Министерство образования и науки РФ
Федеральное государственное автономное образовательное учреждение
высшего образования
«Национальный исследовательский технологический университет «МИСиС»
Новотроицкий филиал

Кафедра математики и естествознания

А.В. Саблин

МОДЕЛИРОВАНИЕ ХИМИКО-ТЕХНОЛОГИЧЕСКИХ ПРОЦЕССОВ

Учебное пособие

Новотроицк 2016

УДК 66,011:519,876(075,8)

ББК 35,11:22,1я73

С 122

Рецензент:

Доцент кафедры машиностроения, материаловедения и автомобильного транспорта Орского гуманитарно-технологического института (филиала) ОГУ, кандидат технических наук

Н.В. Фирсова

Зав. кафедры электроэнергетики и теплоэнергетики Орского гуманитарно-технологического института (филиала) ОГУ, кандидат педагогических наук,

Р.Е. Мажирина

Саблин А.В. Моделирование химико-технологических процессов. – Новотроицк. НФ НИТУ «МИСиС». – 2016. –181 с.

ISBN 978-5-903472-27-7

Учебное пособие предназначено для использования студентами, обучающимися по направлению подготовки «Химическая технология», а также научно-педагогическими работниками, областью интересов которых является математическое описание и моделирование процессов химических производств. В пособии представлены общие сведения о моделировании элементов химического оборудования и процессов, даны примеры построения статических и динамических моделей с использованием прикладных пакетов Matlab и Ms Excel.

Рекомендовано Методическим Советов НФ НИТУ «МИСиС»

ISBN 978-5-903472-27-7

Содержание

Введение

Решение химико-технологических задач методом математического моделирования требует комплексного подхода к разработке и реализации алгоритма на базе активного знания фундаментальных и инженерных дисциплин – математики, численных методов решения, химии, физической химии, процессов и аппаратов химической технологии, основ экономики, общей химической технологии. Настоящее пособие явилось результатом обобщения материалов лекционных и практических занятий по одноименному курсу дисциплин.

Основными задачами курса являются:

- разработка интеллектуального ядра метода математического моделирования;

- формирование группы основных алгоритмов компьютерного решения задач химической технологии;

- анализ, исследование и оптимизация типовых технологических процессов.

По каждому из основных разделов пособия, методы моделирования и расчета иллюстрируются развернутыми примерами, которые позволяют изучить практическую методологию решения большого числа задач химической технологии. Материал пособия составлен на базе основных типовых учебников по моделированию, оптимизации и расчету основных технологических процессов, аппаратов и систем, а также ряда оригинальных материалов.

Пособие предназначено для студентов и научно-педагогических работников, сфера интересов которых связана с металлургическими и химическими технологиями.

В результате проработки контрольных заданий обучаемый получит развитие следующих навыков:

- планировать и проводить физические и химические эксперименты, проводить обработку их результатов и оценивать погрешности, математически моделировать физические и химические процессы и явления, выдвигать гипотезы и устанавливать границы их применения;

- способность использовать знание свойств химических элементов, соединений и материалов на их основе для решения задач профессиональной деятельности;

- проводить анализ состава и качества продукции переработки природных энергоносителей и углеродных материалов;

- разрабатывать и совершенствовать технологические процессы переработки природных энергоносителей и углеродных материалов;

- составлять математические модели типовых профессиональных задач, находить способы их решений и интерпретировать профессиональный (физический) смысл полученного математического результата;

- применять аналитические и численные методы решения поставленных задач, использовать современные информационные технологии, проводить обработку информации с использованием прикладных программ деловой сферы деятельности.

По содержанию данной работы, отметим, что в пособии рассмотрены основные вопросы моделирования систем по результатам пассивного и активного экспериментов, показаны способы математического описания процессов на основе теоретических закономерностей. Следует заметить, что использование данного пособия не избавит читателя от поиска и прочтения литературных источников по рассматриваемым в пособии темам. Основная цель пособия – задать направление движения исследователя, позволяющее решить поставленную перед ним техническую задачу с наименьшими трудозатратами. Так, как решение технических задач моделирования в определенной степени требует творческого подхода, автор не навязывает студенту предложенные в пособии способы и методы, как единственно - верные. Скорее – наоборот, альтернативные решения, которые, как правило, являются частными случаями, нередко приводят к неплохим результатам.

1 Введение в математическое моделирование

1.1 Классификация и назначение моделей процессов и оборудования

Модели позволяют представить в наглядной форме объекты и процессы, недоступные для непосредственного восприятия. Так, например, в физике и термодинамике широко используются модели двигателей; глобус – модель земли, используемая при изучении географии; модели кристаллической решетки, модели молекул и атомов применяют в химии. При проектировании механизмов и устройств, зданий, электрических цепей используют модели – чертежи и макеты.

В развитии современной науки особую роль играют теоретические модели – теории, законы, гипотезы и т.д. Нередко, создание таких моделей качественно меняет представления человека об окружающей действительности.

Следует отметить, что один и тот же объект может иметь множество моделей.

Например, объект «абсорбер» может иметь следующие модели:

1) химия – равновесные и рабочие составы жидкостей и газов;

2) физика – материальная точка;

3) физическая химия – кинетическая кривая абсорбции

4) механика – расчет конструкции на прочность.

Дополним, что разные объекты могут описываться одной моделью. Таким образом, разнообразие объектов изучения современной науки определяет многообразие моделей, имитирующих эти объекты или их определенные свойства. Однако, по ряду признаков все модели, а также способы их построения можно объединить в укрупненные образования – классы. В настоящее время не существует единой, общепринятой классификации моделей. Тем не менее, приведем наиболее устоявшиеся подходы к классификации моделей.

Обобщая предложенные множеством авторов схемы, выделим следующие признаки классификаций моделей:

1) по области использования;

2) по фактору времени;

3) по отрасли знаний;

4) по форме представления

Так, классифицируя модели по области использования можно выделить:

а) учебные модели – они используются при обучении персонала предприятий, студентов или школьников;

б) опытные модели – это уменьшенные или увеличенные копии проектируемого объекта. Их используют для исследования и прогнозирования будущих характеристик изделия или процесса;

в) научно - технические модели – создаются для исследования процессов и явлений окружающей действительности;

г) игровые модели – репетиция поведения объекта в различных условиях;

д) имитационные – отражение реальности в той или иной степени (иначе – это метод проб и ошибок).

Любой из перечисленных классов моделей можно также классифицировать по изменению параметров объекта во времени. В классификации моделей по фактору времени все модели можно разбить на два крупных класса:

а) статические модели – модели, описывающие состояние системы в определенный момент времени (единовременный срез информации по данному объекту). Примеры моделей: строение молекул, список закупок материалов и оборудования, отчет об обследовании состояния оборудования, материальный баланс установки и т.д.;

б) динамические модели – такие модели описывают процессы изменения и развития системы (изменения объекта или параметра объекта во времени). Примеры: описание движения тел, процесс кинетики химических превращений, переходные процессы в реальных системах и т.д.

По отрасли деятельности человека модели можно разделить на математические, биологические, химические, социальные, экономические, исторические и т.д.

По форме представления модели бывают:

а) материальные – это предметные (физические) модели. Они всегда имеют реальное воплощение. Отражают внешнее свойство и внутреннее устройство исходных объектов, суть процессов и явлений объекта-оригинала. Это экспериментальный метод познания окружающей среды. Примеры: детские игрушки, скелет человека, чучело, макет солнечной системы, школьные пособия, физические и химические опыты;

б) абстрактные (нематериальные) – не имеют реального воплощения. Их основу составляет информация. Это теоретический метод познания окружающей среды. По признаку реализации они бывают: мысленные и вербальные; а также информационные.

Процесс построения информационных моделей с помощью формальных языков (математических, логических и т.д.) называется формализацией. Более полное определение формализации – это приведение (сведение) существенных свойств и признаков объекта моделирования к выбранной форме.

Различают следующие формы представления информационных моделей:

а) табличные – объекты и их свойства представлены в виде списка, а их значения размещаются в ячейках прямоугольной формы. Перечень однотипных объектов размещен в первом столбце (или строке), а значения их свойств размещаются в следующих столбцах (или строках);

б) иерархические – объекты распределены по уровням. Каждый элемент высокого уровня состоит из элементов нижнего уровня, а элемент нижнего уровня может входить в состав только одного элемента более высокого уровня;

в) сетевые – применяют для отражения систем, в которых связи между элементами имеют сложную структуру.

По степени формализации информационные модели бывают образно-знаковые и знаковые.

Например, к образно-знаковым моделям относят:

а) геометрические модели (рисунок, пиктограмма, чертеж, карта, план, объемное изображение);

б) структурные модели (таблица, граф, схема, диаграмма);

в) словесные модели (описание свойств объекта изучения естественными языками);

г) алгоритмические модели (нумерованный список, пошаговое перечисление, блок-схема).

К знаковым моделям отнесем:

а) математические модели – они представлены математическими формулами, отображающими связь параметров объекта изучения;

б) специальные – представлены описанием на специальных языках (ноты, химические формулы);

в) алгоритмические. В данном случае – это программы.

При построении математических моделей процессов функционирования систем существуют следующие основные подходы: непрерывно-детерминированный (например, дифференциальные уравнения, уравнения состояния); дискретно-детерминированный (конечные автоматы); дискретно-стохастический (вероятностные автоматы); непрерывно-стохастический (системы массового обслуживания); обобщенный или универсальный (агрегативные системы).

Классификация моделей и видов моделирования объектов и систем в соответствии с теорией подобия должна выделить в них наиболее общие признаки и свойства реальных систем. Ниже приведена одна из возможных классификаций (таблица 1).

Формальная классификация математических моделей основывается на классификации используемых математических средств. Здесь, математические модели можно разделить на:

- линейные или нелинейные модели;
- сосредоточенные или распределённые системы;
- детерминированные или стохастические;
- статические или динамические;
- дискретные или непрерывные.

Каждая построенная модель является линейной или нелинейной, детерминированной или стохастической. Естественно, что возможны и смешанные типы: в одном

отношении сосредоточенные (по части параметров), в другом – распределённые модели и т. д.

Таблица 1 – Классификация моделей и видов моделирования

Признаки классификации	Виды математических моделей
1. Принадлежность к иерархическому уровню	Модели микроуровня Модели макроуровня Модели метауровня
2. Характер взаимоотношений со средой	Открытые непрерывный обмен) Закрытые (слабая связь)
3. Характер отображаемых свойств объекта	Структурные Функциональные
4. Способ представления свойств объекта	Аналитические Алгоритмические Имитационные
5. Способ получения модели	Теоретические Эмпирические
6. Причинная обусловленность	Детерминированные Вероятностные
7. По отношению к времени	Динамические Статические
8. По типу уравнений	Линейные Нелинейные
9. По множеству значений переменных	Непрерывные Дискретные Дискретно-непрерывные
10. По назначению	Технические Экономические Социальные и т.д.

Моделирование в целом включает в себя ряд этапов, базирующихся на системном подходе:

1. Содержательная постановка задачи: выработка общего подхода к исследуемой проблеме; определение подзадач; определение основной цели и путей ее достижения.

2. Изучение и сбор информации об объекте-оригинале: анализ или подбор подходящих гипотез, аналогий, теорий; учет опытных данных, наблюдений и т.д.; определение входных и выходных переменных, связей; принятие упрощающих предположений.

3. Формализация: принимаются условные обозначения и с их помощью описываются связи между элементами объекта в виде математических выражений. Намечается переход к количественному анализу.

4. Выбор метода решения. Для поставленной математической задачи обосновывается метод ее решения с учетом знаний и предпочтений пользователя и разработчика. При проектировании приходится решать, как линейные, так и нелинейные задачи, использовать ручные и машинные методы проектирования, расчета и исследований.

5. Реализация модели. Принимается критерий оценки эффективности модели, разрабатывается алгоритм, пишется и отлаживается программа, чтобы осуществить системный анализ и синтез.

6. Анализ полученных результатов. Сопоставляется предполагаемое и полученное решение, проводится оценка адекватности и погрешности моделирования. Процесс моделирования является итеративным. В случае неудовлетворительных результатов, полученных на этапах 5 или 6, осуществляется возврат к одному из ранних этапов, который мог привести к разработке неудачной модели. Уточнение модели происходит до тех пор, пока не будут получены приемлемые результаты.

Таким образом, после прохождения этих этапов наиболее полно могут быть выполнены требования, предъявляемые к моделям:

Универсальность – характеризует полноту отображения моделью изучаемых свойств реального объекта;

Адекватность – способность отражать нужные свойства объекта с погрешностью не выше допустимой;

Точность – оценивается степенью совпадения значений характеристик реального объекта со значениями этих характеристик, полученных с помощью моделей;

Экономичность – определяется затратами ресурсов ЭВМ (памяти и времени на ее реализацию и эксплуатацию).

Качество моделирования может быть оценено характеристикой его потребительских свойств:

- эффективность использования его по назначению (цели);

- ресурсоемкость;

- стоимость.

Следует отметить, что математический подход к моделированию имеет ряд недостатков, а именно:

- часто, низкая адекватность математической модели реальному объекту;

- проблемы, связанные с решаемостью математических моделей из-за наличия в них разрывных функций;

- непригодность чисто математических моделей для большинства объектов с переменной структурой;

- приближенные методы реализаций моделей с переменными коэффициентами требуют значительных затрат и не обладают достаточной точностью решения.

В настоящее время имитационное моделирование в основном реализуется на ЭВМ. Исходное математическое описание любой динамической системы представляет собой совокупность дифференциальных, алгебраических, логических, разностных уравнений, описывающих физические процессы в отдельных функциональных элементах системы.

