного смещения позволяет сдвигать начало отсчета передаточной функции. Смещение вводится посредством добавления к каждому слою нейронов дополнительного входа, на который подается сигнал, равный +1. В процессе обучения вес данного нейрона корректируется так же, как и остальные веса нейронов.

Введение импульса позволяет ускорить обучение сети при использовании алгоритма обратного распространения. Этот метод заключается в добавлении к коррекции веса члена, пропорционального величине предыдущего изменения веса. Как только происходит коррекция, она запоминается и служит для модификации всех последующих коррекций. Уравнение коррекции принимает следующий вид:

$$w_{pq,k}^{(n+1)} = \alpha \quad \Delta w_{pq,k}^{n} + \eta \quad \left(\delta_{q,k} OUT_{pj}\right) \tag{1.11}$$

Затем вычисляется изменение веса:

$$w_{pq,k}^{n+1} = w_{pq,k}^{n} + \Delta w_{pq,k}^{(n+1)} \tag{1.12}$$

где α - коэффициент импульса, обычно ≈ 0.9 [3].

Описанный метод обучения МСП был применён в данной работе.

1.3. Сложности при обучении и работе с ИНС

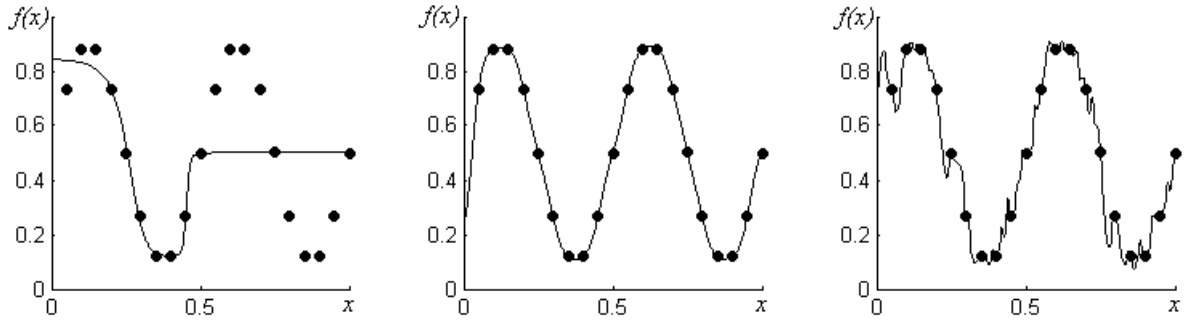
1.3.1. Нелинейность ИНС

При работе с нейронными сетями очень часто возникает вопрос об эффективности структуры сети с точки зрения информационной ёмкости. Понятно, например, что однослойный персептрон способен усвоить меньший объём информации, чем трёхслойный, содержащий к тому же несколько нейронов в каждом скрытом слое. С другой стороны, увеличение количества скрытых слоёв и нейронов в них может быть необоснованным для данной, конкретной задачи и вести, в лучшем случае, к увеличению вычислительных затрат. В худшем случае сеть становится настолько нелинейной системой, что любое малое отклонение её входов от обучающих ведёт к непредсказуемому поведению выходов сети. Таким образом, ИНС теряет одну из своих основных черт – способность к обобщению [2].

Сказанное выше можно проиллюстрировать на следующем примере. Рассмотрим аппроксимацию некоторой функции, например вида

$$f(x) = 0.5 + 0.4\sin(4\pi x), \quad x \in [0,1], \tag{1.13}$$

с помощью трёхслойного персептрона с одним входом и выходом. На вход сети подаются значения x, с выхода снимается соответствующее значение функции. В зависимости от количества нейронов в 1-ом и 2-ом слоях после обучения сети можно получить следующие аппроксимации (рис. 1.4).



а) *г*) *в*)

Рис. 1.4. Результаты аппроксимации функции ИНС с различными количествами нейронов в 1-ом и 2-ом слоях: *а*) по 2; *б*) по 8; *в*) по 50 нейронов.

Рис. 1.4*а* соответствует недостаточной информационной ёмкости (нелинейности) ИНС, рис. 1.4*б* – близкой к оптимальной ёмкости, рис. 1.4*г* избыточной

ёмкости. Точками обозначены обучающие значения функции (1.13), линиями – отклик сети при подаче на вход точек из отрезка [0,1].

Исследования нелинейности ИНС при увеличении скрытых слоёв проводились достаточно давно и их результаты стали уже классикой. В большинстве исследований ИНС рассматривалась как классификатор. Рассмотрим следующую задачу. Пусть дано множество объектов, имеющих *n* признаков. Необходимо обучить ИНС, относить каждый объект к одному из *m* классов на основании этих признаков. Эта задача равносильна разделению точек в n-мерном пространстве на *m*-областей. Было показано, что однослойный персептрон способен разделять объекты в n-мерном пространстве признаков гиперплоскостями, т.е. способен работать только с линейно разделимыми множествами. Двухслойный персептрон работает уже с выпуклыми ограниченными или неограниченными областями, а трёхслойный, при достаточном количестве нейронов, вообще не имеет ограничений при классификации [3].

Несмотря на глубокую теоретическую базу, касающуюся выбора количества слоёв, до сих пор не выработаны общие теоретические принципы, позволяющие находить оптимальное число нейронов в слоях. Поэтому, как правило, это число приходится подбирать эмпирически.

1.3.1. Переобучение ИНС

Вторая проблема, неизменно возникающая при работе с ИНС, связана со сложностью определения критерия останова обучения. Весь процесс обучения ИНС "с нуля" можно условно разбить на три этапа.

1. На первом этапе идёт общая подстройка под задачу, сеть достаточно приблизительно настраивается на величину входных и выходных данных. Этот этап характеризуется весьма высокой скоростью обучения. Если обучающие множества в принципе не реализуемы на такой сети (например, если при использовании логистической активационной функции нейронов выходного слоя пытаться получить на выходе сети значения превышающие 1), процесс обучения дальше не идёт.
2. На втором этапе сеть обучается общим принципам задачи. При обучении сети методом BPE функция ошибки представляет собой ступенчатую кривую. Этот этап наиболее важен для работы сети. Как правило, его результативность сильно зависит от начальных значений весовых коэффициентов ИНС, и по нему с достаточной долей вероятности можно судить о том, удачно ли выбраны исходные веса.

3. Третий этап заключается в адаптации сети к конкретному обучающему множеству. При этом могут "забываться" общие принципы. Строгая настройка сети на конкретную обучающую выборку эквивалентна попаданию обучающего метода в локальный минимум. Результаты работы сети при этом на тестовой выборке могут ухудшаться по мере дальнейшего обучения.

На рис.1.5 проиллюстрировано поведение ошибки при обучении "с нуля" (кривая 1), и при повторном обучении на новых обучающих множествах (кривые 2 и 3).

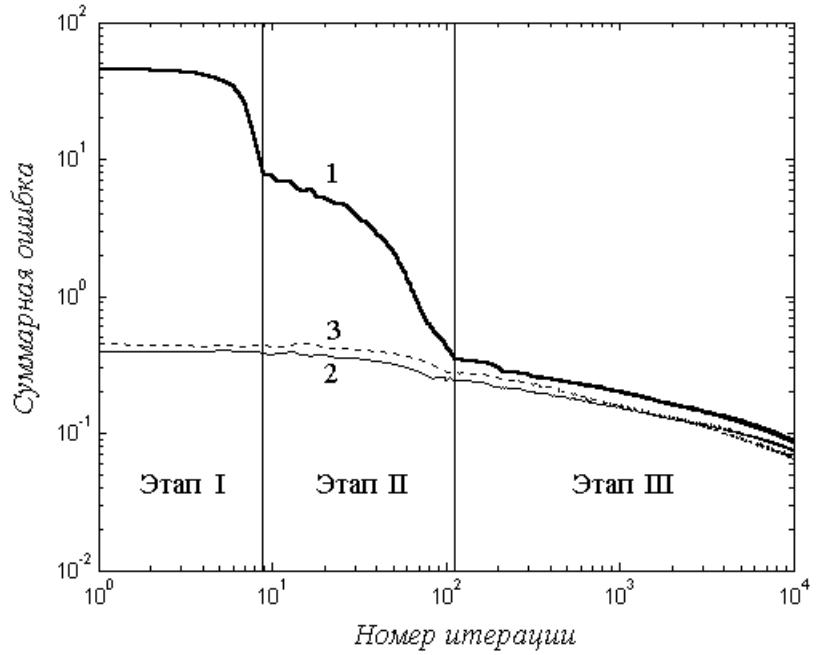


Рис 1.5. Этапы обучения ИНС. Цифрами 1,2 и 3 обозначено поведение ошибки при 1-ом, 2-ом и 3-ем обучении (каждый раз бралась новая обучающая выборка).

Существуют несколько способов, позволяющих избежать или, по крайней мере, снизить негативные последствия переобучения сети. Одним из них является добавления шума к входам на этапе обучения сети. Причём на каждой итерации шумовые искажения должны изменяться. Второй способ состоит во введении кроме обучающего и тестового множеств так называемого контрольного, которое подаётся на сеть после каждой итерации обучения. Если в течение некоторого числа итераций результат работы сети с контрольным множеством не улучшается, обучение прерывают [1].

Обе эти методики применялись при обработке спектров внутрирезонаторного лазерного спектрографа.

ГЛАВА 2. ВНУТРИРЕЗОНАТОРНАЯ ЛАЗЕРНАЯ СПЕКТРОСКОПИЯ

Прежде чем приступать к описанию методов и алгоритмов анализа атомно-абсорбционных ВРЛ спектров необходимо рассмотреть методику их получения, факторы, определяющие их форму и т.д. Эта глава полностью посвящена ВРЛ спектроскопии. В ней приводится описание экспериментальной установки, рассматривается модель абсорбционного спектра и различные негативные факторы, усложняющие анализ спектров. Кроме того, в п.2.4 приведены методы обработки спектров, основанные на вариационном подходе.

2.1. Применение лазеров для целей спектроскопии и преимущества ВРЛС

Широкое распространение лазерной спектроскопии обусловлено в настоящее время также всё более возрастающими метрологическими требованиями к методам и средствам контроля химического состава веществ при производстве сверхчистых материалов, мониторинге окружающей среды и других исследованиях в различных областях современной науки и техники. Поскольку уровень чувствительности традиционных аналитических способов и приборов в основном достиг своего принципиального предела, либо близок к нему, то актуальным и перспективным научным и научно-техническим вопросом аналитической и диагностической спектрометрии стала разработка, оптимизация и последующее практическое использование новых высокочувствительных и экспрессных лазерно-спектроскопических методов и соответствующего аппаратного и программного обеспечения.

Среди многочисленных методов лазерной спектроскопии наиболее эффективной и информативной является внутрирезонаторная лазерная спектроскопия [4]. Она основана на высокой чувствительности интенсивности излучения лазера к малым потерям, размещенным в его резонаторе, и выделяется среди других лазерно-спектроскопических методов сочетанием высокой чувствительности детектирования поглощающих центров со сравнительной простотой экспериментальной реализации и эксплуатации технических средств [4].

Интенсивное развитие ВРЛС и её широкое применение для решения аналитических, спектроскопических, диагностических и других научных и прикладных задач обусловлено рядом очевидных преимуществ этого метода.

1. Чувствительность внутрирезонаторных лазерных (ВРЛ) спектрометров на три-четыре порядка выше, чем других атомно-абсорбционных.
2. Использование в качестве генерирующих сред различных органических красителей (кумарины, родамины и др.), а также их смесей, позволяет варьировать диапазон работы спектрометра от ~400 нм до ~1000 нм, что делает возможным определение концентраций большого количества элементов.
3. Высокое спектральное разрешение. Межмодовое расстояние лазера весьма мало, что обеспечивает практически сплошной спектр в области поиска абсорбционной линии.
4. Высокое временное разрешение, особенно при лазерной накачке, позволяет анализировать короткоживущие радикалы, различные быстро протекающие физико-химические процессы и т.д.

2.2. ВРЛ спектрометр Минск-2

В настоящей работе экспериментальные данные были получены на ВРЛ спектрометре Минск-2 (рис. 2.1), блок-схема которого приведена на рис. 2.2.

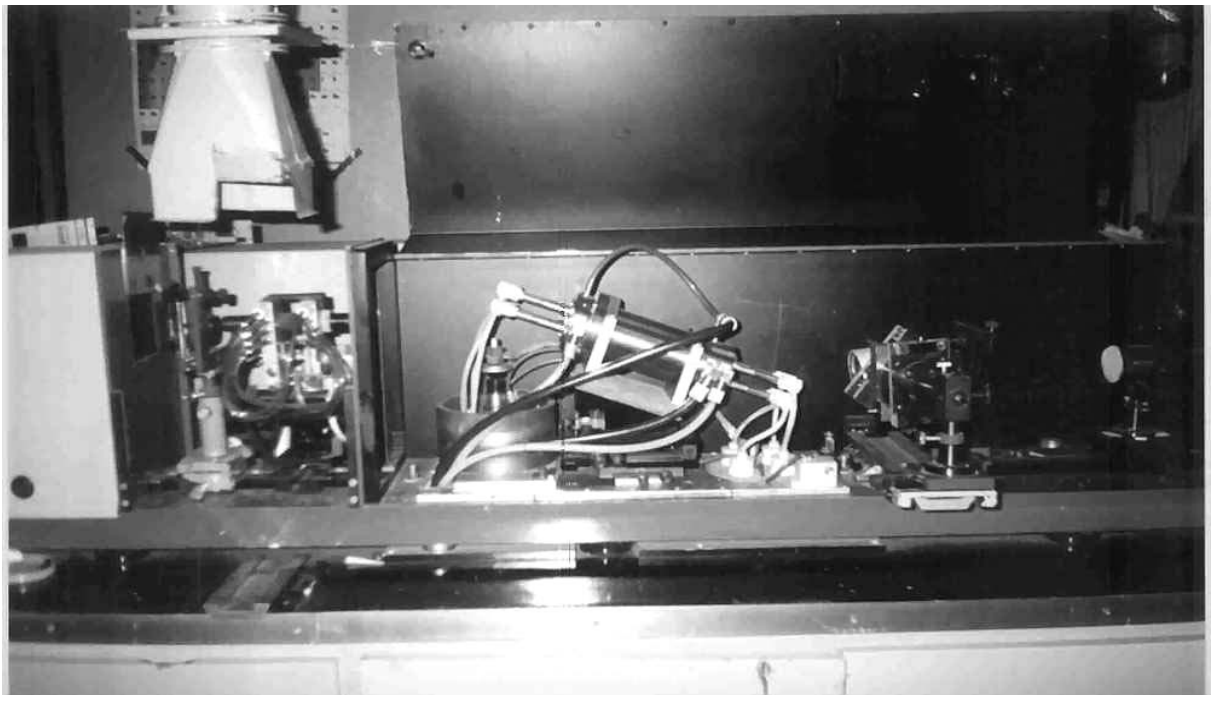


Рис.2.1. ВРЛ спектрометр Минск-2.

Спектрометр представляет из себя сложную измерительную систему, состоящую из четырёх основных модулей. Этими модулями являются: лазер на красителях (ЛК) с ламповой накачкой, электротермический атомизатор (ЭТА), спектрограф высокого разрешения (СВР) и электронная система регистрации спектров (ЭСРС).

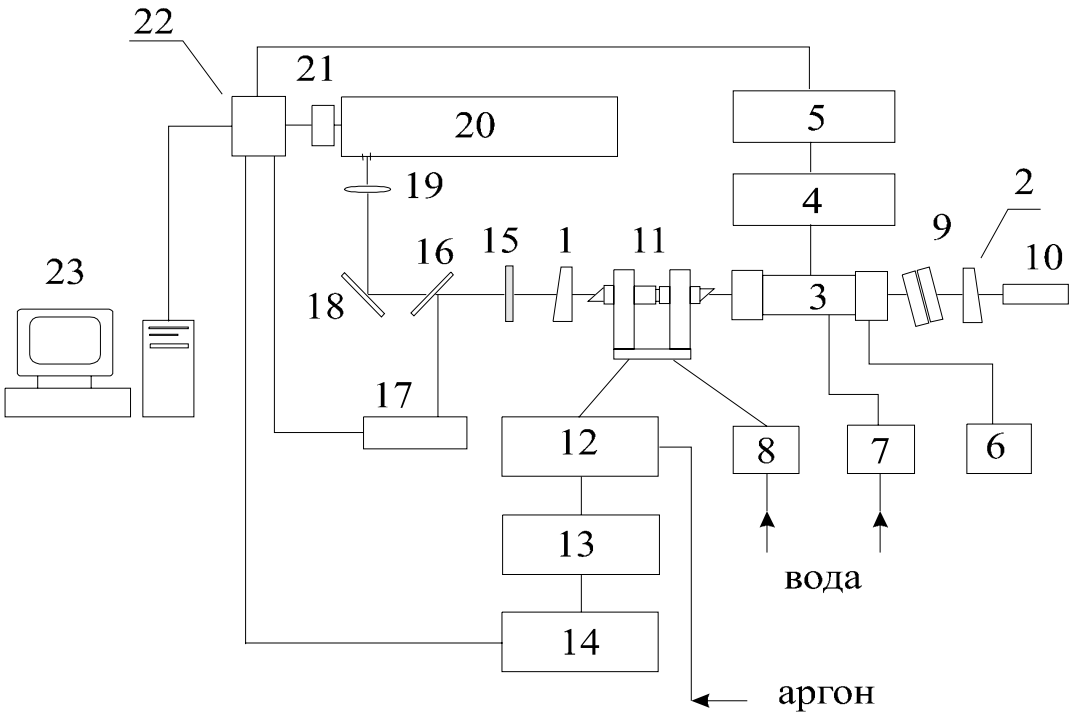


Рис.2.2. Блок-схема спектрометра:

1,2 - зеркала резонатора, 3 - осветитель с кюветой, содержащей генерирующий раствор красителя, 4 - батарея конденсаторов, 5 - блок питания и управления, 6 - блок прокачки растворов красителя, 7 – блок охлаждения осветителя, 8 - блок охлаждения атомизатора, 9 - интерферометр Фабри-Перо, 10 - гелий-неоновый лазер, 11 - ЭТА, 12 - блок подачи газа, 13 - блок питания атомизатора, 14 - блок программируемого нагрева, 15 - ослабитель излучения лазера, 16 - светоделитель, 17 - фотодиод, 18 - поворотное зеркало, 19 - цилиндрическая линза, 20 - СВР, 21,22 - ЭСРС со схемой питания и управления, 23 - персональная ЭВМ.

2.2.1. Лазер на красителях с ламповой накачкой

Резонатор ЛК с базой 700 мм был образован зеркалами 1 и 2 с коэффициентами отражения 90 и 99,9 %, соответственно. Генерирующий раствор красителя прокачивался через кювету внутренним диаметром 6 мм и длиной 120 мм с помощью блока прокачки 6. Торцы кюветы были срезаны под углом Брюстера, что обеспечивало

вместе с клиновидными подложками зеркал резонатора слабую выраженность интерференционной структуры в спектре генерации. Накачка ЛК осуществлялась четырьмя лампами типа ИНП2-7/120. Электрическое питание ламп осуществлялось блоком 5. Блок питания обеспечивал подачу на каждую лампу стабилизированных импульсов напряжением 6-20 кВ, энергией 20-220 Дж. В зависимости от параметров импульсов накачки длительность импульсов генерации изменялась в пределах 1-10 мкс. Спектральный диапазон излучения ЛК определялся типом красителя и составлял в проведённых экспериментах 420-600 нм.

Информация о поглощающей среде в случае применения метода внутрирезонаторной лазерной спектроскопии содержится в широкополосных спектрах излучения ЛК, в которых наблюдается ослабление генерируемого излучения в отдельных узких спектральных интервалах, обусловленных линиями поглощения анализируемого вещества, внесенного в резонатор лазера. Для стабилизации спектра генерации и интенсивности излучения внутрь резонатора был установлен интерферометр Фабри-Перо 9 с базой 5 мкм без отражающего покрытия. Ширина спектра генерации при этом существенно не уменьшалась и составляла 3 - 4 нм. Это позволяло стабилизировать положение и спектральную форму импульса генерации, однако нестабильность интенсивности генерации полностью не исчезала [5].

2.2.2. Электротермический атомизатор

Одним из наиболее важных устройств, определяющих чувствительность, правильность и точность анализов, проводимых на атомно-абсорбционных и ВРЛ спектрометрах, является атомизатор. Идеальный атомизатор представляет собой устройство, которое переводит в атомный пар все атомы исследуемого вещества и локализует это облако вдоль оси зондирующего пучка света. Причем облако локализованных паров должно существовать в атомизаторе все время, пока идет измерение и количество атомов при одинаковых условиях приготовления растворов, при одной концентрации, при одинаковой дозировке и при одном и том же режиме нагрева атомизатора должно воспроизводиться с большой точностью. Наиболее близок к указанным характеристикам электротермический атомизатор (ЭТА).

ЭТА (11) с графитовой трубчатой печью, который использовался в данной работе, являлся стандартным атомизирующим комплексом “Графит-2”. Он используется в атомно-абсорбционных спектрометрах типа “Сатурн-2” и его модификациях. Воспроизводимость количества атомов атомизируемого вещества в нем

известна и составляет величину ± 5 % от измеряемой концентрации. Данный атомизатор позволяет выполнять необходимые для работы программы нагрева с общим количеством шагов, равным 64. Интервал регулировки температуры от 40^0C до 3070^0C. Водные растворы исследуемых образцов вводились внутрь трубчатой графитовой печи атомизатора с помощью микродозатора на 20 - 50 μл.

Основными источниками ошибок атомизатора являются: 1) ошибки дозировки, 2) ошибки электронного программатора. Эти ошибки проявляются в неточностях установки нужной температуры атомизатора, а также в невоспроизводимости формы импульса абсорбции. Следует отметить, что вклад в суммарную погрешность, вносимый атомизатором, меньше, чем погрешность, обусловленная невоспроизводимостью интенсивности и формы спектра генерации лазера [5].

2.2.3. Спектрограф высокого разрешения

Спектрограф высокого разрешения (СВР) (20) позволяет регистрировать узкие линии поглощения в широкополосном спектре генерации ЛК. Получить достаточно высокое спектральное разрешение (0,003 нм) позволило применение в качестве диспергирующего элемента эшелле (300 штр/мм), работающего в высоких порядках спектра и оптической схемы с двукратной дисперсией при сравнительно небольшом фокусном расстоянии (1377 мм). Сканирование спектра производилось поворотом эшелле с помощью шагового двигателя по определенной программе [5].