Отметим, что никакое определение не может в полном объёме охватить реально существующую деятельность по математическому моделированию. Несмотря на это, определения полезны тем, что в них делается попытка выделить наиболее существенные черты.

Также, как не существует единой классификации моделей, так и нет единого, наиболее полного определения процесса математического моделирования объекта.

Так, по Ляпунову, математическое моделирование — это опосредованное практическое или теоретическое исследование объекта, при котором непосредственно

изучается не сам интересующий нас объект, а некоторая вспомогательная искусственная или естественная система (модель), находящаяся в некотором объективном соответствии с познаваемым объектом, способная замещать его в определенных отношениях и дающая при её исследовании, в конечном счете, информацию о самом моделируемом объекте.

В других вариантах, математическая модель определяется как объект-заместитель объекта-оригинала, обеспечивающий изучение некоторых свойств оригинала, как эквивалент объекта, отражающий в математической форме важнейшие его свойства – законы, которым он подчиняется, связи, присущие составляющим его частям, как систему уравнений, или арифметических соотношений, или геометрических фигур, или комбинацию того и другого, исследование которых средствами математики должно ответить на поставленные вопросы о свойствах некоторой совокупности свойств объекта реального мира, как совокупность математических соотношений, уравнений, неравенств, описывающих основные закономерности, присущие изучаемому процессу, объекту или системе.

Структурные модели представляют объект как систему со своим устройством и механизмом функционирования. Функциональные модели не используют таких представлений и отражают только внешне воспринимаемое поведение (функционирование) объекта. В их предельном выражении они называются также моделями «чёрного ящика». Возможны также комбинированные типы моделей, которые иногда называют моделями «серого ящика».

Математические модели сложных систем можно разделить на три типа:

- модели типа чёрный ящик;
- модели типа серый ящик;
- модели типа белый ящик.

Практически все авторы, описывающие процесс математического моделирования, указывают, что сначала строится особая идеальная конструкция, содержательная модель. Устоявшейся терминологии здесь нет, и другие авторы называют этот идеальный объект концептуальная модель, умозрительная модель или предмодель. При

этом финальная математическая конструкция называется формальной моделью или просто математической моделью, полученной в результате формализации данной содержательной модели (предмодели). Построение содержательной модели может производиться с помощью набора готовых идеализаций, как в механике, где идеальные пружины, твёрдые тела, идеальные маятники, упругие среды и т. п. дают готовые структурные элементы для содержательного моделирования. Однако в областях знания, где не существует полностью завершенных формализованных теорий, создание содержательных моделей резко усложняется.

Важнейшие математические модели обычно обладают важным свойством универсальности: принципиально разные реальные явления могут описываться одной и той же математической моделью. Скажем, гармонический осциллятор описывает не только поведение груза на пружине, но и другие колебательные процессы, зачастую имеющие совершенно иную природу: малые колебания маятника, колебания уровня жидкости в U-образном сосуде или изменение силы тока в колебательном контуре. Таким образом, изучая одну математическую модель, мы изучаем сразу целый класс описываемых ею явлений.

1.2 Задачи математического моделирования

Существует множество задач, связанных с математическим моделированием. Во-первых, надо придумать основную схему моделируемого объекта, воспроизвести его в рамках идеализаций данной науки. Традиционно выделяют два основных класса задач, связанных с математическими моделями: прямые и обратные.

Прямая задача: структура модели и все её параметры считаются известными, главная задача — провести исследование модели для извлечения полезного знания об объекте. Какую статическую нагрузку выдержит мост? Как он будет реагировать на динамическую нагрузку (например, на марш роты солдат, или на прохождение поезда на различной скорости), как самолёт преодолеет звуковой барьер, не развалится ли он

от флаттера, — вот типичные примеры прямой задачи. Постановка правильной прямой задачи (задание правильного вопроса) требует специального мастерства. Если не заданы правильные вопросы, то мост может обрушиться, даже если была построена хорошая модель для его поведения. В простейшем случае (одно уравнение осциллятора, например) прямая задача очень проста и сводится к явному решению этого уравнения.

Обратная задача: известно множество возможных моделей, надо выбрать конкретную модель на основании дополнительных данных об объекте. Чаще всего структура модели известна, и необходимо определить некоторые неизвестные параметры. Дополнительная информация может состоять в дополнительных эмпирических данных, или в требованиях к объекту (задача проектирования). Дополнительные данные могут поступать независимо от процесса решения обратной задачи (пассивное наблюдение) или быть результатом специально планируемого в ходе решения эксперимента (активное наблюдение).

1.3 Компьютерные системы моделирования

Для поддержки математического моделирования разработаны системы компьютерной математики, например, Maple, Mathematica, Mathcad, MATLAB, VisSim, МВТУ и др. Они позволяют создавать формальные и блочные модели как простых, так и сложных процессов, и устройств и легко менять параметры моделей в ходе моделирования. Блочные модели представлены блоками (чаще всего графическими), набор и соединение которых задаются диаграммой модели.

Нередко, простые модели и расчеты, которые также, по сути, являются моделями объектов, можно реализовать при помощи распространенных офисных пакетов, например, MS Office. При необходимости реализовать в офисном пакете итерационный процесс расчета прибегают к встроенным в него средствам программирования – Visual Basic for Application (VBA).

Однако, блочные модели обладают рядом преимуществ, наиболее значимыми из которых является простота составления модели и наглядность. Модель, построенная по такому принципу, становится похожей на обычную функциональную или технологическую схему. Довольно широкий функционал в области моделирования и оперирования блочными структурами моделей предоставляет пакет Simulink, входящий в структуру Matlab. Стоит заметить, что это коммерческий продукт, и для нужд студента покупка индивидуальной лицензии может оказаться неоправданно дорогой. Альтернативным решением в этом случае может стать установка бесплатно распространяемой программы отечественного производителя программных решений, под названием «МВТУ» («Моделирование в технических устройствах»).

Заметим, что принципы построения модели в Simulink Matlab и МВТУ мало отличаются. А основной результат зависит не столько от выбранного программного решения, сколько от выбранного и реализованного численного метода решения уравнений модели (далее – решателя). Изменение типа решателя и его настройка может привести к увеличению быстродействия ЭВМ при расчете модели, к увеличению точности моделирования, убережет исследователя от ложных выводов и умозаключений.

Контрольные вопросы и задачи к разделу 1

1) Дайте определения понятию «Математическая модель».

2) Приведите примеры использования статических моделей в инженерной практике.

3) Для чего нужны динамические модели? Приведите примеры их использования.

4) В чем отличие прямой и обратной задач моделирования?

5) По каким признакам можно классифицировать модели процессов и оборудования в химической технологии?

6) Приведите примеры использования табличных моделей в химической технологии.

7) Приведите пример использования графических моделей в инженерной практике.

8) Обязательно ли при составлении математической модели и её использовании на практике применять компьютерные средства и вычислительные машины?

9) Чем обуславливается выбор программной среды для моделирования?

10) Перечислите используемые в процессе Вашего обучения программные средства и комплексы, используемые для моделирования процессов и производств. Дайте краткую характеристику, укажите области применения, достоинства и недостатки при решении определенного Вами круга задач.

11) Какие методы получения исходных данных Вы можете указать?

12) Что такое пассивный эксперимент, в чем его достоинство, какие недостатки?

13) В каких случаях для получения экспериментальных данных и зависимостей целесообразно использовать метод постановки активного эксперимента?

14) Что такое адекватность модели и как её оценить?

15) Проиллюстрируйте свойство универсальности обобщенных моделей на конкретных примерах.

2 Построение статистических моделей, обработка экспериментальных данных

Статистические модели строятся по обширному эмпирическому материалу (например, по данным технологических рапортов, по данным протокола испытания и т.д.) и позволяют получить математическое выражение, связывающее предполагаемые независимые факторы и выходной параметр объекта или процесса. С развитием компьютерных технологий и техники построение таких моделей и обработка больших массивов экспериментальных данных не представляет большой сложности и трудоемкости. Но, такой подход позволяет описать точно только объекты, попавшие в исследование, все выводы и зависимости нельзя распространять на однотипную группу существующих объектов или процессов. Тем не менее, простота построения и использования таких моделей обуславливает их широкую применимость.

2.1 Линейная регрессия

Рассмотрим простейший случай уравнения регрессии – линейную регрессию, когда уравнение имеет вид прямой линии: y = ax+b. Можно показать, что в соответствии с методом наименьших квадратов для нахождения неизвестных параметров a и b нужно использовать следующие формулы:

$$a = \frac{\overline{xy} - \bar{x} \cdot \bar{y}}{\overline{x^2} - \bar{x}^2}, \quad (1)$$

$$b = \bar{y} - a \cdot \bar{x}, \quad (2)$$

$$\bar{x} = \frac{x_1 + x_2 + \cdots + x\_n}{n}, \quad (3)$$

$$\bar{y} = \frac{y_1 + y_2 + \cdots + y\_n}{n}, \quad (4)$$

$$\overline{xy} = \frac{x_1 \cdot y_1 + x_2 \cdot y_2 + \cdots + x_n \cdot y_n}{n}, \quad (5)$$

$$\overline{x^2} = \frac{x_1^2 + x_2^2 + \cdots + x_n^2}{n}, \quad (6)$$

где n –количество опытов (серийность эксперимента).

Для проверки полученных результатов можно построить график, на который наносятся исходные точки и линия регрессии. Рассмотрим теперь вопрос оценки качества статистической связи. Мерой оценки силы статистической зависимости между показателями x и y служит коэффициент парной корреляции r_{xy}, который в случае линейной связи между факторами вычисляется по формуле:

$$r_{xy} = \frac{\overline{xy} - \bar{x} \cdot \bar{y}}{\sqrt{(\overline{x^2} - \bar{x}^2) \cdot (\overline{y^2} - \bar{y}^2)}}, \quad (7)$$

где

$$\overline{y^2} = \frac{y_1^2 + y_2^2 + \cdots + y_n^2}{n} \quad (8)$$

Для ответа на вопрос: можно ли считать связь между показателями достаточно сильной, чтобы считать x и y зависимыми и уравнение их регрессии имело смысл, используется методика проверки значимости коэффициента корреляции. При проверке исследователем задается некоторая маленькая вероятность α, называемая уровнем значимости. Он имеет смысл вероятности совершить ошибку, заключающуюся в принятии предположения о независимости показателей, в то время, когда они зависимы. Вместо α иногда задают величину p = 1−α, называемую доверительной вероятностью. Ее можно интерпретировать как вероятность, с которой можно доверять полученному результату. На практике обычно выбирают α = 0,1, 0,05, 0,01. Далее, задав α или p, для случая парной линейной регрессии вычисляется величина t по формуле:

$$t = |r_{xy}| \cdot \sqrt{\frac{n-2}{1-r_{xy}^2}} \quad (9)$$

По специальной таблице, называемой критическими значениями распределения Стьюдента, которая приведена ниже (таблица 2), определяется критическое значение данной величины $t_{kr} = t((1-\alpha); (n-2))$.

Таблица 2 – Критические значения распределения Стьюдента

n	**p**							
	0,80	**0,90**	**0,95**	**0,98**	**0,99**	**0,995**	**0,998**	**0,999**
1	3,0770	6,3130	12,7060	31,820	63,656	127,656	318,306	636,619
2	1,8850	2,9200	4,3020	6,964	9,924	14,089	22,327	31,599
3	1,6377	2,35340	3,182	4,540	5,840	7,458	10,214	12,924
4	1,5332	2,13180	2,776	3,746	4,604	5,597	7,173	8,610
5	1,4759	2,01500	2,570	3,649	4,0321	4,773	5,893	6,863
6	1,4390	1,943	2,4460	3,1420	3,7070	4,316	5,2070	5,958
7	1,4149	1,8946	2,3646	2,998	3,4995	4,2293	4,785	5,4079
8	1,3968	1,8596	2,3060	2,8965	3,3554	3,832	4,5008	5,0413
9	1,3830	1,8331	2,2622	2,8214	3,2498	3,6897	4,2968	4,780
10	1,3720	1,8125	2,2281	2,7638	3,1693	3,5814	4,1437	4,5869
11	1,363	1,795	2,201	2,718	3,105	3,496	4,024	4,437
12	1,3562	1,7823	2,1788	2,6810	3,0845	3,4284	3,929	4,178
13	1,3502	1,7709	2,1604	2,6503	3,1123	3,3725	3,852	4,220
14	1,3450	1,7613	2,1448	2,6245	2,976	3,3257	3,787	4,140
15	1,3406	1,7530	2,1314	2,6025	2,9467	3,2860	3,732	4,072
16	1,3360	1,7450	2,1190	2,5830	2,9200	3,2520	3,6860	4,0150
17	1,3334	1,7396	2,1098	2,5668	2,8982	3,2224	3,6458	3,965
18	1,3304	1,7341	2,1009	2,5514	2,8784	3,1966	3,6105	3,9216
19	1,3277	1,7291	2,0930	2,5395	2,8609	3,1737	3,5794	3,8834
20	1,3253	1,7247	2,08600	2,5280	2,8453	3,1534	3,5518	3,8495
21	1,3230	1,7200	2,2.0790	2,5170	2,8310	3,1350	3,5270	3,8190
22	1,3212	1,7117	2,0739	2,5083	2,8188	3,1188	3,5050	3,7921
23	1,3195	1,7139	2,0687	2,4999	2,8073	3,1040	3,4850	3,7676
24	1,3178	1,7109	2,0639	2,4922	2,7969	3,0905	3,4668	3,7454
25	1,3163	1,7081	2,0595	2,4851	2,7874	3,0782	3,4502	3,7251
26	1,315	1,705	2,059	2,478	2,778	3,0660	3,4360	3,7060
27	1,3137	1,7033	2,0518	2,4727	2,7707	3,0565	3,4210	3,6896
28	1,3125	1,7011	2,0484	2,4671	2,7633	3,0469	3,4082	3,6739
29	1,3114	1,6991	2,0452	2,4620	2,7564	3,0360	3,3962	3,8494
30	1,3104	1,6973	2,0423	2,4573	2,7500	3,0298	3,3852	3,6460
32	1,3080	1,6930	2,0360	2,4480	2,7380	3,0140	3,3650	3,6210
34	1,3070	1,6909	2,0322	2,4411	2,7284	3,9520	3,3479	3,6007
36	1,3050	1,6883	2,0281	2,4345	2,7195	9,490	3,3326	3,5821