2.2.4. Электронная система регистрации спектров

Электронная система регистрации спектров (ЭСРС) (21) предназначена для регистрации спектров на выходе СВР. Она превращает световое изображение спектра в электрические сигналы. В качестве детектирующего элемента в ЭСРС используется линейка ПЗС. Сигналы преобразуются в цифровую форму (10-ти разрядный АЦП) и передаются в ЭВМ для математической обработки и вывода данных. Все управление получением информации, ее транспортировкой, хранением, отображением и обработкой осуществляется персональной ЭВМ [5].

2.3. Атомно-абсорбционные спектры и сложности при их обработке

На рис. 2.3 представлен необработанный спектр с ярко выраженной абсорбционной линией Cs 455.531 нм.

Вследствие того, что в гл. 3 будут играть немаловажную роль искусственно сгенерированные спектры, необходимо подробно рассмотреть факторы, оказывающие значительное влияние на их форму.

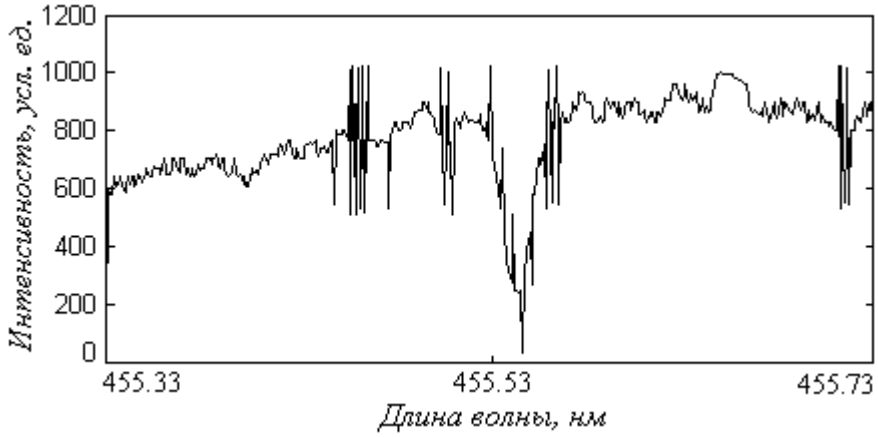


Рис. 2.3. Необработанный спектр с абсорбционной линией.

2.3.1. Спектральный контур линии генерации

Как уже отмечалось выше, спектр генерации лазера на красителях не отличается временной устойчивостью. В значительной степени это связано со сложностью создания ламинарного потока спиртового раствора красителя в водоохлаждаемой кювете. Во время импульса накачки повышается температура потока генерирующего раствора, что приводит к появлению неоднородностей, рассеивающих излучение и ведущих к возникновению или угасанию мод.

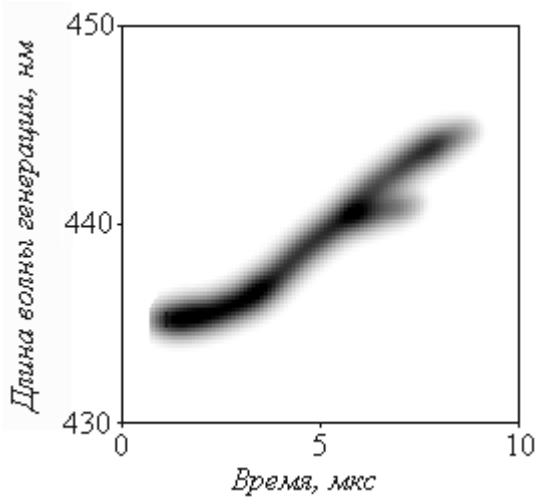


Рис. 2.4. Пример поведения импульса генерации лазера на красителях (не экспериментальные данные).

Рис. 2.4 иллюстрирует возможное поведение генерации лазера в спектрально-временных координатах. Этот рисунок представляет собой не экспериментальные данные, а их обобщение.

В проведённых опытах ширина линии генерации лазера колебалась в пределах 3-6 нм. В дальнейшем условимся называть идеализированную спектральную форму импульса генерации лазера – базовой линией. Фиксируемая ПЗС-линейкой область спектра составляла порядка 0.5 нм, что обеспечивало достаточную гладкость базовой линии.

2.3.2. Интерференционные составляющие

Введение в резонатор интерферометра Фабри-Перо вызывает, наряду со стабилизацией спектра, появление интерференционных составляющих. Вследствие невоспроизводимости спектральной формы импульса их анализ и устранение становятся невозможны. Интерференционные составляющие влияют на точность выделения базовой линии и приводят к маскировке абсорбционного пика при малых концентрациях анализируемого элемента.

2.3.3. Шумы ПЗС-линейки

При регистрации спектра ПЗС-линейкой в сигнале неизбежно появляются мультипликативные шумы. Эти шумы достаточно высокочастотны, что позволяет без труда удалить их из сигнала путём низкочастотной цифровой фильтрации.

2.3.4. Наводки в АЦП системы регистрации

При разряде батареи конденсаторов в блоке регистрации имеют место значительные наводки, что приводит к сбоям в работе АЦП. Как следствие этого в выходном сигнале появлялются δ-импульсы с амплитудой ±256 единиц, что составляет порядка 0.25 динамического диапазона.

2.3.5. Абсорбционная линия

Для большого числа элементов аналитическая абсорбционная линия близка по форме к функции Гаусса (допплеровское уширение). Однако, аналитические линии некоторых элементов, например, йода, имеют сдвоенную структуру, что необходимо всегда иметь в виду при обработке спектров [6].

Вследствие погрешностей механики монохроматора, при настройке на рабочую длину волны с каждым новым включении прибора положение абсорбционной линии одного и того же элемента изменяется.

2.3.6. Методика классического анализа атомно-абсорбционных спектров

Определение неизвестных концентраций элементов в пробах различного состава с помощью ИСП-спектрометра проводится методом построения калибровочных кривых.

В работах [5] и [7] в качестве концентрационно-зависимых параметров спектра использовались относительная глубина провала $\Delta I/I_0$ и эквивалентная ширина $\Delta S/I_0$, физический смысл которых можно понять из рис. 2.5.

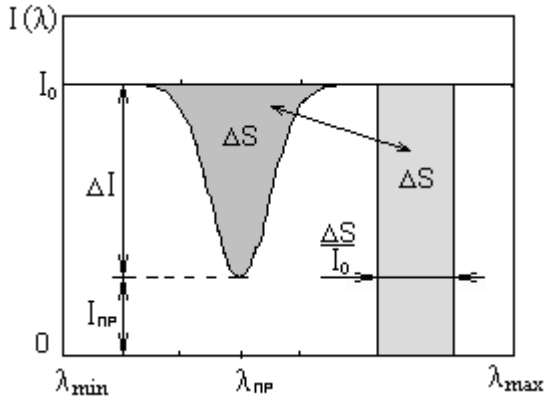


Рис. 2.5. Параметры абсорбционного спектра.

Следует отметить, что величины $\Delta I/I_0$ и $\Delta S/I_0$ неудобны для построения калибровочных графиков, а значит и для практического применения, поскольку они не линейно зависят от концентрации поглощающего элемента.

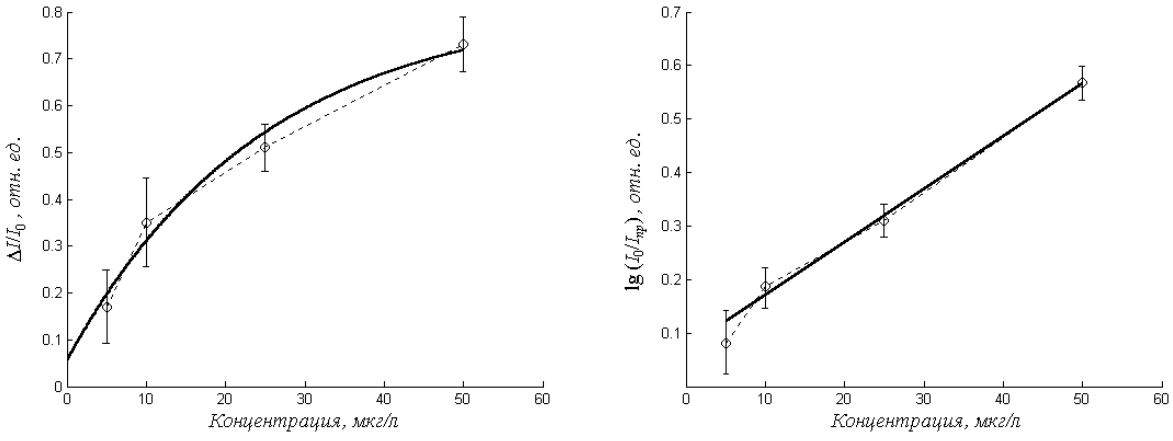
Для того, чтобы найти более удобную величину рассмотрим поглощение излучения некоторым веществом с концентрацией C и длиной поглощающей области L. Интенсивность излучения, прошедшего такое вещество, связана с интенсивностью падающего излучения известным соотношением Лабмерта-Бера:

$$I = I_0 e^{-kCL}, \qquad (2.1)$$

где I – интенсивность прошедшего излучения, I_0 – интенсивность падающего излучения, k – некоторая константа, связанная с поглощающими свойствами вещества. Преобразовав (2.1) получаем

$$\lg\left(\frac{I}{I_0}\right) = -kCL\,, \text{ или } \lg\left(\frac{I_0}{I}\right) \propto C \qquad (2.2)$$

Таким образом мы получили характеристику излучения, прямо пропорциональную концентрации поглощающего вещества. В случае атомно-абсорбционной спектроскопии в качестве I следует брать интенсивность излучения в области абсорбционного провала $I_{пр}$ (см. рис. 2.5) . На рис. 2.6 представлены два калибровочных графика, построенные по одним и тем же данным, но с использованием различных концентрационно-зависимых мер.



а) *б*)

Рис. 2.6. Примеры калибровочных графиков для цезия при использовании в качестве концентрационно-зависимой меры $\Delta I/I_0$ (*а*) и $\lg(I_0/I_{пр})$ (*б*).

2.4. Вариационные методы обработки

2.4.1. Предварительная обработка спектров

Прежде чем приступить к описанию алгоритмов анализа спектров, следует рассмотреть методы предварительной обработки экспериментальных данных. Такие методы позволяют значительно снизить погрешности измерения концентраций [7].

Градиентный анализ. Вначале необходимо избавиться от шумовых выбросов, обусловленных наводками АЦП системы регистрации. Эти шумовые отсчёты не несут полезной информации, вследствие чего могут быть, без вреда для дальнейшей обработки, заменены на усреднённые значения соседних отсчётов.

Фильтрация. Фильтрация, призванная избавить спектр от малоинформативных высокочастотных шумов ПЗС линейки, производилась следующим образом.

Вычислялось преобразование Фурье от спектра. Его действительная и мнимая части умножались на окно специальной формы, уменьшающее влияние высокочастотных шумов. Было исследовано несколько отсекающих окон, таких как: прямоугольное, треугольное, косинусное, гауссово. Наименьшие погрешности дало использование окна гауссовой формы. На последнем этапе фильтрации от результата вычислялось обратное преобразование Фурье.

Ниже рассмотрены два вариационных метода экстракции параметров абсорбционных спектров, описанные в [4] и [7]. Методы тестировались на 74 калибровочных спектрах цезия при концентрациях 5, 10, 25, 50 μг/л и 61 тестовом спектре, снятом при постоянной концентрации Cs – 25 μг/л. В качестве аналитической линии Cs была выбрана линия 455.531 нм.

2.4.2. Метод 1 – прямой поиск

Аппроксимация линии лазера. Для выяснения значения базовой линии в области провала спектр аппроксимировался полиномом 3-го порядка. Аппроксимация проводилась методом локальных вариаций. Выбор порядка полинома связан с тем, что сглаженный экспериментальный спектр имеет, как правило, одну точку перегиба (изменения знака кривизны). Для аппроксимации такого спектра необходимо использовать полином как минимум третьего порядка, поскольку вторая производная аппроксимирующей функции должна быть знакопеременной.

Расчёт параметров проводился напрямую по экспериментальным данным. В заданном диапазоне искался глобальный минимум, расстояние от минимума до базовой линии – ΔI, а площадь между ней и спектром – ΔS.

В таблице 2.1 и на рис. 2.7 представлены результаты обработки спектров калибровочных концентраций.

Таблица 2.1. Результат работы метода 1 для калибровочных концентраций Cs.

Параметр	**Значения по концентрациям и относительное СКО**			
	5 μг/л	**10 μг/л**	**25 μг/л**	**50 μг/л**
$\lg(I_0/I_{пр})$	0.08 (±38%)	0.19 (±15%)	0.31 (±11%)	0.56 (±6%)

Метод был применён для анализа 61 спектра Cs при концентрации 25 μг/л. Относительное среднеквадратическое отклонение определяемых концентраций составило 11 %.

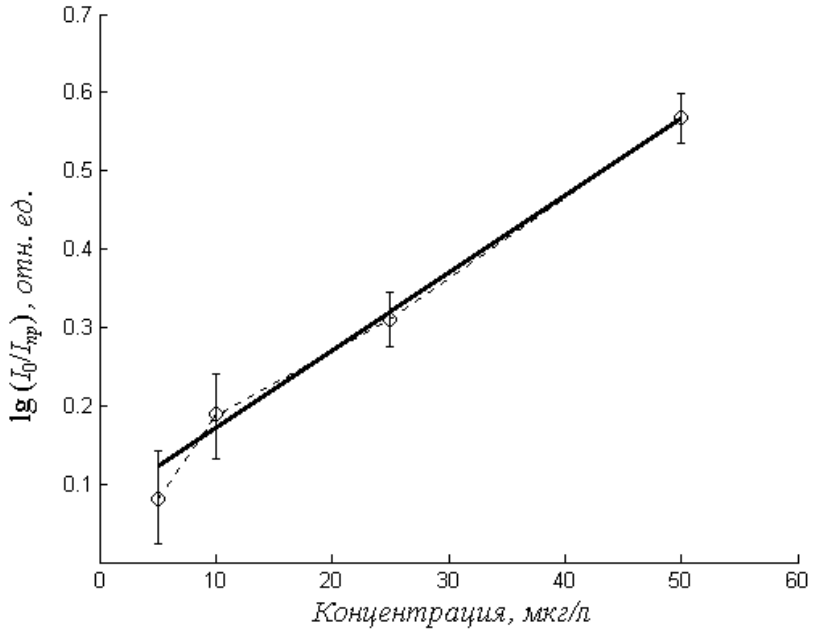


Рис. 2.7. Калибровочный график для Cs, полученный при помощи метода 1.

2.4.3. Метод 2 – аппроксимация спектров

Аппроксимация линии лазера. Так же как и в предыдущем случае, для выяснения значения базовой линии спектр аппроксимировался полиномом 3-го порядка.

Поиск положения провала. Приблизительное положение провала искалось путём построения функций корреляции между некоторым эталоном (например – функцией Гаусса) и данными, полученными при вычитании из аппроксимирующей линии исходного спектра. Найдя по максимуму корреляционной функции приближённое положение провала, и зная приблизительную ширину, можно аппроксимировать спектр функцией

$$f(\lambda) = \sum_{i=0}^{3} a_i \lambda^i - \Delta I \cdot \exp\left[-\left(\frac{\lambda - \lambda_{np}}{\Delta\lambda} \right)^2 \right], \qquad (2.3)$$

где a_i – коэффициенты полинома, ΔI – величина провала, λ_{np} – положение провала, $\Delta\lambda$ – коэффициент ширины абсорбционной линии. Аппроксимация строится по параметром a_i, ΔI, λ_{np}, $\Delta\lambda$. Она даёт достаточно точный результат и не встречает трудностей при

реализации, поскольку заранее были вычислены достаточно точные оценки всех коэффициентов. Следует отметить, что здесь подразумевается наличие одной явной абсорбционной линии. Для значительного количества анализируемых веществ это выполняется. На рис. 2.8 представлен скорректированный спектр.

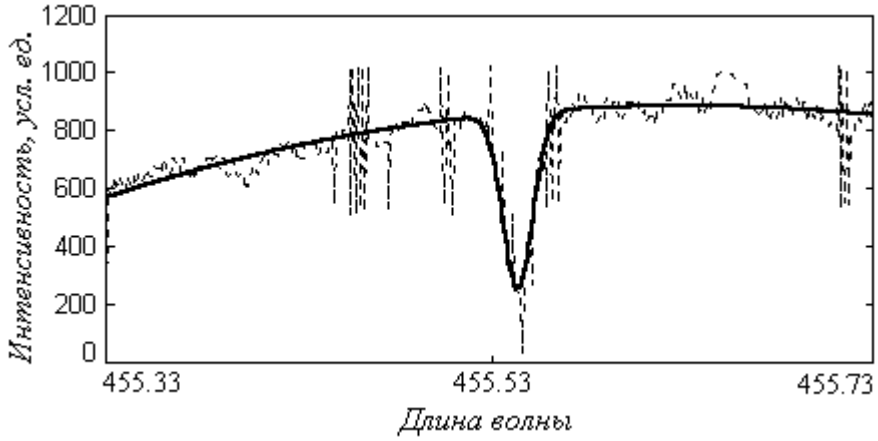


Рис. 2.8. Аппроксимация абсорбционного спектра.

Описанный метод был протестирован на спектрах Cs при концентрации 25μг/л. Относительная среднеквадратическая ошибка при нахождении этой концентрации составила 9.1%, что несколько меньше, чем в предыдущем методе. Алгоритм устойчив к сдвигам провала, к шумам ПЗС и блока регистрации. К недостаткам метода можно отнести его ресурсоёмкость. Так, время обработки одного спектра на машинах класса Pentium II составляет порядка 5с. В табл. 2.2 приведены калибровочные значения параметра абсорбции, а на рис. 2.9 представлен калибровочный график для линии Cs.

Таблица 2.2. Результат работы метода 2 для калибровочных концентраций Cs.

Параметр	**Значения по концентрациям и относительное СКО**			
	5 μг/л	**10 μг/л**	**25 μг/л**	**50 μг/л**
$\lg(I_0/I_{пр})$	0.1 (±20%)	0.19 (±12%)	0.31 (±10%)	0.57 (±6%)

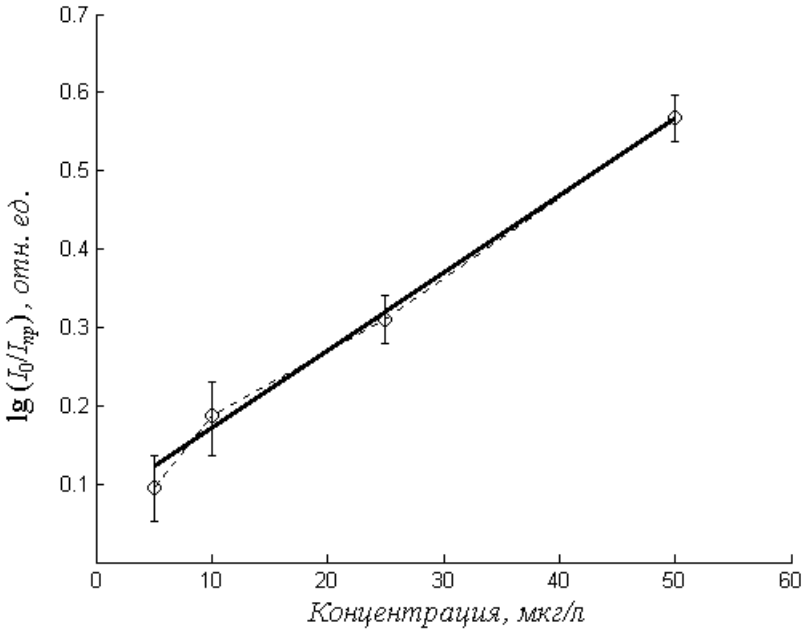


Рис. 2.9. Калибровочный график для Cs, полученный методом 2.

Таким образом, в этой главе были рассмотрены основы ВРЛС и стандартных методов обработки спектров.

При работе с экспериментальными данными было замечено, что результаты автоматической обработки спектров описанными алгоритмами значительно хуже тех, что получает опытный оператор, обрабатывающий спектры "вручную". Это заставило искать новые подходы к анализу спектров.

ГЛАВА 3. НЕЙРОСЕТЕВАЯ ОБРАБОТКА СПЕКТРОВ

В этой главе описан разработанный алгоритм анализа атомно-абсорбционных спектров, основанный на нейросетевом подходе. Для анализа спектров были испробованы несколько классов ИНС: сети с радиальными базисными функциями (РБФ-сети), линейные сети и многослойные персептроны. Лучшие результаты на практике показали многослойные персептроны и метод прямого определения концентрации. Именно этому методу и посвящена данная глава.

3.1. Прямое определение концентрации

Метод прямого определения концентраций состоит в следующем. На вход сети подаётся нормированный спектр, с выхода снимается концентрация, или величина, для которой существует простое аналитическое выражение, переводящее её в концентрацию (например, логарифм концентрации).

Поскольку в опытах ширина фиксируемого ПЗС-линейкой спектра составляла ~0.5 нм, а абсорбционный провал имел ширину ~0.05 нм, то на вход сети подавалась только центральная, наиболее информативная часть спектра (300 отсчётов).

Значения отсчётов исходных спектров принадлежали отрезку [0,1023], тогда как ИНС успешно работает лишь с величинами порядка 10^{-2}–10^{1}. Поэтому отсчёты исходных спектров переводились в диапазон [0,1) путём деления на 1024.

Несколько сложнее обстояло дело с выходами сети. Концентрация некоторых элементов могла иметь значительный динамический диапазон (10^{-6} – 10^{-3} г/л), тогда как значение на выходе сети находится в диапазоне (0,1). Было предложено несколько способов, позволяющих обойти это препятствие:

1. Перевод всего диапазона концентраций линейными преобразованиями в интервал (0,1). Этот способ хорошо работал при небольшом динамическом диапазоне, когда различие между крайними концентрациями составляло 1 десятичный порядок, например для цезия с концентрациями 5 – 50 мг/л. В результате проведенных экспериментов было установлено, что для придания методу большей эффективности диапазон концентраций следует переводить в отрезок [0.2, 0.8].
2. Использование логарифма концентрации. Способ позволял работать с большим диапазоном концентраций, однако приводил к росту абсолютных погрешностей при

высоких концентрациях, в результате чего уровень относительной ошибки не уменьшался, а зачастую и увеличивался с ростом концентрации.