Продолжение таблицы 2

n	p							
	0,80	**0,90**	**0,95**	**0,98**	**0,99**	**0,995**	**0,998**	**0,999**
38	1,3042	1,6860	2,0244	2,4286	2,7116	3,9808	3,3190	3,5657
40	1,303	1,6839	2,0211	2,4233	2,7045	3,9712	3,3069	3,5510
42	1,320	1,682	2,018	2,418	2,6980	2,6930	3,2960	3,5370
44	1,301	1,6802	2,0154	2,4141	2,6923	3,9555	3,2861	3,5258
46	1,300	1,6767	2,0129	2,4102	2,6870	3,9488	3,2771	3,5150
48	1,299	1,6772	2,0106	2,4056	2,6822	3,9426	3,2689	3,5051
50	1,298	1,6759	2,0086	2,4033	2,6778	3,9370	3,2614	3,4060
55	1,2997	1,673	2,0040	2,3960	2,6680	2,9240	3,2560	3,4760
60	1,2958	1,6706	2,0003	2,3901	2,6603	3,9146	3,2317	3,4602
65	1,2947	1,6686	1,997	2,3851	2,6536	3,9060	3,2204	3,4466
70	1,2938	1,6689	1,9944	2,3808	2,6479	3,8987	3,2108	3,4350
80	1,2820	1,6640	1,9900	2,3730	2,6380	2,8870	3,1950	3,4160
90	1,2910	1,6620	1,9867	2,3885	2,6316	2,8779	3,1833	3,4019
100	1,2901	1,6602	1,9840	2,3642	2,6259	2,8707	3,1737	3,3905
120	1,2888	1,6577	1,9719	2,3578	2,6174	2,8598	3,1595	3,3735
150	1,2872	1,6551	1,9759	2,3515	2,6090	2,8482	3,1455	3,3566
200	1,2858	1,6525	1,9719	2,3451	2,6006	2,8385	3,1315	3,3398
250	1,2849	1,6510	1,9695	2,3414	2,5966	2,8222	3,1232	3,3299
300	1,2844	1,6499	1,9679	2,3388	2,5923	2,8279	3,1176	3,3233
400	1,2837	1,6487	1,9659	2,3357	2,5882	2,8227	3,1107	3,3150

Если $t > t_{kr}$, то можно считать, что коэффициент корреляции значим, показатели x и y зависимы, уравнение регрессии можно использовать для прогнозов и оценок. Если $t \leq t_{kr}$, то коэффициент корреляции незначим, показатели x и y независимы, уравнение регрессии теряет смысл.

2.2 Парная нелинейная модель

Во многих случаях при решении технических задач оказывается, что зависимость между показателями x и y не является пропорциональной. В этом случае нельзя использовать линейное регрессионное уравнение, а необходимо строить парную нелинейную модель. По методике построения нелинейные модели делятся на два класса: внутренне линейные и внутренне нелинейные. Внутренне линейные модели

можно с помощью алгебраических преобразований данных привести к линейному виду. Например, рассмотрим случай гиперболической регрессии y = a/x + b. Сделаем замену переменной X = 1/x, в результате получим линейное уравнение: y = aX + b. Таким образом, необходимо вместо исходных данных (x_i, y_i) нужно взять преобразованные данные ($1/x_i$, y_i), а затем с ними проделать все вычисления, присущие парной линейной регрессии. Коэффициент корреляции вычисляется также, как и для линейной регрессии по формуле (7), но по преобразованным исходным данным. Проверка на значимость выполняется по формуле (9). По окончании регрессионного анализа и нахождения преобразованных коэффициентов следует выполнить обратные преобразования и записать уравнение регрессии нелинейной модели.

Рассмотрим теперь ситуацию, когда нельзя никакими преобразованиями привести уравнение к линейному виду. Такие модели называются внутренне нелинейными. Неизвестные параметры таких моделей получают непосредственно по методу наименьших квадратов. Среди внутренне нелинейных моделей в технике чаще всего используется параболическая регрессия, уравнение которой имеет вид:

$$y = a \cdot x^2 + b \cdot x + c \tag{10}$$

В соответствии с методом наименьших квадратов, для нахождения неизвестных параметров a, b и c необходимо решать систему уравнений вида:

$$\begin{cases} a \cdot \sum x_i^4 + b \cdot \sum x_i^3 + c \cdot \sum x_i^2 = \sum (y_i \cdot x_i^2) \\ a \cdot \sum x_i^3 + b \cdot \sum x_i^2 + c \cdot \sum x_i = \sum (y_i \cdot x_i) \\ a \cdot \sum x_i^2 + b \cdot \sum x_i + c \cdot n = \sum y_i \end{cases}, \tag{11}$$

где указанные суммы вычислены по всем эмпирическим данным:

$$\sum x_i^4 = x_1 + x_2 + \cdots + x_n, \tag{12}$$

$$\sum (y_i \cdot x_i^2) = y_1 \cdot x_1^2 + y_2 \cdot x_2^2 + \cdots + y_n \cdot x_n^2 \tag{13}$$

и так далее.

Для вычисления нелинейного коэффициента парной корреляции в случае параболической регрессии используется формула:

$$r_{xy} = \sqrt{(1 - \frac{\sum_{i=1}^{n}(y(x_i) - y_i)^2}{\sum_{i=1}^{n}(y_i - \bar{y_i})^2})} \quad (14)$$

Для проверки значимости вычисляется критерий Стьюдента:

$$t = r_{xy} \cdot \sqrt{\frac{n-3}{1-r_{xy}^2}}, \quad (15)$$

который сравнивается с $t_{kr} = t((1-\alpha); (n-3))$, значение которого табулировано.

Если $t > t_{kr}$, то можно считать, что коэффициент корреляции значим, показатели x и y зависимы, уравнение регрессии можно использовать для прогнозов и оценок.

2.3 Множественная линейная регрессия

В рассмотренных ранее задачах на фактор y влиял только один фактор x, а влияние всех остальных было мало и приводило к случайному разбросу значений y. Однако, часто на результирующий фактор y достаточно сильно может влиять сразу несколько других факторов. Если на переменную y в равной степени влияют несколько независимых переменных, то такая зависимость описывается множественной регрессией. Переменная y при этом называется результирующим признаком или результатом, а остальные, влияющие на него показатели – независимыми факторами.

Рассмотрим случай, когда независимые переменные входят в уравнение регрессии линейно. Такая множественная регрессия называется линейной. Рассмотрим простейший случай линейной множественной регрессии – двухфакторную регрессию. В этом случае на результат y влияет два фактора: x_1 и x_2. Для такого случая уравнение регрессии имеет вид:

$$y = a_1 \cdot x_1 + a_2 \cdot x_2 + b \tag{16}$$

Для нахождения неизвестных параметров уравнения a_1, a_2, b, в соответствии с методом наименьших квадратов, необходимо решать систему линейных уравнений вида:

$$\begin{cases} a_1 \cdot \sum(x_1 \cdot x_2) + a_2 \cdot \sum x_2^2 + b \cdot \sum x_2 = \sum(x_2 \cdot y) \\ a_1 \cdot \sum x_1^2 + a_2 \cdot \sum(x_1 \cdot x_2) + b \cdot \sum x_1 = \sum(x_1 \cdot y) \\ a_1 \cdot \sum x_1 + a_2 \cdot \sum x_2 + b \cdot n = \sum y \end{cases}, \tag{17}$$

где суммы – параметры системы уравнений - находятся, как и в случае параболической регрессии, по всем эмпирическим данным.

Для оценки качества уравнения регрессии используются парные коэффициенты корреляции, которые вычисляются по формулам, аналогичным (7):

$$r_{x_1 y} = \frac{\overline{x_1 \cdot y} - \overline{x_1} \cdot \bar{y}}{\sqrt{(\overline{x_1^2} - \overline{x_1}^2) \cdot (\overline{y^2} - \bar{y}^2)}} \tag{18}$$

$$r_{x_2 y} = \frac{\overline{x_2 \cdot y} - \overline{x_2} \cdot \bar{y}}{\sqrt{(\overline{x_2^2} - \overline{x_2}^2) \cdot (\overline{y^2} - \bar{y}^2)}} \tag{19}$$

$$r_{x_1 x_2} = \frac{\overline{x_1 \cdot x_2} - \overline{x_1} \cdot \overline{x_2}}{\sqrt{(\overline{x_1^2} - \overline{x_1}^2) \cdot (\overline{x_2^2} - \overline{x_2}^2)}} \tag{20}$$

Коэффициенты $r_{x_1 y}$ и $r_{x_2 y}$ характеризуют влияние каждого фактора x_1 и x_2 на результат y. Если влияние одного из факторов на результат мало, соответствующий коэффициент корреляции невысок (грубо можно считать малым коэффициент, меньший 0,6). Следовательно, этот фактор слабо влияет на результат и его можно исключить из модели. Коэффициент $r_{x_1 x_2}$ характеризует влияние факторов друг на друга. Если это влияние высоко, то это негативный признак, т.к. факторы x_1 и x_2 должны

быть независимыми. Если независимые факторы влияют друг на друга, то они называются интеркоррелированными или мультиколлинеарными и их наличие ухудшает качество регрессионной модели.

Если факторы зависимы, то нет смысла их оба включать в модель, необходимо либо оставить в модели один из факторов (тот, который сильнее влияет на результат), либо объединить оба фактора в один общий, в обоих случаях получив парную регрессию.

2.4 Пример построения регрессионной модели в MS Excel

Составим уравнение регрессии для зависимости плотности серной кислоты при 20 °С от её концентрации в растворе. Для этого воспользуемся справочными данными таблицы 3 и перенесем эти данные в MS Excel.

Выберем: «Вставка» - «Диаграммы» – «Точечная с гладкими кривыми и маркерами» для того, чтобы создать макет диаграммы. В области диаграммы – щелчок правой кнопкой мыши и в контекстном меню выбираем пункт «Выбрать данные», затем в поле «Элементы легенды (ряды)» необходимо нажать кнопку «Добавить» и указать имя ряда (например, «С(р)») и указать диапазоны значений переменных «х» и «у». В качестве аргумента функции укажем процентные концентрации, а в качестве значений функции укажем область значений плотности растворов. После этого нажмем «Ok» и закроем окно выбора данных. На макете диаграммы теперь отрисована кривая зависимости концентрации от плотности.

Найдем уравнение регрессии, описывающее поведение данной кривой. Для этого выделим кривую, вызовем контекстное меню и выберем пункт «Добавить линию тренда». После чего остается только выбрать предполагаемую форму зависимости и отметить пункты «показывать уравнение на диаграмме» и «поместить на диаграмму величину достоверности аппроксимации (R^2)».

Результаты манипуляций с различными типами кривых показаны на рисунках (1) и (2).

Таблица 3 – Плотность серной кислоты (при 20 °С) при различных концентрациях

Плотность, г/мл	Концентрация		
	%	г/л	моль/л
1,0051	1	10,05	0,103
1,0118	2	20,24	0,206
1,0184	3	30,55	0,312
1,025	4	41	0,418
1,0317	5	51,59	0,526
1,0385	6	62,31	0,635
1,0453	7	73,17	0,746
1,0522	8	84,18	0,858
1,0591	9	95,32	0,972
1,0661	10	106,6	1,087
1,0731	11	118	1,203
1,0802	12	129,6	1,321
1,0874	13	141,4	1,442
1,0947	14	153,3	1,563
1,102	15	165,3	1,685
1,1094	16	177,5	1,81
1,1168	17	189,9	1,936
1,1243	18	202,4	2,063
1,1318	19	215	2,192