3. Использование нескольких выходов. В случае широкого динамического диапазона концентраций можно использовать несколько выходов, с которых снимать, например, десятичные разряды концентрации. Однако этот способ приводил к усложнению сети и росту её нелинейности.

Поскольку метод тестировался на спектрах Cs с концентрациями 5–50 мкг/л, было решено использовать линейный перевод концентраций в отрезок [0.2, 0.8].

Необходимо отметить, что эффективность метода значительно повышалась, если на вход сети подавались спектры, подвергнутые предварительной обработке (градиентный анализ и фильтрация), описанной в п.1.4.1.

3.2. Стратегии обучения

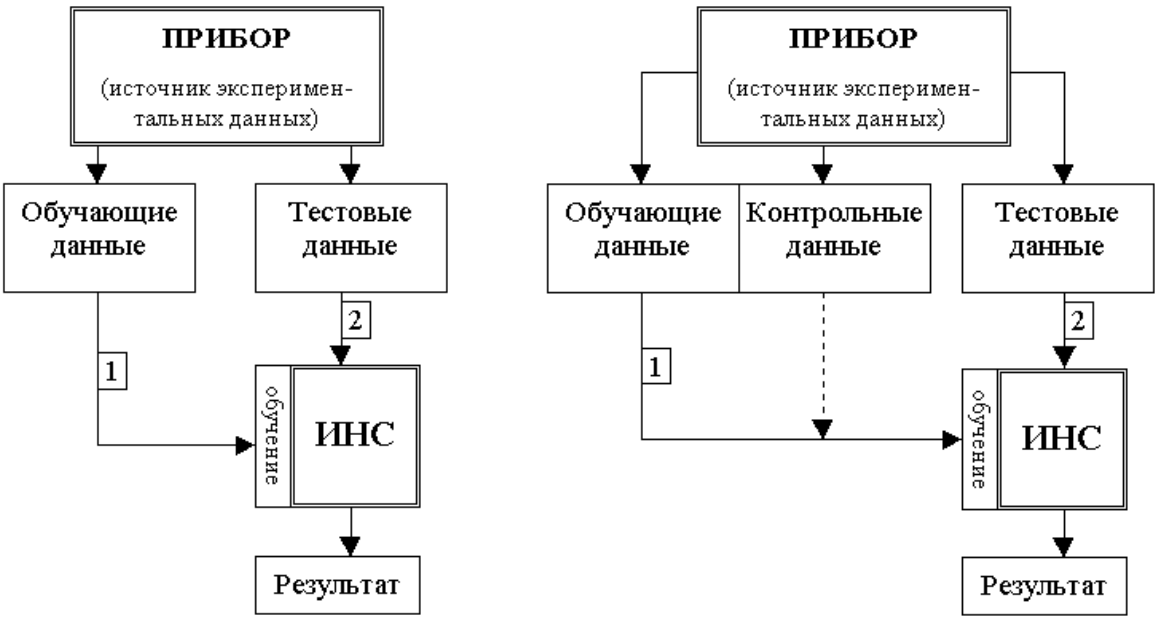
Одним из важнейших этапов работы с ИНС, без которого в принципе не возможно её дальнейшее использование, является обучение сети. При этом можно предложить несколько способов (стратегий) обучения, рассмотренных ниже.

3.2.1. Стандартное обучение

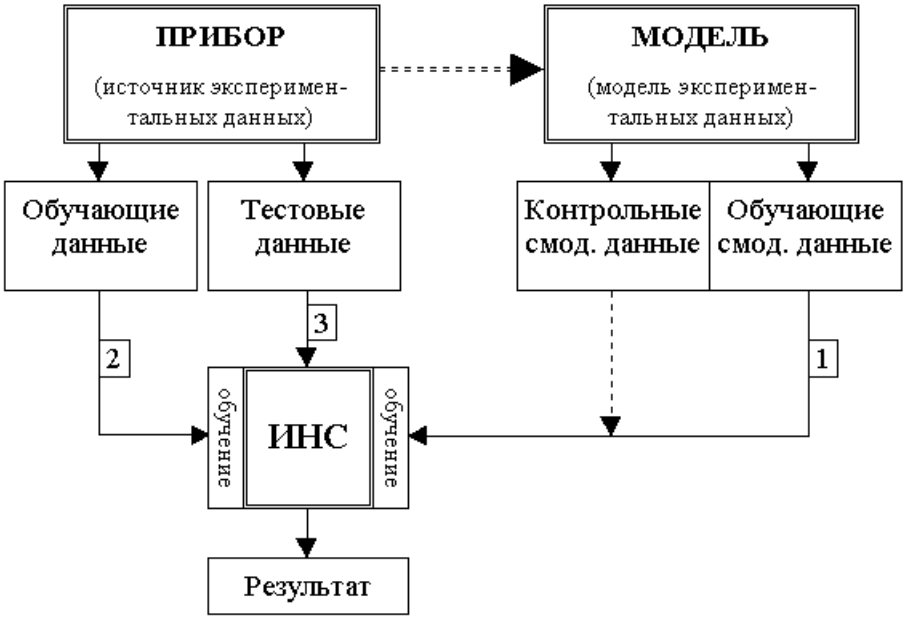
Обучение производится на экспериментальных данных. Из всего множества данных формируется два подмножества: обучающие данные и тестовые данные. ИНС обучается, а затем её работа тестируется (рис. 3.1*а*). Этот метод имеет два основных недостатка. Во-первых, существует опасность переобучения сети. Во-вторых, для эффективной работы сети надо обучить её на достаточно большой и репрезентативной выборке, что на практике связано с получением значительных объёмов экспериментальных данных, а следовательно, с большими материальными и временными затратами.

3.2.2. Обучение с контрольной группой

Об этом методе уже упоминалось в п. 2.3.1. Он состоит в выделении третьего подмножества – контрольного, периодически тестирующего работу сети в процессе обучения [1]. Метод значительно снижает опасность переобучения, однако, как и предыдущий требует большой и репрезентативной обучающей выборки экспериментальных данных. Блок-схема метода представлена на рис. 3.1 *б*.



а) *б*)



в)

Рис. 3.1. Схемы стратегий обучения ИНС (цифрами обозначена последовательность шагов): стандартное обучение (*а*), обучение с контрольной группой (*б*), обучение с помощью модельных данных (*в*).

3.2.3. Обучение с помощью смоделированных данных

Обе предыдущие стратегии имеют один весьма важный недостаток – для их реализации необходим значительный объём экспериментальных данных. Так, для нормальной работы сети нужна обучающая выборка из нескольких сотен спектров, причём, концентрации анализируемого элемента должны покрывать весь динамический диапазон.

Для того, чтобы избавиться от необходимости снятия большого количества экспериментальных данных была предложена следующая стратегия обучения (рис.3.1*в*). Вначале строилась модель экспериментальных спектров. Для этого они аппроксимировались функцией (1.3). Вычитая из спектра его аппроксимацию и применяя цифровую фильтрацию можно получить оценку интерференционной и шумовой составляющих. Подобное разделение проиллюстрировано на рис. 3.2. Зная параметры аппроксимирующей функции, характеристики интерференционной и шумовой составляющих. При этом не составляет труда смоделировать неограниченное число спектров. В процессе моделировании вводились некоторые допуски для параметров, чтобы сделать выборку наиболее репрезентативной.

При обучении генерировались две группы спектров: обучающая и контрольная. Нейронная сеть обучалась на смоделированных спектрах (использовалось порядка 200 спектров с различными параметрами). Изменяющиеся интерференционные составляющие накладывались на спектры уже в процессе обучения. Критерием останова обучения являлся результат работы сети с контрольной группой.

На втором этапе сеть дообучалась на экспериментальных данных. Функция ошибки вела себя так, как показано на рис. 3.3.

Здесь в качестве критерия останова выбиралось поведение функции ошибки. Сеть обучалась до тех пор, пока ошибка монотонно убывала. Появление немонотонности являлось результатом попадания процесса обучения в локальный минимум и считалось началом переобучения.

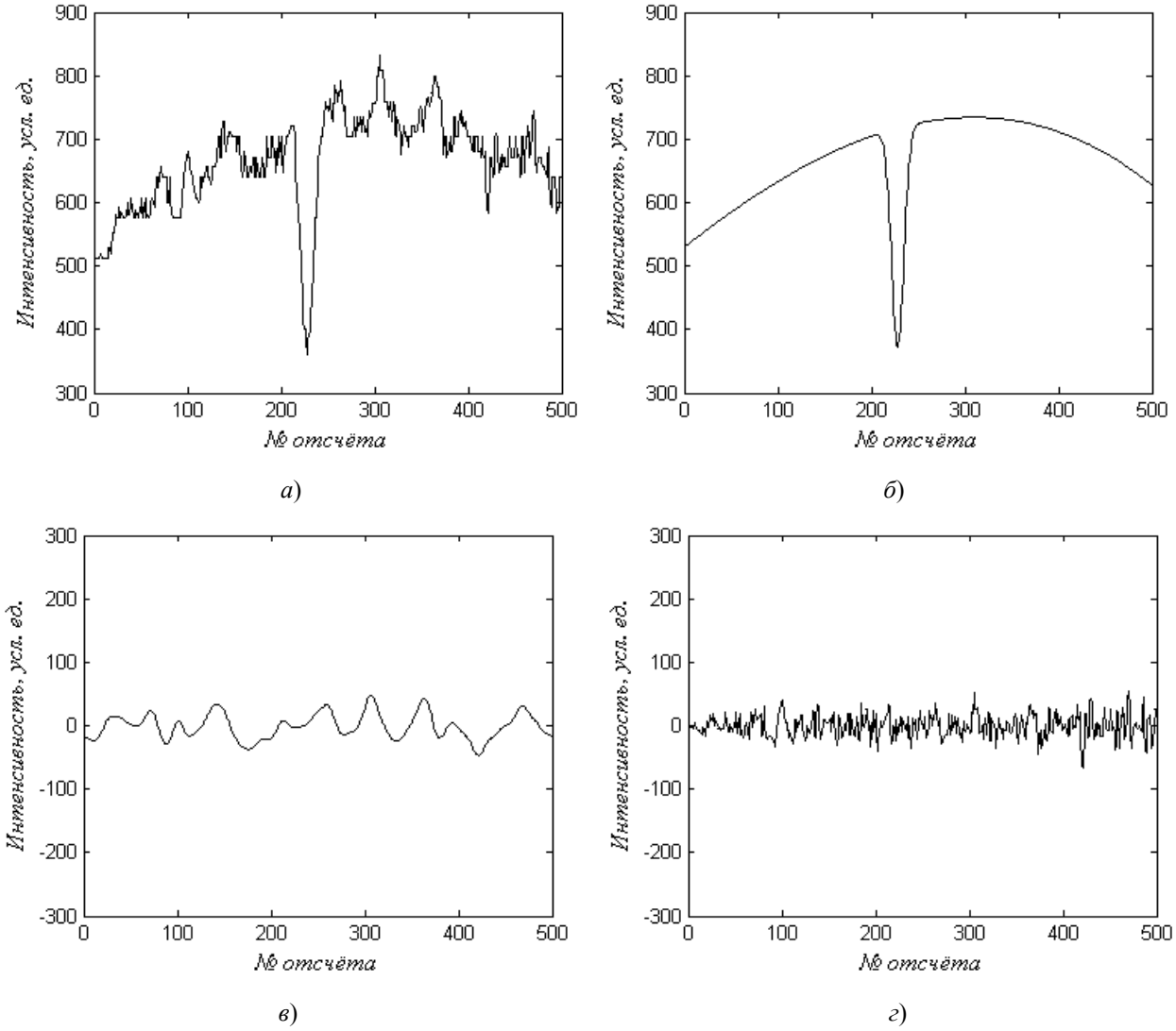


Рис. 3.2. Разложения экспериментального спектра на составляющие: исходный спектр (*а*), результат аппроксимации (*б*), интерференционные составляющие (*в*), высокочастотные шумы (*г*).

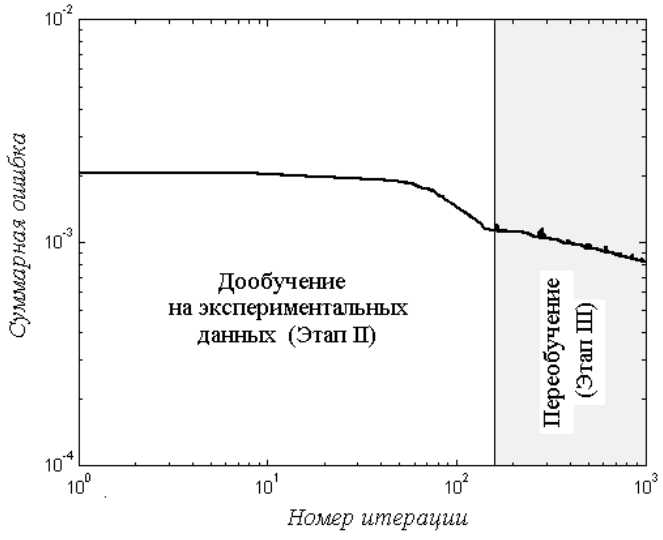


Рис. 3.3. Поведение ошибки при дообучении сети.

3.3. Выбор оптимальной структуры

Как уже упоминалось в п. 2.3.1 весьма важно найти структуру сети близкую к оптимальной для поставленной задачи. Был проведён эмпирический поиск оптимальной структуры. В качестве переменных при этом выступали количества нейронов в 1-ом и 2-ом слоях, т.е. априори постулировалось использование трёхслойного персептрона. Поиск оптимального числа нейронов путём построения двумерной функции ошибок сети, например для значений числа нейронов от 1 до 50 в обоих слоях, требовал значительных вычислительных затрат (до двух суток машинного времени на PC с тактовой частотой процессора 566MHz). Поэтому поиск минимума ошибки осуществлялся методом локальных вариаций. Значение ошибки ИНС при каждом значении числа нейронов в слоях вычислялось как среднее после 20 независимых обучений сети.

Таким образом, было получено, что оптимальной структурой для задачи анализа ВРЛ спектров является трёхслойный персептрон с 17 и 4 нейронами в первом и втором слоях, соответственно.

3.4. Результаты работы метода

Работа метода прямого определения концентраций тестировалась на абсорбционных спектрах цезия (линия 455.531 нм).

Дообучающие экспериментальные спектры снимались при концентрациях Cs 5, 10, 25 и 50 мкг/л, по 16, 23, 23 и 12 спектров, соответственно. Тестовое множество – 61 спектр снималось при концентрации Cs 25 мкг/л.

При определении тестовой концентрации погрешность составила 6.2 %, что в 1.5 раза ниже, чем при использовании метода из п.1.4.3. На рис. 3.5 приведена гистограмма распределения распознанных концентраций, а в табл. 3.1 – статистические параметры распределения.

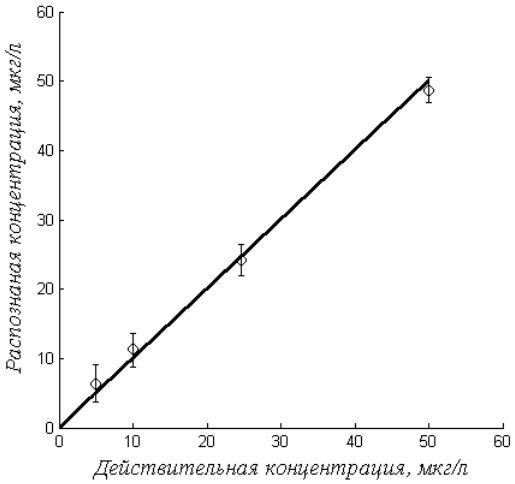


Рис. 3.4. Результат работы сети с обучающим экспериментальным множеством.

Таблица 3.1. Статистические параметры распределения распознанных концентраций.

Параметр	**Значение**
Математическое ожидание	24.8934
Среднеквадратическое отклонение	1.5426
Коэффициент вариации	0.0620
Эксцесс	3.6601
Коэффициент асимметрии	-0.2288

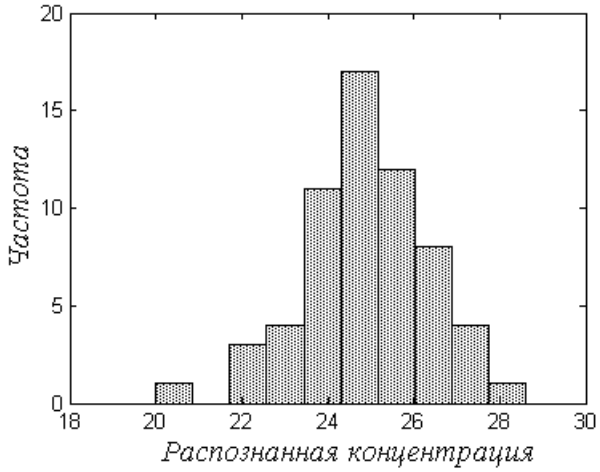


Рис.3.5. Гистограмма распределения распознанных концентраций.

На основании параметров распределения концентраций можно утверждать, что оно имеет симметричную, близкую к нормальной форму.

ЗАКЛЮЧЕНИЕ

Разработанный метод обработки абсорбционных спектров позволил снизить погрешности определения концентраций в среднем в 1.5 раза по отношению к результату работы вариационного метода анализа.

Следует отметить, что снижение погрешности достигается сравнительно дешево, без переделки материальной части прибора. Значительно дороже стоит модернизация ВРЛ-спектрометра, направленная на стабилизацию генерации лазера, на уменьшение шумов электронных устройств. Модернизация ВРЛ-спектрометра совместно с применением современных методов обработки данных позволит ощутимо снизить пределы обнаружения элементов.

Проведённое в работе исследование применимости нейросетевых методов обработки спектров показало их преимущества перед классическими методами при решении обратных задач. Основным недостатком применения ИНС на практике являются:

1) необходимость обучения сети на значительной репрезентативной выборке экспериментальных данных, получение которой ведёт к росту материальных и временных затрат
2) проблемы, возникающие при самом обучении (опасность переобучения).

Эти недостатки, были устранены с помощью разработанной стратегии обучения, использующей модель экспериментальных данных.

Итогом проведённых исследований явилась программа обработки абсорбционных спектров, реализованная в среде разработки DELPHI, а так же библиотека процедур среды MATLAB. Эти программные средства используются при коррекции спектров ВРЛ-спектрометра Минск-2 в лаборатории лазерной диагностики плазмы ИМАФ НАН РБ.

Результаты работы докладывались на 27-ой международной конференции по когерентной и нелинейной оптике (ICONO 2001) [7]. В печати находится 1 журнальная статья (спецвыпуск SPIE).

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

[1]. Boychuk, Churina, Dolenko, Dolenko, Fadeev, Persiantsev. New Opportunities in Solution of Inverse Problems in Laser Spectroscopy Due to Application of Artificial Neural Networks – XVII International Conference on Coherent and Nonlinear Optics (ICONO 2001), Minsk, June 26- July 1, 2001, – p.91.

[2] Bishop M. Neural Networks for Pattern Recognition. – Oxford: Clarendon Press, 1997.

[3]. Уоссерман Ф. Нейрокомпьютерная техника: теория и практика. – М.: Мир, 1992.

[4]. Исаевич А.В. Кандидатская диссертация, – НАН РБ, Мн: 2001. 2842763

[5]. Исаевич А.В., Мелещенко Л.А., Мисаков П.Я. Методы обработки результатов измерений, полученных с помощью внутрирезонаторного лазерного спектрометра. – Препринт № 699, ИМАФ НАН РБ, Мн: 1994, – 23 с.

[6]. Фриш С. Э. и др. Спектроскопия газоразрядной плазмы. Под ред. Фриша С. Э. – Л.: Наука, 1970, – с. 7-16.

[7]. Apanasovich, Buracov, Isaevich, Misakov, Lutkovski, Nazarov. Data Processing and Estimation of Measurement Errors in Intracavity Laser Spectroscopy – XVII International Conference on Coherent and Nonlinear Optics (ICONO 2001), Minsk, June 26- July 1, 2001, – p.81.

ПРИЛОЖЕНИЕ 1. ПРОГРАММА ОБРАБОТКИ ВРЛ СПЕКТРОВ

Ниже приведено описание программного продукта, позволяющего производить нейросетевой анализ ВРЛ спектров (Рис.1).

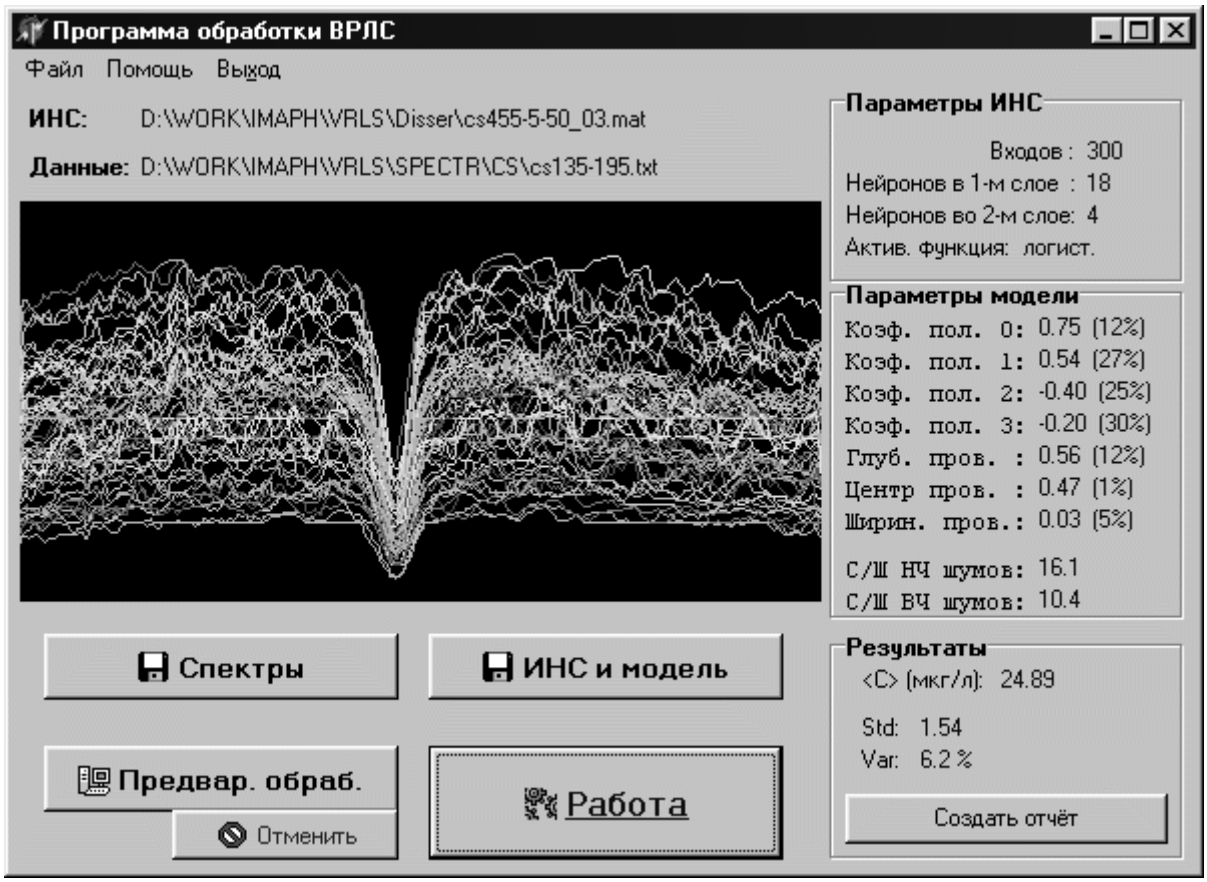


Рис. 1. Внешний вид разработанного приложения.