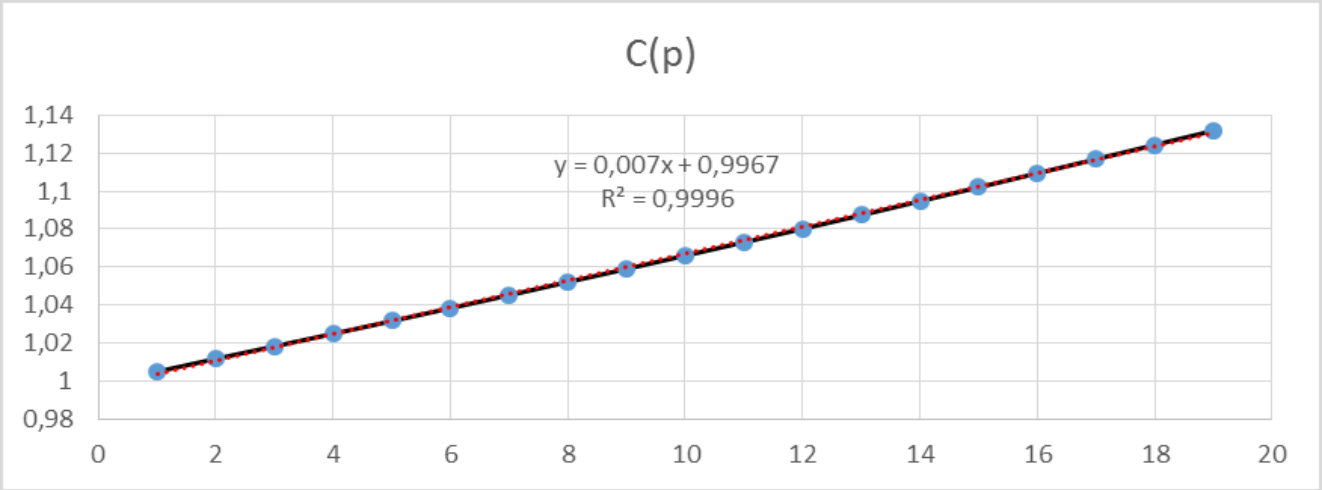


Рисунок 1 – Аппроксимация зависимости линейной функцией

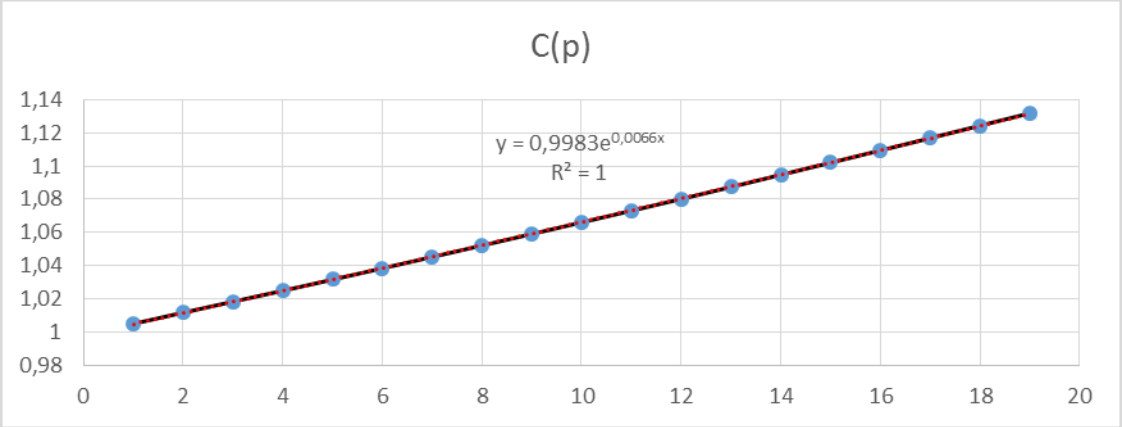


Рисунок 2 – Аппроксимация зависимости экспоненциальной функцией

Как видно из рисунков, достоверность аппроксимации при использовании экспоненциальной модели выше, однако усложнение модели не всегда может быть оправдано. Для грубого, прикидочного расчета вполне подходящей и простой является именно линейная модель.

2.5 Пример построения регрессионной модели в Matlab

Для выполнения задач аппроксимации в программном комплексе Matlab имеется множество возможностей, организованных в подпрограммы (Toolbox-ы). Наиболее простым решением поставленной ранее задачи по поиску регрессионной зависимости между плотностью раствора серной кислоты и концентрацией (данные представлены в таблице 3) является использование «Curve Fitting Toolbox».

Curve Fitting Toolbox - это пакет расширения Matlab для различных прикладных задач подгонки, аппроксимации и интерполяции данных.

Включает в себя интерактивные средства предварительной обработки данных, сравнения стандартных моделей и разработки моделей пользователя, подгонки с помощью стандартных и робастных методов и анализа качества аппроксимации.

Curve Fitting Toolbox содержит полиномиальную модель подгонки, экспоненциальную, Гауссову, Фурье и др. и включает средства идентификации вылетов в данных, а также функции сглаживания Савитского–Голея и скользящего среднего.

Matlab позволяет импортировать данные в Curve Fitting Toolbox из большого числа файловых форматов.

Все функции Curve Fitting Toolbox написаны на открытом языке Matlab, что позволяет пользователю контролировать исполнение алгоритмов, изменять исходный код, а также создавать свои собственные функции и процедуры.

В отличие от MS Excel, имеется ряд особенностей, значительно расширяющих возможности поиска экспериментальных зависимостей, а именно:

- интерактивный интерфейс пользователя, унифицирующий задачи подгонки и обработки данных;

- предварительная обработка данных, включая масштабирование, секционирование, сглаживание и определение вылетов;

- обширная библиотека моделей, линейных и нелинейных методов подгонки, оптимизированный алгоритм поиска стартовой точки;

- метод наименьших квадратов, взвешенных наименьших квадратов, методы робастной оценки;

- разработка пользовательских линейных и нелинейных моделей подгонки;

- непараметрическая подгонка с использованием сплайнов, интерполяции и регрессии;

- интерполяция, экстраполяция, дифференцирование и интегрирование модели.

Curve Fitting Toolbox имеет наиболее широкий набор методов для моделирования кривых и поверхностей, среди них линейные и нелинейные регрессии, интерполяционные и сглаживающие сплайны. Инструментарий имеет опции для робастной регрессии, позволяющей аппроксимировать данные, имеющие выбросы. Все алгоритмы могут быть доступны из командной строки, а также с помощью графического интерфейса.

Построение моделей с графическим интерфейсом позволяет осуществить:

- импорт данных из рабочего пространства Matlab;

- выполнить визуализацию данных для анализа;

- создание модели с помощью алгоритма множественной подгонки;

- оценку точности модели;

- создание доверительных интервалов, вычисление интегралов и производных;

- экспорт моделей в рабочую среду Matlab для дальнейшего анализа, доступна автоматическая генерация Matlab кода для сохранения результатов и автоматизации процесса моделирования.

Для опытных пользователей зачастую удобнее использовать интерфейс командной строки. Работа в командной строке позволяет разрабатывать пользовательские функции для анализа и визуализации. Эти функции предоставляют возможность:

- дублировать анализ с новым набором данных;

- реплицировать анализ с несколькими наборами данных (пакетная обработка);

- вставлять процедуру подгонки в функцию Matlab;

- расширять базовые возможности инструмента.

Curve Fitting Toolbox предоставляет простой интуитивно понятный синтаксис командной строки для подгонки данных. В качестве демонстрации приведены следующие примеры:

- линейная регрессия: fittedmodel = fit([X,Y], Z, 'poly11');

- нелинейная регрессия: fittedmodel = fit(X, Y, 'fourier2');

- интерполяция: fittedmodel = fit([Time,Temperature], Energy, 'cubicinterp');

- сглаживание: fittedmodel = fit([Time,Temperature], Energy, 'lowess', 'span', 0,12).

Результат созданной модели сохраняется в объект «fittedmodel». Постобработка анализа, такая как построение модели, оценка, расчет интегралов и производных могут быть выполнены с применением метода к этому объекту, как в следующих примерах:

- визуализация: plot(fittedmodel);

- дифференцирование: differentiate(fittedmodel, X, Y);

- оценка: fittedmodel(80, 40).

Curve Fitting Toolbox позволяет перейти от графического интерфейса к командной строке. Используя графический интерфейс, можно создать функции Matlab, которые дублируют любой анализ, проделанный с помощью графического интерфейса. Вы также можете создать объект модели в графическом интерфейсе и экспортировать его в рабочую область Matlab для дальнейшего анализа.

После выбора кривой или поверхности, которая наилучшим образом описывает данные, имеется возможность сделать постобработку. Curve Fitting Toolbox позволяет:

- строить графики;

- использовать созданную модель для оценки значений;

- вычислять доверительные интервалы;

- создавать границы прогноза;

- осуществлять определение площади кривой (интегрирование);

- выполнять расчет производных.

Покажем, каким образом может быть выполнена постобработка из командной строки. Команды выполняются над объектом модели, созданным с помощью функций Curve Fitting Tool:

- оценка: EnergyConsumption = fittedmodel(X, Y);

- визуализация: EnergySurface = plot(fittedmodel);

- интегрирование: Volume_Under_Surface = quad2d(fittedmodel, Min_X, Max_X, Min_Y, Max_Y);

- дифференцирование: Gradient = differentiate(fittedmodel, X,Y);

- расчет доверительных интервалов: Confidence_Intervals = confint(fittedmodel).

И так, решим задачу аппроксимации данных таблицы 3 при помощи графических средств выбранного и описанного Toolbox-а. Для начала создадим переменную, содержащую исходные данные. В качестве концентрации также будем использовать процентное выражение содержания вещества в растворе. Для создания переменной выбираем пункт контекстного меню «New» окна «Workspace», переименовываем переменную командой «Rename» контекстного меню, присваивая ей нужное нам имя

(по умолчанию создается переменная unnamed). Зададим имя переменной, например, «C». Аналогичным образом задаем переменную, в которой будем хранить значения плотности растворов, назовем её «P».

Двойной щелчок левой кнопки мыши на переменной открывает редактор переменных. помещаем наши данные в первом столбце каждой переменной (рисунок 3).

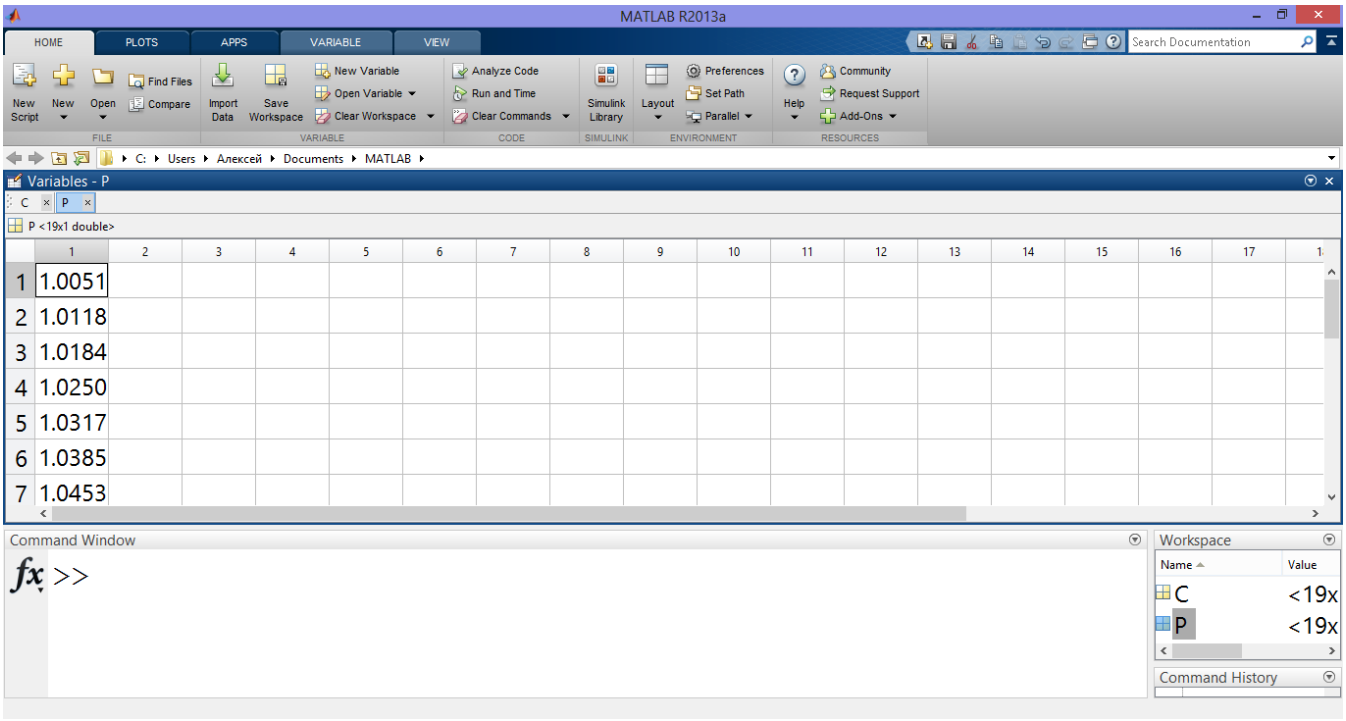


Рисунок 3 – Заполнение переменной исходными данными в Matlab

После выполненных процедур окно редактора переменных можно закрыть. Далее обращаемся к использованию Curve Fitting Tool. В командной строке набираем команду «cftool», запуская графический компонент анализа данных (рисунок 4).