Программа позволяет загружать экспериментальные спектры, параметры обученной ИНС и модели, производить предварительную (фильтрация, устранение δ-выбросов) и нейросетевую обработку спектров. При загрузке поддерживаются следующие форматы данных: стандартный формат данных MATLAB Internal Format V.5 и формат ASCII.

Основой нейросетевой части приложения является объект класса cMultyLayerPerceptron, представленный в Приложении 2.

По результатам работы приложения генерируется отчёт (в формате ASCII). в него включаются: имена файла данных, значение распознанных концентраций для каждого спектра, статистические параметры концентраций.

Обучение нейронной сети и экстракция параметров модели производятся в среде MATLAB. Листинги соответствующих процедур приведены в Приложении 3.

Ниже представлен пример файла отчёта.

```
% Report file
%
% Data file name: D:\WORK\IMAPH\VRLS\SPECTR\cs135-195.txt
% ANN  file name: D:\WORK\IMAPH\VRLS\Disser\cs455-5-50_03.mat
%
%-------------------------------------------------------------
% Number of Spectrums
61

%-------------------------------------------------------------
% Conc   #Spc
22.0593   %1
25.1766   %2
24.3511   %3
22.6436   %4
26.1435   %5
24.9847   %6
26.1317   %7
23.5571   %8
24.9967   %9

  .  .  .

%-------------------------------------------------------------
% Stat Param

% M=
24.8934

% STD=
1.5426

% VAR=
0.0620
```

ПРИЛОЖЕНИЕ 2. КЛАСС cMultyLayerPerceptron

```
{*************************************
 *  Multy Layer Perceptron class        *
 *                        by P.Nazarov  *
 *************************************}

unit MLP;
interface
Type
    aReal=array[1..1] of real;

    paReal=^aReal;

    aInt=array[1..1] of integer;

    paInt=^aInt;

    cNeuron=class
             N_W:integer;
             W:^aReal;
             B:real;
             data:real;
             constructor Create(_N_W:byte);
             destructor Erase;
            end;

    aNeuron=array[1..1] of cNeuron;

    cLayer=class
            N_Neuron:byte;
            Neuron:^aNeuron;
            constructor Create(_N_Neuron:byte);
            destructor Erase;
           end;

    aLayer=array[1..1] of cLayer;

    cMultyLayerPerceptron=class
           N_X,N_Y:byte;
           N_Layer:byte;
           Layer:^aLayer;
           X,Y:^aReal;
           constructor Create(_N_X,_N_Y,_N_Layer:integer;_PN:paInt);
           destructor Erase;
           procedure Calculate;
           function Test(_X,_Y:aReal):real;
           procedure SetWeight(_L,_N,_W:integer;data:real);
           procedure SetBias(_L,_N:integer;data:real);
    end;

    function Logsig(x:real):real;

implementation

//--------------- cNeuron -------------
 Constructor cNeuron.Create(_N_W:byte);
  Begin
   N_W:=_N_W;
   if N_W<>0 then GetMem(W,sizeof(real)*N_W)
             else W:=Nil;
   B:=0;
   data:=0;
  End;

 Destructor cNeuron.Erase;
```

```
  Begin
   if N_W>0 then FreeMem(W,sizeof(real)*N_W);
  End;

//--------------- cNeuron -------------
 Constructor cLayer.Create(_N_Neuron:byte);
  Begin
   N_Neuron:=_N_Neuron;
   if N_Neuron<>0 then
    GetMem(Neuron,SizeOf(aNeuron)*N_Neuron)
   else Neuron:=0;
  End;

 Destructor cLayer.Erase;
  Begin
   if N_Neuron>0 then
  FreeMem(Neuron,sizeof(aNeuron)*N_Neuron);
  End;

//------- cMultyLayerPerceptron -------
Constructor cMultyLayerPerceptron.Create(_N_X,_N_Y,_N_Layer:integer;_PN:paInt);
  var i,j,l:byte;
      PN:paInt;
  Begin
   N_Layer:=_N_Layer;
   GetMem(Layer,SizeOf(aLayer)*N_Layer);
   N_X:=_N_X;
   N_Y:=_N_Y;
   PN:=_PN;
   GetMem(X,SizeOf(aReal)*N_X);
   GetMem(Y,SizeOf(aReal)*N_Y);
   for i:=1 to N_X do X^[i]:=0;
   for i:=1 to N_Y do Y^[i]:=0;
   for i:=1 to N_Layer do begin
    Layer^[i].Create(PN^[i]);
    for j:=1 to Layer^[i].N_Neuron do
     if i=1 then Layer^[i].Neuron^[j].Create(N_X)
            else Layer^[i].Neuron^[j].Create(Layer^[i-1].N_Neuron);
   end
  End;

 Destructor cMultyLayerPerceptron.Erase;
  var i,j:integer;
 Begin
   FreeMem(X,SizeOf(aReal)*N_X);
   FreeMem(Y,SizeOf(aReal)*N_Y);
   FreeMem(Layer,SizeOf(aLayer)*N_Layer);
   for i:=1 to N_Layer do
    for j:=1 to Layer^[i].N_Neuron do
      Layer^[i].Neuron^[j].Erase;
   Layer^[i].Erase;
 End;

 Procedure cMultyLayerPerceptron.Calculate;
  var i,j,l,n,v:byte;
  Begin
   for l:=1 to N_Layer do
    for n:=1 to Layer^[l].N_Neuron do
     Layer^[l].Neuron^[n].data:=0;
   for l:=1 to N_Layer do
    Begin
     if l=1 then
      for i:=1 to Layer^[l].N_Neuron do begin
       for j:=1 to N_X do
        Layer^[l].Neuron^[i].data:=Layer^[l].Neuron^[i].data +
Layer^[l].Neuron^[i].W^[j]*X^[j];
       Layer^[l].Neuron^[i].data:=Logsig(Layer^[l].Neuron^[i].data -
Layer^[l].Neuron^[i].B);
```

```
      end
     else
      for i:=1 to Layer^[l].N_Neuron do
       for j:=1 to Layer^[l-1].N_Neuron do
        Layer^[l].Neuron^[i].data:=Layer^[l].Neuron^[i].data+
Layer^[l-1].Neuron^[j].data*Layer^[l].Neuron^[i].W[j];
      Layer^[l].Neuron^[i].data:=Logsig(Layer^[l].Neuron^[i].data-
Layer^[l].Neuron^[i].B);
    End;
    for i:=1 to N_Y do Y^[i]:=Layer^[N_Layer].Neuron^[i].data;
  End;

Function cMultyLayerPerceptron.Test(_X,_Y:aReal):real;
  var e:real;
      i:byte;
  Begin
   X^:=_X;     {загоняем в NNet.X вход ОП}
   Calculate;
   e:=0;
   for i:=1 to N_Y do
    e:=e+abs(Y^[i]-_Y[i]);
   Test:=e;
  End;

Procedure cMultyLayerPerceptron.SetWeight(_L,_N,_W:integer;data:real);
 Begin
  Layer^[_l].Neuron^[_n].W^[_w]:=data;
 End;

Procedure cMultyLayerPerceptron.SetBias(_L,_N:integer;data:real);
 Begin
  Layer^[_l].Neuron^[_n].B:=data;
 End;

//---------------------------------------------------------------------

Function Logsig(x:real):real;
  Begin  Result:=1/(1+exp(-x)); End;
end.
```

ПРИЛОЖЕНИЕ 3. ЛИСТИНГИ ФУНКЦИЙ (MATLAB)

Функция VRLS_MASP, реализующая аппроксимацию спектра.

```
%VRLS Method of Aproximation Spectra Processing
function [dI,dS,I0,S0,X_new,Ground,K]=VRLS_MASP(X,a);
if nargin==1
    a=0.1;
end;
nSmp=length(X);
nSmp2=round(nSmp/2);

%-----------------------------------------
% Gradient analysis
Y(1)=0;
for i=2:nSmp
      Y(i)=abs(X(i)-X(i-1));
end;
X1=X;
for i=2:nSmp
    if Y(i)>200
         X1(i)=X1(i-1);
    end;
end;
X2=X;
for i=1:nSmp-1
      if Y(nSmp-i)>200
         X2(nSmp-i)=X2(nSmp-i+1);
      end;
end;
X=(X1+X2)./2;
clear Y X1 X2;

%-----------------------------------------
%Filtering
%W=exp(-((1:nSmp2)/50).^2);
W=exp(-((1:nSmp2)/100).^2);
X=fftfilter(X,W);

%-----------------------------------------
%Approximation 1
Border1= [-10,10;
         -10,10;
         -10,10;
         -10,10];
P1=[0.6,0.8,-0.4,-0.3];
Weights1=ones(1,nSmp);
Weights1(200:300)=0;Weights1(1:10)=0;Weights1(490:500)=0;
MP=[100,1e-10,1e-10,000,0.001,1.5,1.1];
freq=(1:nSmp)./nSmp;
[P1,e,a]=Min_LocalVariations2(freq,(X./max(X)),'F_Polynom3',P1,'E_square',Border1,W
eights1,MP);
Ground=(feval('F_Polynom3',freq,P1).*max(X));
X1=X-Ground;

%-----------------------------------------
%Approximation 2
Border2= [P1(1)-0.1*abs(P1(1)),P1(1)+0.1*abs(P1(1));
          P1(2)-0.1*abs(P1(2)),P1(2)+0.1*abs(P1(2));
          P1(3)-0.1*abs(P1(3)),P1(3)+0.1*abs(P1(3));
          P1(4)-0.1*abs(P1(4)),P1(4)+0.1*abs(P1(4));
          0,1;
          0.4,0.6;
```

```
          1e-5,1];
P2=[P1,max(-X1)/max(X),0.5,0.1];
Weights2=ones(1,nSmp);Weights2(1:10)=0;Weights2(490:500)=0;
MP=[100,1e-10,1e-10,000,0.001,1.5,1.1];
[P2,e,a]=Min_LocalVariations2(freq,(X./max(X)),'F_VRLS',P2,'E_square',Border2,Weigh
ts2,MP);
X_new=(feval('F_VRLS',freq,P2).*max(X));
K=P2;
P1_new=P2(1:4);
Ground=(feval('F_Polynom3',freq,P1_new).*max(X));

%-----------------------------------------
dI=P2(5).*max(X);
I0=Ground(round(P2(6)*nSmp));
dS=sum(Ground-X_new);
g1=round((P2(6)-P2(7)*sqrt(-log(a)))*nSmp);
g2=round((P2(6)+P2(7)*sqrt(-log(a)))*nSmp);
S0=GetArea(Ground,0,[g1 g2],1);
```

Функция, реализующая экстракцию параметров модели из набора экспериментальных данных GetModelParamFunction.

```
%Get Model's Parameters from real spectra
function    [M_K    M_m_LF    M_m_HF    M_s_LF    M_s_HF    M_SN_HF    M_SN_LF]    =
GetModelParamFunction(X0);
[nSmp,nSpc]=size(X0);

M_K=zeros(nSpc,7);
M_NoiseHF=zeros(nSpc,nSmp);
M_NoiseLF=zeros(nSpc,nSmp);
M_SN_HF=zeros(1,nSpc);
M_SN_LF=zeros(1,nSpc);

for iSp=1:nSpc
    disp(sprintf('Spectrum #%d',iSp,SpcNum(iSp),nSpc));
    [dI(iSp),dS(iSp),I0(iSp),S0(iSp),x,g,M_K(iSp,:)]=VRLS_MASP(X(:,iSp)',0.01);
    X(:,iSp)=x';
    Ground(:,iSp)=g';
end;

MNoiseLF=X-X0;
W=gausspdf(1:250,0,7);W=W./max(W);
for iSp=1:nSpc
    MNoiseLF(:,iSp)=X(:,iSp)-X0(:,iSp);
    MNoiseLF(:,iSp)=(fftfilter(MNoiseLF(:,iSp)',W))';
    MNoiseHF(:,iSp)=(X(:,iSp)-X0(:,iSp))-MNoiseLF(:,iSp);
    M_SN_LF(iSp)=dI(iSp)/std(MNoiseLF(:,iSp));
    M_SN_HF(iSp)=dI(iSp)/std(MNoiseHF(:,iSp));
end;

M_m_LF=mean(MNoiseLF);
M_m_HF=mean(MNoiseHF);
M_s_LF=std(MNoiseLF);
M_s_HF=std(MNoiseHF);

save Model01 -ascii M_K M_m_LF M_m_HF M_s_LF M_s_HF M_SN_HF M_SN_LF;
```

Процедура my_traingdx, обучающая сеть по разработанной стратегии.

```
function [net,tr,Ac,El] = my_traingdx(net,Pd,Tl,Ai,Q,TS,VV,TV)

if isstr(net)
  switch (net)
    case 'pnames',
      net = fieldnames(traingdx('pdefaults'));
    case 'pdefaults',
         trainParam.epochs = 100;
         trainParam.goal = 0;
         trainParam.lr = 0.01;
         trainParam.lr_dec = 0.7;
         trainParam.lr_inc = 1.05;
         trainParam.max_fail = 5;
         trainParam.max_perf_inc = 1.04;
         trainParam.mc = 0.9;
         trainParam.min_grad = 1.0e-6;
         trainParam.show = 25;
         trainParam.time = inf;
         net = trainParam;
    otherwise
      error('Unrecognized code.')
  end
  return
end

% CALCULATION
% ===========
% Parameters
epochs = net.trainParam.epochs;
goal = net.trainParam.goal;
lr = net.trainParam.lr;
lr_inc = net.trainParam.lr_inc;
lr_dec = net.trainParam.lr_dec;
max_fail = net.trainParam.max_fail;
max_perf_inc = net.trainParam.max_perf_inc;
mc = net.trainParam.mc;
min_grad = net.trainParam.min_grad;
show = net.trainParam.show;
time = net.trainParam.time;

% Parameter Checking
if (~isa(epochs,'double')) | (~isreal(epochs)) | (any(size(epochs)) ~= 1) | ...
  (epochs < 1) | (round(epochs) ~= epochs)
  error('Epochs is not a positive integer.')
end
if (~isa(goal,'double')) | (~isreal(goal)) | (any(size(goal)) ~= 1) | ...
  (goal < 0)
  error('Goal is not zero or a positive real value.')
end
if (~isa(lr,'double')) | (~isreal(lr)) | (any(size(lr)) ~= 1) | ...
  (lr < 0)
  error('Learning rate is not zero or a positive real value.')
end
if (~isa(lr_inc,'double')) | (~isreal(lr_inc)) | (any(size(lr_inc)) ~= 1) | ...
  (lr_inc < 1)
  error('LR_inc is not a positive real value greater or equal to 1.0.')
end
if (~isa(lr_dec,'double')) | (~isreal(lr_dec)) | (any(size(lr_dec)) ~= 1) | ...
  (lr_dec < 0) | (lr_dec > 1)
  error('LR_dec is not a positive real value greater or between 0.0 and 1.0.')
end
if (~isa(max_fail,'double')) | (~isreal(max_fail)) | (any(size(max_fail)) ~= 1) |
...
  (max_fail < 1) | (round(max_fail) ~= max_fail)
```

```
  error('Max_fail is not a positive integer.')
end
if       (~isa(max_perf_inc,'double'))       |       (~isreal(max_perf_inc))       |
(any(size(max_perf_inc)) ~= 1) | ...
  (max_perf_inc < 1)
  error('Max_perf_inc is not a positive real value greater or equal to 1.0.')
end
if (~isa(mc,'double')) | (~isreal(mc)) | (any(size(mc)) ~= 1) | ...
  (mc < 0) | (mc > 1)
  error('MC is not real value between 0.0 and 1.0.')
end
if  (~isa(min_grad,'double')) | (~isreal(min_grad)) | (any(size(min_grad)) ~= 1) |
...
  (min_grad < 0)
  error('Min_grad is not zero or a positive real value.')
end
if (~isa(show,'double')) | (~isreal(show)) | (any(size(show)) ~= 1) | ...
  (isfinite(show) & ((show < 1) | (round(show) ~= show)))
  error('Show is not ''NaN'' or a positive integer.')
end
if (~isa(time,'double')) | (~isreal(time)) | (any(size(time)) ~= 1) | ...
  (time < 0)
  error('Time is not zero or a positive real value.')
end

% Constants
this = 'TRAINGDX';
doValidation = ~isempty(VV);
doTest = ~isempty(TV);

% Initialize
flag_stop=0;
stop = '';
startTime = clock;
X = getx(net);
[perf,El,Ac,N,Zb,Zi,Zl] = calcperf(net,X,Pd,Tl,Ai,Q,TS);
perf_1=perf;
[gX,normgX] = calcgx(net,X,Pd,Zb,Zi,Zl,N,Ac,El,perf,Q,TS);
dX = lr*gX;
if (doValidation)
  VV.net = net;
  vperf = calcperf(net,X,VV.Pd,VV.Tl,VV.Ai,VV.Q,VV.TS);
  VV.perf = vperf;
  VV.numFail = 0;
end
tr = newtr(epochs,'perf','vperf','tperf','lr');

%-------------------------------------------
Window=gausspdf(250,0,7);
Window=Window./max(Window);
%-------------------------------------------

% Train
for epoch=0:epochs
  % Training Record
  epochPlus1 = epoch+1;
  tr.perf(epochPlus1) = perf;
  tr.lr(epochPlus1) = lr;
  if (doValidation)
    tr.vperf(epochPlus1) = vperf;
  end
  if (doTest)
    tr.tperf(epochPlus1) = calcperf(net,X,TV.Pd,TV.Tl,TV.Ai,TV.Q,TV.TS);
  end

%-----------------------------------------------------------
% Check by the Control Set
load ControlSet KS;
```

```
perf_0=perf_1;
perf_1=calcperf(net,X,Pd,KS,Ai,Q,TS);
clear KS;
%-------------------------------------------------------

% Stopping Criteria
  currentTime = etime(clock,startTime);
  if (perf <= goal)
    stop = 'Performance goal met.';
  elseif (epoch == epochs)
    stop = 'Maximum epoch reached, performance goal was not met.';
  elseif (currentTime > time)
    stop = 'Maximum time elapsed, performance goal was not met.';
  elseif (normgX < min_grad)
    stop = 'Minimum gradient reached, performance goal was not met.';
  elseif (doValidation) & (VV.numFail > max_fail)
    stop = 'Validation stop.';
  elseif flag_stop
    stop = 'User stop.';
  elseif (epoch>100)&(perf_1>perf_0)
    stop = 'Stop by Control Set';
  end

  % Progress
  if isfinite(show) & (~rem(epoch,show) | length(stop))
    fprintf(this);
       if isfinite(epochs) fprintf(', Epoch %g/%g',epoch, epochs); end
       if isfinite(time) fprintf(', Time %g%%',currentTime/time/100); end
       if isfinite(goal) fprintf(', %s %g/%g',upper(net.performFcn),perf,goal); end
       if isfinite(min_grad) fprintf(', Gradient %g/%g',normgX,min_grad); end
       fprintf('\n')
       flag_stop=plotperf(tr,goal,this,epoch);
    if length(stop) fprintf('%s, %s\n\n',this,stop); end
  end

  % Stop when criteria indicate its time
  if length(stop)
    if (doValidation)
         net = VV.net;
       end
    break
  end
%-------------------------------------------------------------
%                           Change Trainers
[NI,NT]=size(Tl);

load ModelParam01 m_K s_K s_N;

for i=1:NT
    Tl(i,:)=F_VRLS((1:500)./500,m_K+s_K.*randr(1,7));
    Noise=randn(500,1);
    Noise=fftfilter(Noise,Window);
    Noise=Noise./std(Noise);
    Noise=Noise.*s_N;
    Tl(i,:)=Tl(i,:)+Noise;
end;

clear m_K;
clear s_K;
%-------------------------------------------------------------

  % Gradient Descent with Momentum and Adaptive Learning Rate
  dX = mc*dX + (1-mc)*lr*gX;
  X2 = X + dX;
  net2 = setx(net,X2);
  [perf2,El,Ac,N,Zb,Zi,Zl] = calcperf(net2,X2,Pd,Tl,Ai,Q,TS);
  if (perf2/perf) > max_perf_inc
    lr = lr*lr_dec;
```

```
      dX = lr*gX;
  else
    if (perf2 < perf)
      lr = lr*lr_inc;
      end
      X = X2;
      net = net2;
      perf = perf2;
      [gX,normgX] = calcgx(net,X,Pd,Zb,Zi,Zl,N,Ac,El,perf,Q,TS);
    norm_g = sqrt(sum(sum(gX.^2)));
  end

  % Validation
  if (doValidation)
    vperf = calcperf(net,X,VV.Pd,VV.Tl,VV.Ai,VV.Q,VV.TS);
      if (vperf < VV.perf)
        VV.perf = vperf; VV.net = net; VV.numFail = 0;
      elseif (vperf > VV.perf)
      VV.numFail = VV.numFail + 1;
      end
  end
end

% Finish
tr = cliptr(tr,epoch);
```