В окне Curve Fitting Tool задаем имена:

1) аппроксимационной модели (в нашем случае зададим имя «CP»);

2) в полях «X data» и «Y data» указываем переменные «C» и «P»;

3) отмечаем «Auto fit»;

4) выбираем нужный тип кривой аппроксимации, например, «Exponential»;

5) в поле «Results» наблюдаем статистические сведения о достоверности модели и уравнение регрессии (рисунок 5)

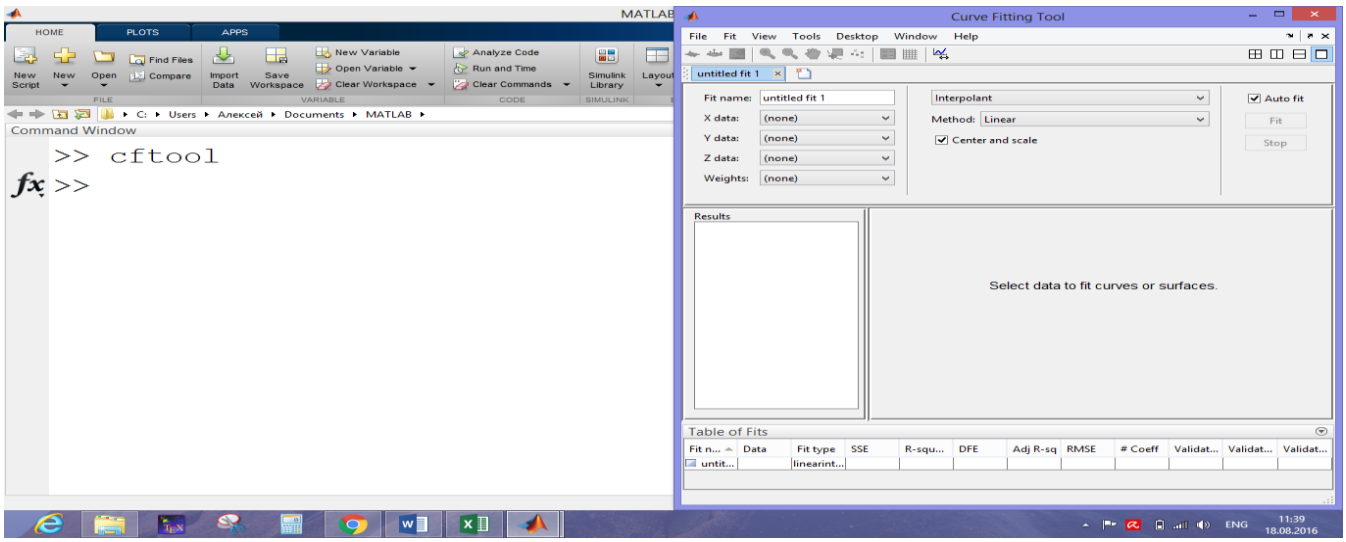


Рисунок 4 – Запуск и окно Curve Fitting Tool

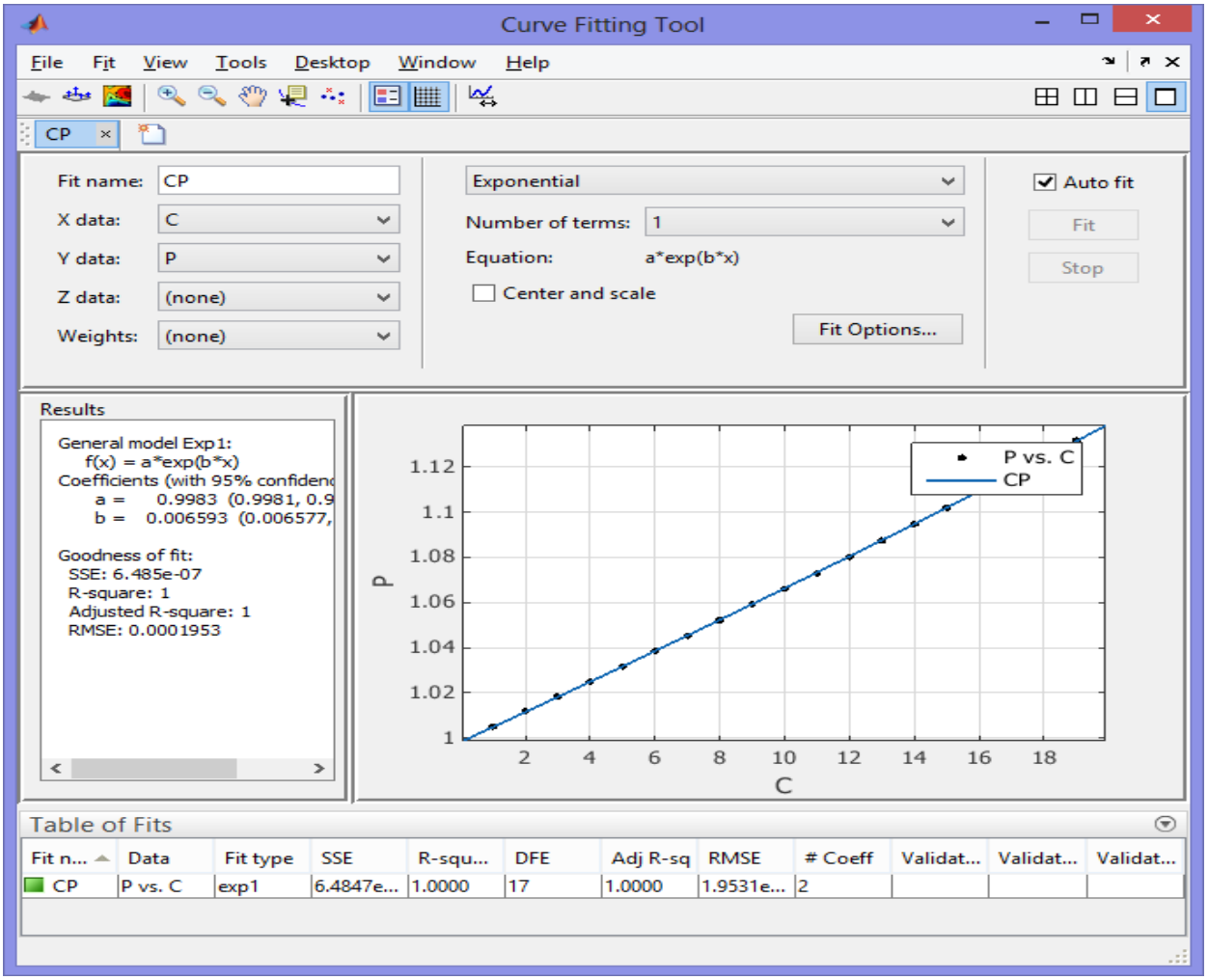


Рисунок 5 – Результат работы Curve Fitting Tool

Как видно на рисунке 5, коэффициенты уравнения не отличаются от полученных ранее с помощью MS Excel, однако помимо величины коэффициента достоверности аппроксимации в окне результатов нам доступны также сведения о дисперсии и среднеквадратичных отклонениях, что позволяет выбрать наиболее подходящую модель из множества моделей, дающих удовлетворительный результат.

2.6 Применение нечеткой логики в моделировании. Fuzzy-logic модели

В сочетании слов «нечеткий» и «логика» есть что-то необычное. Логика в обычном смысле слова есть представление механизмов мышления, то, что никогда не может быть нечетким, но всегда строгим и формальным. Однако математики, исследовавшие эти механизмы мышления, заметили, что в действительности существует не одна логика (например, булева), а столько, сколько мы пожелаем, потому что все определяется выбором соответствующей системы аксиом. Конечно, как только аксиомы выбраны, все утверждения, построенные на их основе, должны быть строго, без противоречий увязаны друг с другом согласно правилам, установленным в этой системе аксиом. Человеческое мышление — это совмещение интуиции и строгости, которое, с одной стороны, рассматривает мир в целом или по аналогии, а с другой стороны — логически и последовательно и, значит, представляет собой нечеткий механизм. Законы мышления, которые мы захотели бы включить в программы компьютеров, должны быть обязательно формальными; законы мышления, проявляемые в диалоге человека с человеком — нечеткие. Можем ли мы поэтому утверждать, что нечеткая логика может быть хорошо приспособлена к человеческому диалогу? Да — если математическое обеспечение, разработанное с учетом нечеткой логики, станет операционным и сможет быть технически реализовано, то человеко-машинное общение станет намного более удобным, быстрым и лучше приспособленным к решению проблем. Термин «нечеткая логика» используется обычно в двух различных значениях.

Нечеткая логика — это логическое исчисление, являющееся расширением многозначной логики.

Нечеткая логика равнозначна теории нечетких множеств. С этой точки зрения, нечеткая логика в узком смысле является разделом нечеткой логики в широком смысле.

Таким образом, путем несложных рассуждений можно сделать следующее заключение: предметом нечёткой логики является построение моделей приближенных рассуждений человека и использование их в компьютерных системах

В настоящее время существует по крайней мере два основных направления научных исследований в области нечёткой логики: нечёткая логика в широком смысле (теория приближенных вычислений); нечёткая логика в узком смысле (символическая нечёткая логика).

Характеристикой нечеткого множества выступает функция принадлежности (Membership Function). Обозначим через MFc(x) – степень принадлежности к нечеткому множеству C, представляющей собой обобщение понятия характеристической функции обычного множества. Тогда нечетким множеством C называется множество упорядоченных пар вида C = MFc(x)/x, где MFc(x) располагается в интервале [0,1]. Значение MFc(x) = 0 означает отсутствие принадлежности к множеству, 1 – полную принадлежность. Проиллюстрируем это на простом примере. Формализуем неточное определение «горячий раствор». В качестве x (область рассуждений) будет выступать шкала температуры в градусах Цельсия. Очевидно, что она будет изменяется от 0 до 100 градусов. Нечеткое множество для понятия «горячий раствор» может выглядеть следующим образом:

C = {0/0;0/10;0/20;0,15/30;0,30/40;0,60/50;0,80/60;0,90/70;1/80;1/90;1/100}.

Так, раствор с температурой 60 °C принадлежит к множеству «горячий» со степенью принадлежности 0,80. Для одного состава раствор при температуре 60 °C может оказаться горячим, для другого – не слишком горячим. Именно в этом и проявляется нечеткость задания соответствующего множества. Для нечетких множеств, как

и для обычных, определены основные логические операции. Самыми основными, необходимыми для расчетов, являются пересечение и объединение.

Пересечение двух нечетких множеств (нечеткое «И»):

$$AB: MFAB(x) = \min(MFA(x),MFB(x)).$$

Объединение двух нечетких множеств (нечеткое «ИЛИ»):

$$AB: MFAB(x) = \max(MFA(x),MFB(x)).$$

В теории нечетких множеств разработан общий подход к выполнению операторов пересечения, объединения и дополнения, реализованный в так называемых треугольных нормах и конормах. Приведенные выше реализации операций пересечения и объединения – наиболее распространенные случаи t-нормы и t-конормы. Для описания нечетких множеств вводятся понятия нечеткой и лингвистической переменных. Нечеткая переменная описывается набором (N, X, A), где N – это название переменной, X – универсальное множество (область рассуждений), A – нечеткое множество на X. Значениями лингвистической переменной могут быть нечеткие переменные, т.е. лингвистическая переменная находится на более высоком уровне, чем нечеткая переменная.

Каждая лингвистическая переменная состоит из: названия; множества своих значений, которое также называется базовым терм-множеством T. Элементы базового терм-множества представляют собой названия нечетких переменных; универсального множества X; синтаксического правила G, по которому генерируются новые термы с применением слов естественного или формального языка; семантического правила P, которое каждому значению лингвистической переменной ставит в соответствие нечеткое подмножество множества X. Рассмотрим такое нечеткое понятие как «Производительность установки». Это и есть название лингвистической переменной. Сформируем для нее базовое терм-множество, которое будет состоять из трех нечетких переменных:

«Низкая»;

«Умеренная»;

«Высокая».

Зададим область рассуждений в виде X = [100;200] (единиц). Последнее, что осталось сделать – построить функции принадлежности для каждого лингвистического терма из базового терм-множества T. Существует свыше десятка типовых форм кривых для задания функций принадлежности. Наибольшее распространение получили: треугольная; трапецеидальная; гауссова функции принадлежности.

Треугольная функция принадлежности

При (b − a) = (c − b) имеем случай симметричной треугольной функции принадлежности, которая может быть однозначно задана двумя параметрами из тройки (a, b, c).

Трапецеидальная функция принадлежности

При (b − a) = (d − c) трапецеидальная функция принадлежности принимает симметричный вид.

Гауссова функция принадлежности

Гауссова функция принадлежности оперирует двумя параметрами. Параметр «c» обозначает центр нечеткого множества, «a» – параметр отвечает за крутизну функции.

Основой для проведения операции нечеткого логического вывода является база правил, содержащая нечеткие высказывания в форме «Если-то» и функции принадлежности для соответствующих лингвистических термов. При этом должны соблюдаться следующие условия:

1) существует хотя бы одно правило для каждого лингвистического терма выходной переменной;

2) для любого терма входной переменной имеется хотя бы одно правило, в котором этот терм используется в качестве предпосылки (левая часть правила).

Пусть в базе правил имеется m правил вида:

$$R_1: \text{ЕСЛИ } \mathbf{x_1} \text{ это } \mathbf{A_{11}} \text{ ...И ...} \mathbf{x_n} \text{ это } \mathbf{A_{1n}}, \text{ ТО } \mathbf{y} \text{ это } \mathbf{B_1}$$

$$R_i: \text{ЕСЛИ } \mathbf{x_1} \text{ это } \mathbf{A_{i1}} \text{ ...И ...} \mathbf{x_n} \text{ это } \mathbf{A_{in}}, \text{ ТО } \mathbf{y} \text{ это } \mathbf{B_i}$$

$$R_m: \text{ЕСЛИ } \mathbf{x_1} \text{ это } A_{i1} \text{ ...И ...} \mathbf{x_n} \text{ это } \mathbf{A_{mn}}, \text{ ТО } \mathbf{y} \text{ это } \mathbf{B_m},$$

где x_k при $k = 1..n$ – входные переменные;

y – выходная переменная;

A_{ik} – заданные нечеткие множества с функциями принадлежности.

Результатом нечеткого вывода является четкое значение переменной «y^*» на основе заданных четких значений x_k при $k = 1..n$.

В общем случае механизм логического вывода включает четыре этапа:

1) введение нечеткости (фазификация);

2) нечеткий вывод;

3) композиция

4) приведение к четкости (дефазификация).

Алгоритмы нечеткого вывода различаются главным образом видом используемых правил, логических операций и разновидностью метода дефазификации. Разработаны модели нечеткого вывода Мамдани, Сугено, Ларсена, Цукамото. Рассмотрим подробнее нечеткий вывод на примере механизма Мамдани (Mamdani). Это наиболее распространенный способ логического вывода в нечетких системах. В нем используется минимаксная композиция нечетких множеств. Данный механизм включает в себя следующую последовательность действий.

1 Процедура фазификации: определяются степени истинности, то есть значения функций принадлежности для левых частей каждого правила (предпосылок). Для базы правил с m правилами обозначим степени истинности как $A_{ik}(xk)$, i = 1..m, k = 1..n.

2 Нечеткий вывод. Сначала определяются уровни «отсечения» для левой части каждого из правил. Далее находятся «усеченные» функции принадлежности.

3 Композиция, или объединение полученных усеченных функций, для чего используется максимальная композиция нечетких множеств.

4 Дефазификация, или приведение к четкости. Существует несколько методов дефазификации. Например, метод среднего центра, или центроидный метод.

Однако, рассмотренные алгоритмы нечеткого вывода Мамдани не охватывают весь спектр решаемых в технике задач. Не менее значимыми для решения технических задач моделирования представляются нечеткие нейронные сети – модели Такаги – Сугено - Канга (TKS). Схема вывода при наличии M правил и N переменных X_i имеет вид (i = 1,2,...M):

ЕСЛИ $\mathbf{x_1}$ это $\mathbf{A_{1i}}$ И $\mathbf{x_2}$ это $\mathbf{A_{2i}}$ ... И $\mathbf{x_N}$ это $\mathbf{A_{Ni}}$ ТО

$$y_i = p_{io} + \sum_{j=1}^{N} p_{ij} \cdot x_j$$

Условие (X_i это A_i) реализуется функцией фазификации:

$$\mu_A(x_i) = \frac{1}{1 + \left(\frac{x_i - c_i}{\sigma_i}\right)^{2b_i}}$$

При M правилах агрегированный выходной результат сети имеет вид:

$$\begin{cases} y(x) = \sum_{i=1}^{M} u_i \cdot y_i(x) / \sum_{i=1}^{M} u_i \\ y_i = p_{io} + \sum_{j=1}^{N} p_{ij} \cdot x_j \end{cases} \quad (21)$$

Веса интерпретируются как значимость компонентов. Тогда формуле (21) можно поставить в соответствие многослойную нейронную сеть.

Первый слой выполняет фазификацию каждой переменной. Это параметрический слой с параметрами c^i_j , σ^i_j , b^i_j , подлежащими адаптации в процессе обучения.

Второй слой выполняет агрегирование отдельных переменных, определяя результирующее значение коэффициента принадлежности $u_i = \mu^i_{A(x)}$ для вектора x (непараметрический слой).

Третий слой - генератор функции TSK, рассчитывает значения:

$$y_i = p_{io} + \sum_{j=1}^{N} p_{ij} \cdot x_j$$

В этом слое также производится умножение $y_i(x)$ на u_i, сформированные в предыдущем слое. Здесь адаптации подлежат веса p_{ij}, i = 1,2,⋯ ,M; j = 1,2,⋯,N, определяющие функцию следствия модели TSK.

Четвертый слой составляют два нейрона-сумматора, один из которых рассчитывает взвешенную сумму сигналов $y_k(x)$, а второй - сумму весов u_i, где i = 1,2,...,M (непараметрический слой).

Пятый слой из одного нейрона - это нормализующий слой, в котором выходной сигнал сети агрегируется по формуле (21).

Таким образом, в процессе обучения происходит уточнение параметров только первого (нелинейного) и третьего (линейного) слоев.

Параметры, подлежащие адаптации, разделяются на две группы: первая состоит из параметров p_{ij} линейного третьего слоя; вторая состоит из параметров нелинейной функции принадлежности первого слоя.

Уточнение параметров проводится в два этапа. На первом этапе при фиксации определенных значений параметров функции принадлежности путем решения системы линейных уравнений рассчитываются параметры p_{ij} полинома TSK. При известных значениях функции принадлежности преобразование, реализуемое сетью, можно представить в виде:

$$\begin{cases} y(x) = \sum_{i=1}^{M} u_i \cdot (p_{io} + \sum_{i=1}^{N} p_{ij} \cdot x_j) \\ u_i = [\prod_{j=1}^{N} \mu_A^i(x_j)] / \sum_{k=1}^{N} [\prod_{j=1}^{N} \mu_A^k(x_j)] = const \end{cases} \quad (22)$$

При p обучающих выборках и замене выходного сигнала сети ожидаемым значением d(l) получим систему из p линейных уравнений вида:

$$W \cdot P = d, \quad (23)$$

где

$$W = \begin{Vmatrix} u_{11} & u_{11} \cdot x_1^1 & u_{11} \cdot x_N^1 & \dots & u_{1M} & u_{1M} \cdot x_1^1 & u_{1M} \cdot x_N^1 & \dots \\ u_{21} & u_{21} \cdot x_1^2 & u_{21} \cdot x_N^2 & \dots & u_{2M} & u_{2M} \cdot x_1^2 & u_{2M} \cdot x_N^2 & \dots \\ \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots \\ u_{p1} & u_{p1} \cdot x_1^p & u_{p1} \cdot x_N^p & \dots & u_{pM} & u_{pM} \cdot x_1^p & u_{pM} \cdot x_N^p & \dots \end{Vmatrix} \quad (24)$$

$$P = \| p_{10} \dots p_{1N} \dots p_{M0} \dots p_{MN} \|^T, \quad (25)$$

где u_{ki}– уровень активации (вес) i-го правила при предъявлении k-го входного вектора x(k).

Размерность матрицы W равна p × (N + 1) M, при этом обычно количество строк (количество выборок) значительно больше количества столбцов.

Решение этой системы уравнений можно получить за один шаг при помощи псевдоинверсии матрицы W.

Псевдоинверсия матрицы заключается в решении задачи минимизации

$$min||W + W - E||, \quad (26)$$

где E – единичная матрица.

На втором этапе (линейные параметры p_{ij}, i = 1,2...,M - фиксированы) рассчитываются фактические выходные сигналы y_k, k = 1,2,...,p:

$$y = W \cdot p, \tag{27}$$

вектор ошибки

$$\varepsilon = y - d, \tag{28}$$

и градиент целевой функции E(n) по параметрам первого слоя.

Если применяется метод наискорейшего спуска, то формулы адаптации принимают вид:

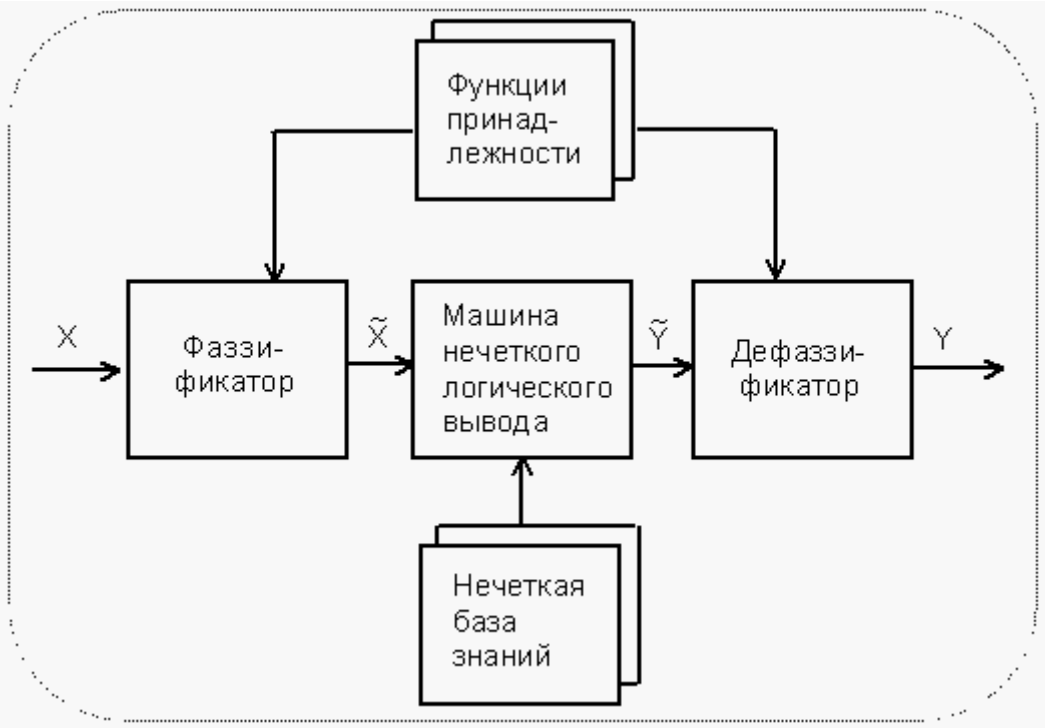
$$\begin{cases} c_j^i(n+1) = c_j^i(n) - \alpha_c \cdot \partial E(n)/\partial c_j^i \\ \sigma_j^i(n+1) = \sigma_j^i(n) - \alpha_\sigma \cdot \partial E(n)/\partial \sigma_j^i \\ b_j^i(n+1) = b_j^i(n) - \alpha_b \cdot \partial E(n)/\partial b_j^i \end{cases} \tag{29}$$

где n обозначает номер очередной итерации.

После уточнения нелинейных параметров вновь запускается процесс адаптации линейных параметров TSK (первый этап) и нелинейных параметров (второй этап). Этот цикл повторяется вплоть до стабилизации всех параметров процесса.

2.7 Пример построения нечеткой модели в Matlab

За реализацию моделей нечеткой логики в Matlab отвечает Fuzzy Logic Toolbox. Fuzzy Logic Toolbox – это пакет прикладных программ, который позволяет создавать системы нечеткого логического вывода и нечеткой классификации в рамках среды Matlab, с возможностью их интегрирования в Simulink. Базовым понятием Fuzzy Logic Toolbox является FIS-структура – система нечеткого вывода (Fuzzy Inference System). FIS-структура содержит все необходимые данные для реализации функционального отображения “входы-выходы” на основе нечеткого логического вывода согласно схеме, приведенной на рисунке 6.



Обозначения:
X - входной четкий вектор;
$\tilde{X}$ - вектор нечетких множеств, соответствующий входному вектору X;
$\tilde{Y}$ - результат логического вывода в виде вектора нечетких множеств;
Y - выходной четкий вектор.

Рисунок 6 – Схема процесса нечеткого моделирования в Matlab

Первая категория программных инструментов пакета Fuzzy Logic Toolbox содержит функции, которые могут быть вызваны непосредственно путем набора имени функции в командном окне («Command line») или из собственных пользовательских приложений. Большинство из этих функций представляют собой матлабовские функции в виде m-файлов. В этом случае пользователь может посмотреть запрограммированные в этих функциях алгоритмы, а также редактировать и корректировать эти файлы. Ниже приведены названия функций с кратким описанием их назначения:

addmf - добавление функции принадлежности в FIS;

addrule - добавление правила в FIS;

addvar - добавление переменной в FIS;

anfis - обучение FIS типа Сугэно (Sugeno type);

convertfis - преобразование FIS-матрицы (Fuzzy Logic Toolbox v.1) в FIS-структуру (Fuzzy Logic Toolbox v.2);

defuzz - дефаззификация нечеткого множества;

discfis - дискретизация функций принадлежности всех термов, входящих в FIS;

dsigmf - функция принадлежности в виде разности между двумя сигмоидными функциями;

evalfis - выполнение нечеткого логического вывода;

evalmf - вычисление значений произвольной функции принадлежности;

evalmmf - расчет степеней принадлежностей для нескольких функций принадлежностей;

fcm - поиск кластеров по алгоритму fuzzy c-means;

findrow - нахождение строки в матрице, совпадающей с входной строкой;

fstrvcat - конкатенация матриц различного размера;

fuzarith - нечеткий калькулятор;

gauss2mf - двухсторонняя гауссовская функция принадлежности;

gaussmf - гауссовская функция принадлежности;

gbellmf - обобщенная колокообразная функция принадлежности;

genfis1 - генерирование из данных исходной FIS типа Сугэно без использования кластеризации;

genfis2 - генерирование из данных исходной FIS типа Сугэно с использованием субтрактивной кластеризации;

genparam - генерирование исходных параметров функций принадлежности для обучения ANFIS (Adaptive-Network-based Fuzzy Inference System);

gensurf - генерирование поверхности “входы-выход”, соответствующей FIS;

getfis - получение свойств FIS;

mam2sug - преобразование FIS типа Мамдани в FIS типа Сугэно;

mf2mf - пересчет параметров встроенных функций принадлежности различных типов;

newfis - создание новой FIS;

parsrule - вставка в FIS правил, заданных в виде предложений на естественном языке;

pimf - пи-подобная функция принадлежности;

plotfis - вывод основных параметров FIS в виде графической схемы;

plotmf - вывод графиков функций принадлежности термов одной переменной;

probor - вероятностная реализация логической операции ИЛИ;

psigmf - произведение двух сигмоидных функций принадлежности;

readfis - загрузка FIS из файла;

rmmf - удаление функции принадлежности терма из FIS;

rmvar - удаление переменной из FIS;

setfis - назначение свойств FIS;

showfis - вывод на экран в текстовом формате данных, составляющих FIS-структуру;

showrule - вывод базы знаний FIS;

sigmf - сигмоидная функция принадлежности;

smf - s-подобная функция принадлежности;

subclust - оценка количества кластеров в субтрактивной кластеризации;

sugmax - нахождения диапазона изменения выходной переменной в FIS типа Сугэно;

trapmf - трапециевидная функция принадлежности;

trimf - треугольная функция принадлежности;

writefis - сохранение FIS на диске;

zmf - z-подобная функция принадлежности;

distfcm - расчет расстояния по Евклиду;

initfcm - генерирование исходной матрицы степеней принадлежности для нечеткой c-means кластеризации;

isfis - проверка структуры данных системы нечеткого логического вывода.

Вторая категория программных инструментов пакета Fuzzy Logic Toolbox содержит диалоговые модули, которые обеспечивают доступ к большинству функций через графический интерфейс. Кроме того, эти модули обеспечивают удобную среду для проектирования, исследования и внедрения систем на основе нечеткого логического вывода. Для запуска интерактивных модулей достаточно напечатать имя модуля в командной строке. Ниже приведены названия модулей с кратким описание их назначения:

anfisedit - модуль для генерирования из данных FIS типа Сугэно, ее обучения с использованием ANFIS алгоритма и тестирования;

findcluster - модуль кластеризации данных с использованием алгоритма fuzzy c-means и алгоритма нечеткой субтрактивной кластеризации;

fuzzy - основной редактор FIS. Позволяет создавать и редактировать FIS двух типов - Мамдани и Сугэно, обеспечивает визуализацию процедуры нечеткого логического вывода и поверхностей “входы-выход”. Для этого модуль fuzzy вызывает следующие GUI-модули: mfedit, ruleedit, ruleview и surfview;

mfedit - редактор функций принадлежностей. Позволяет выбирать тип функции принадлежности и устанавливать ее параметры в символьном и в интерактивном графическом (drag) режимах;

ruleedit - редактор базы знаний;

ruleview - модуль визуализации процедуры нечеткого логического вывода. Обеспечивает вывод графической диаграммы нечеткого вывода по каждому правилу, включая процедуры фаззификации, агрегации и дефаззификации. Позволяет вводить значения входных переменных в символьном и в интерактивном графическом (drag) режимах.

Третья категория программных инструментов пакета Fuzzy Logic Toolbox содержит следующие модули, которые обеспечивают интеграцию систем нечеткого логического вывода с пакетом Simulink:

fuzblock - модули контроллеров на основе нечеткого логического вывода;

sffis - функция выполнения нечеткого логического вывода, оптимизированная под Simulink.

Четвертая категория программных инструментов пакета Fuzzy Logic Toolbox содержит следующие демонстрационные примеры:

defuzzdm - дефаззификация различными методами;

fcmdemo - 2D-кластеризация с использование алгоритма fuzzy c-means;

fuzdemos - список всех демонстрационных примеров Fuzzy Logic Toolbox;

gasdemo - применение алгоритма ANFIS и субтрактивной кластеризации для идентификации зависимости топливной эффективности (расход топлива на одну милю) от шести параметров автомобиля;

invkine - инверсная кинематика робота-манипулятора;

irisfcm - применение алгоритма fuzzy c-means для кластеризации ирисов;

juggler - жонглирование шариком с помощью теннисной ракетки с демонстрацией нечеткой базы знаний;

noisedm - адаптивное подавление шумов;

slbb - управление системой «шарик на коромысле» (необходим пакет Simulink);

slcp - управление системой «перевернутый маятник» (необходим пакет Simulink);

sltank - управление уровнем воды (необходим пакет Simulink);

sltankrule - управление уровнем воды с демонстрацией нечеткой базы знаний (необходим пакет Simulink);

sltbu - парковка грузовика (необходим пакет Simulink);

mgtsdemo - предсказание временного ряда Маккея-Глэсса;

trips - построение модели прогнозирования количества автомобильных поездок;

shower - управление душем;

slcp1 - перемещение неустойчивой системы «перевернутый маятник переменной длины на тележке» в заданную точку;

slcpp1 - перемещение неустойчивой системы «два перевернутых маятника на тележке» в заданную точку;

slcpp1 - перемещение неустойчивой системы «два перевернутых маятника на тележке» в заданную точку;

mfdemo - вывод на экран окна, содержащего графики всех запрограммированных в Fuzzy Logic Toolbox типов функций принадлежностей;

drydemo - иллюстрация применения технологии ANFIS для идентификации нелинейных динамических систем на примере процесса нагрева воздуха в фене.

Модуль fuzzy позволяет строит нечеткие системы двух типов - Мамдани и Сугэно. В системах типа Мамдани база знаний состоит из правил вида «Если x_1=низкий и x_2=средний, то y=высокий». В системах типа Сугэно база знаний состоит из правил вида «Если x_1=низкий и x_2=средний, то $y=a_0+a_1x_1+a_2x_2$». Таким образом, основное отличие между системами Мамдани и Сугэно заключается в разных способах задания значений выходной переменной в правилах, образующих базу знаний. В системах типа Мамдани значения выходной переменной задаются нечеткими термами, в системах типа Сугэно - как линейная комбинация входных переменных.

Рассмотрим основные этапы проектирования систем типа Мамдани на примере создания системы нечеткого логического вывода, моделирующей зависимость $y = x_1^2 \cdot \sin(x_2 - 1), x_1 \in [-7; 3], x_2 \in [-4{,}4; 1{,}7]$. Проектирование системы нечеткого логического вывода будем проводить на основе графического изображения указанной зависимости.

Для построения трехмерного изображения функции в заданной области составим следующую программу:

```
%Построение графика функции y=x1^2*sin(x2-1)
%в области x1є[-7,3] и x2є[-4.4,1.7].
n=15;
x1=-7:10/(n-1):3;
x2=-4.4:6.1/(n-1):1.7;
y=zeros(n,n);
for j=1:n
y(j,:)=x1.^2*sin(x2(j)-1);
```

end

surf(x1,x2,y)

xlabel('x1')

ylabel('x2')

zlabel('y')

title('Target');

В результате выполнения программы получим графическое изображение, приведенное на рисунке 7. Проектирование системы нечеткого логического вывода, соответствующей приведенному графику, состоит в выполнении следующей последовательности шагов.

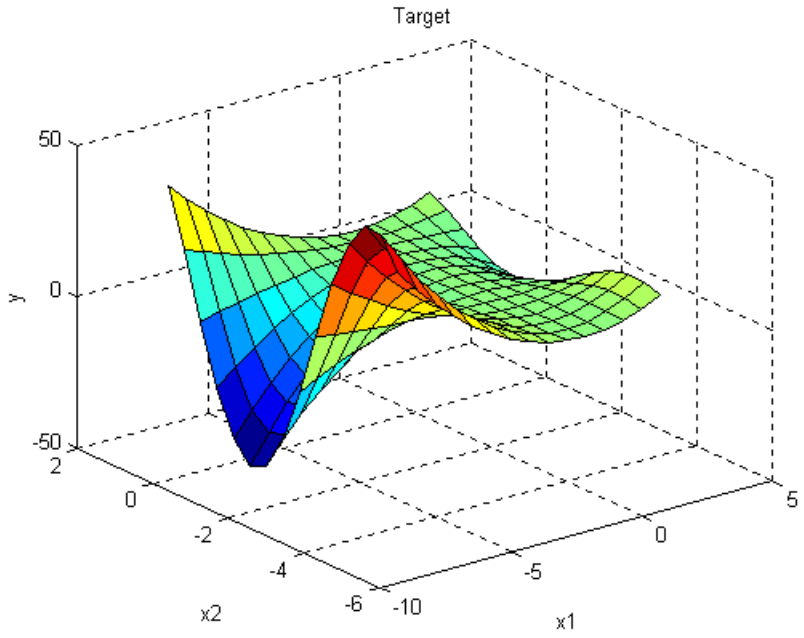


Рисунок 7 – Эталонная поверхность

Шаг 1. Для загрузки основного fis-редактора напечатаем слова «fuzzy» в командной строке. После этого откроется нового графическое окно, показанное на рисунке 8.

Шаг 2. Добавим вторую входную переменную. Для этого в меню «Edit» выбираем команду «Add input».

Шаг 3. Переименуем первую входную переменную. Для этого сделаем один щелчок левой кнопкой мыши на блоке «input1», введем новое обозначение «x1» в поле редактирования имени текущей переменной и нажмем <Enter>.

Шаг 4. Переименуем вторую входную переменную. Для этого сделаем один щелчок левой кнопкой мыши на блоке «input2», введем новое обозначение «x2» в поле редактирования имени текущей переменной и нажмем <Enter>.

Шаг 5. Переименуем выходную переменную. Для этого сделаем один щелчок левой кнопкой мыши на блоке «output1», введем новое обозначение «y» в поле редактирования имени текущей переменной и нажмем <Enter>.

Шаг 6. Зададим имя системы. Для этого в меню «File» выбираем в подменю «Export» команду «To disk» и вводим имя файла, например, «first».

Шаг 7. Перейдем в редактор функций принадлежности. Для этого сделаем двойной щелчок левой кнопкой мыши на блоке «x1».

Шаг 8. Зададим диапазон изменения переменной «x1». Для этого напечатаем «-7 3» в поле «Range» и нажмем <Enter>.

Шаг 9. Зададим функции принадлежности переменной «x1». Для лингвистической оценки этой переменной будем использовать 3 терма с треугольными функциями принадлежности. Для этого в меню «Edit» выберем команду «Add MFs». В результате появиться диалоговое окно выбора типа и количества функций принадлежностей. По умолчанию это 3 терма с треугольными функциями принадлежности. Поэтому просто нажимаем <Enter>.

Шаг 10. Зададим наименования термов переменной «x1». Для этого делаем один щелчок левой кнопкой мыши по графику первой функции принадлежности. Затем вводим наименование терма, например, «Низкий», в поле «Name» и нажмем <Enter>. Затем делаем один щелчок левой кнопкой мыши по графику второй функции принадлежности и вводим наименование терма, например, «Средний», в поле «Name» и нажмем <Enter>. Еще раз делаем один щелчок левой кнопкой мыши по графику

третьей функции принадлежности и вводим наименование терма, например, «Высокий», в поле «Name» и нажмем <Enter>. В результате получим графическое окно, изображенное на рисунке 9.

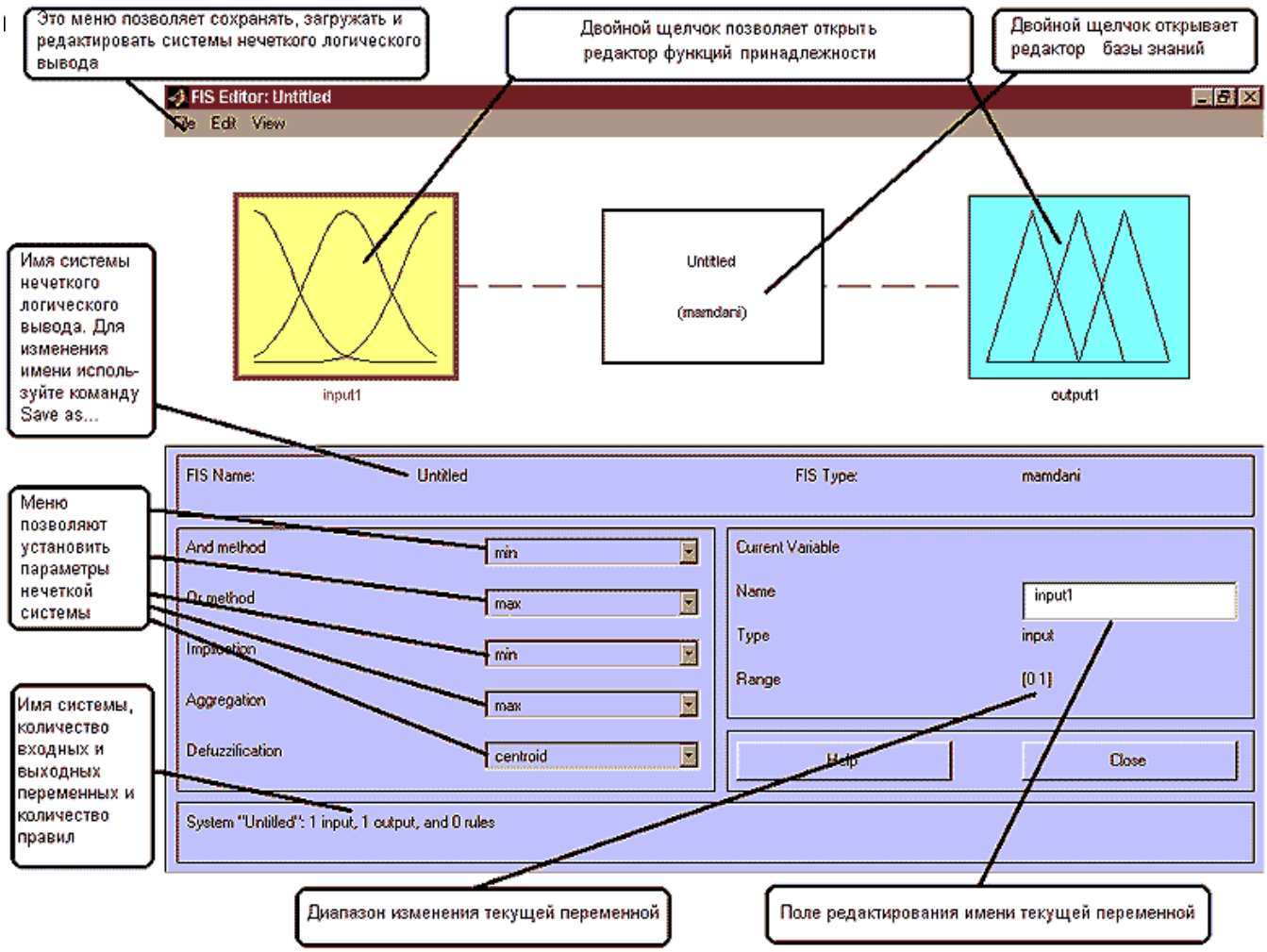


Рисунок 8 – Общий вид окна редактора нечеткой логики

Шаг 11. Зададим функции принадлежности переменной «x2». Для лингвистической оценки этой переменной будем использовать 5 термов с гауссовскими функциями принадлежности. Для этого активизируем переменную «x2» с помощью щелчка левой кнопки мыши на блоке «x2». Зададим диапазон изменения переменной «x2». Для этого напечатаем «-4.4 1.7» в поле «Range» и нажмем <Enter>. Затем в меню «Edit» выберем команду «Add MFs». В появившимся диалоговом окне выбираем тип

функции принадлежности «gaussmf» в поле «MF type» и 5 термов в поле «Number of MFs». После этого нажимаем <Enter>.

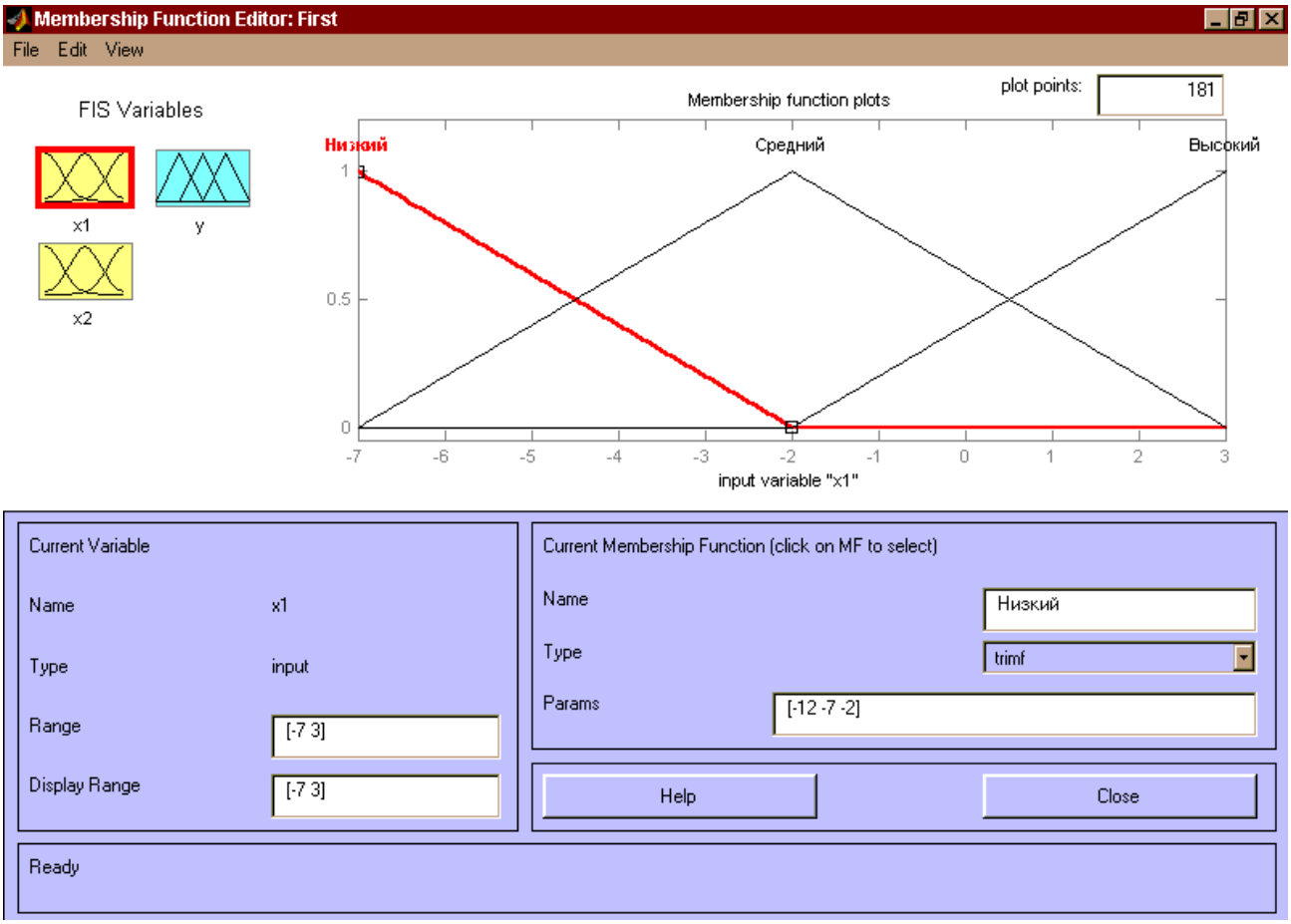


Рисунок 9 – Функции принадлежности переменной «x1»

Шаг 12. По аналогии с шагом 10 зададим следующие наименования термов переменной «x2»: «Низкий», «Ниже среднего», «Средний», «Выше среднего», «Высокий». В результате получим графическое окно, изображенное на рисунке 10.

Шаг 13. Зададим функции принадлежности переменной «y». Для лингвистической оценки этой переменной будем использовать 5 термов с треугольными функциями принадлежности. Для этого активизируем переменную «y» с помощью щелчка левой кнопки мыши на блоке «y». Зададим диапазон изменения переменной «y». Для этого напечатаем «-50 50» в поле «Range» и нажмем <Enter>. Затем в меню «Edit» выберем команду «Add MFs». В появившимся диалоговом окне выбираем 5 термов в поле «Number of MFs». После этого нажимаем <Enter>.

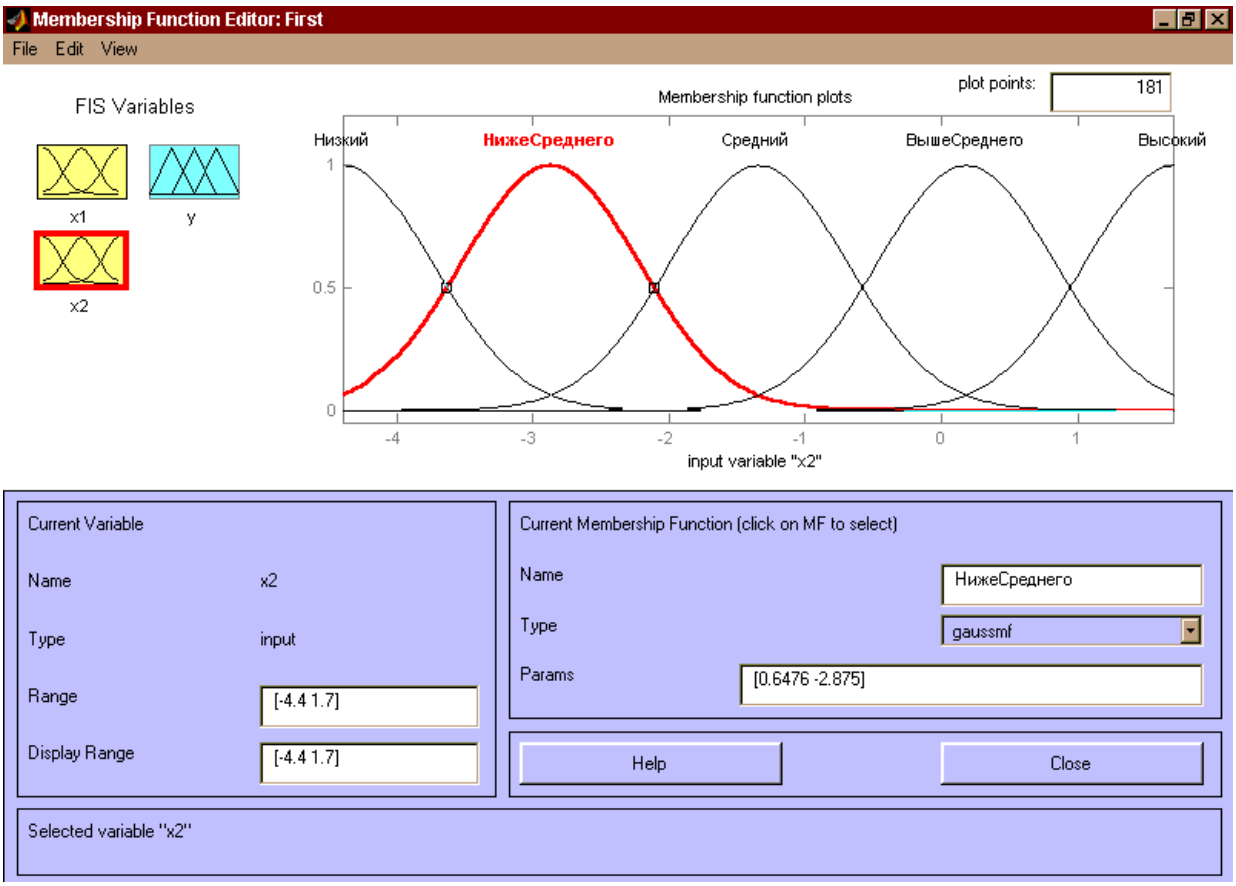


Рисунок 10 – Функции принадлежности переменной «x2»

Шаг 14. По аналогии с шагом 10 зададим следующие наименования термов переменной «y»: «Низкий», «Ниже среднего», «Средний», «Выше среднего», «Высокий». В результате получим графическое окно, изображенное на рисунке 11.

Шаг 15. Перейдем в редактор базы знаний «RuleEditor». Для этого выберем в меню «Edit» выберем команду «Edit rules».

Шаг 16. На основе визуального наблюдения за графиком, изображенным на рисунке 7 сформулируем следующие девять правил:

Если x1=Средний, то y=Средний;

Если x1=Низкий и x2=Низкий, то y=Высокий;

Если x1=Низкий и x2=Высокий, то y=Высокий;

Если x1=Высокий и x2=Высокий, то y=Выше Среднего;

Если x1=Высокий и x2=Низкий, то y=Выше Среднего;

Если x1=Высокий и x2=Средний, то y=Средний;

Если x1=Низкий и x2=Средний, то y=Низкий;

Если x1=Высокий и x2=Выше Среднего, то y=Средний;

Если x1=Высокий и x2=Ниже Среднего, то y=Средний.

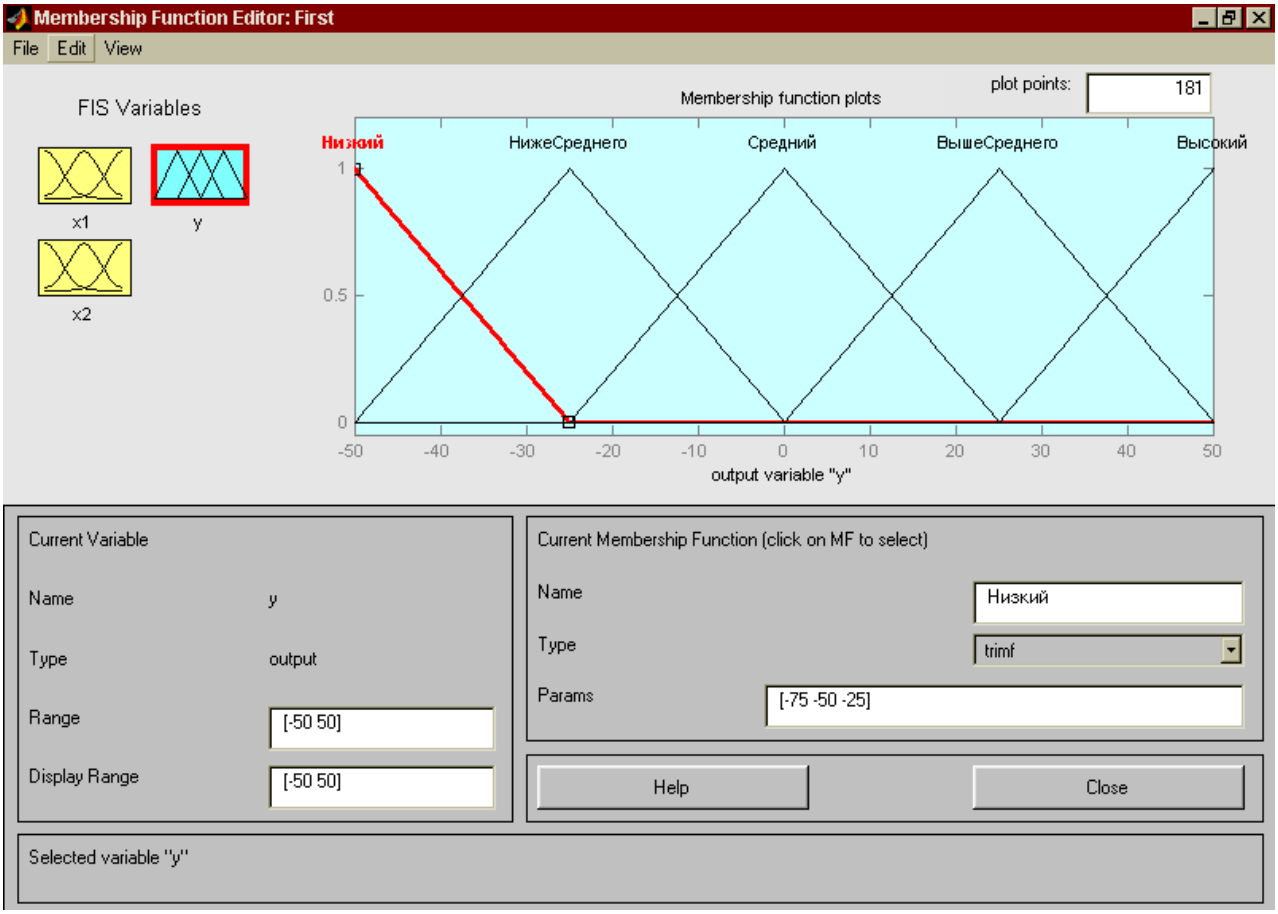


Рисунок 11 – Функции принадлежности переменной «y»

Для ввода правила необходимо выбрать в меню соответствующую комбинацию термов и нажать кнопку «Add rule». На рисунке 12 изображено окно редактора базы знаний после ввода всех девяти правил. Число, приведенное в скобках в конце каждого правила, представляет собой весовым коэффициент соответствующего правила.

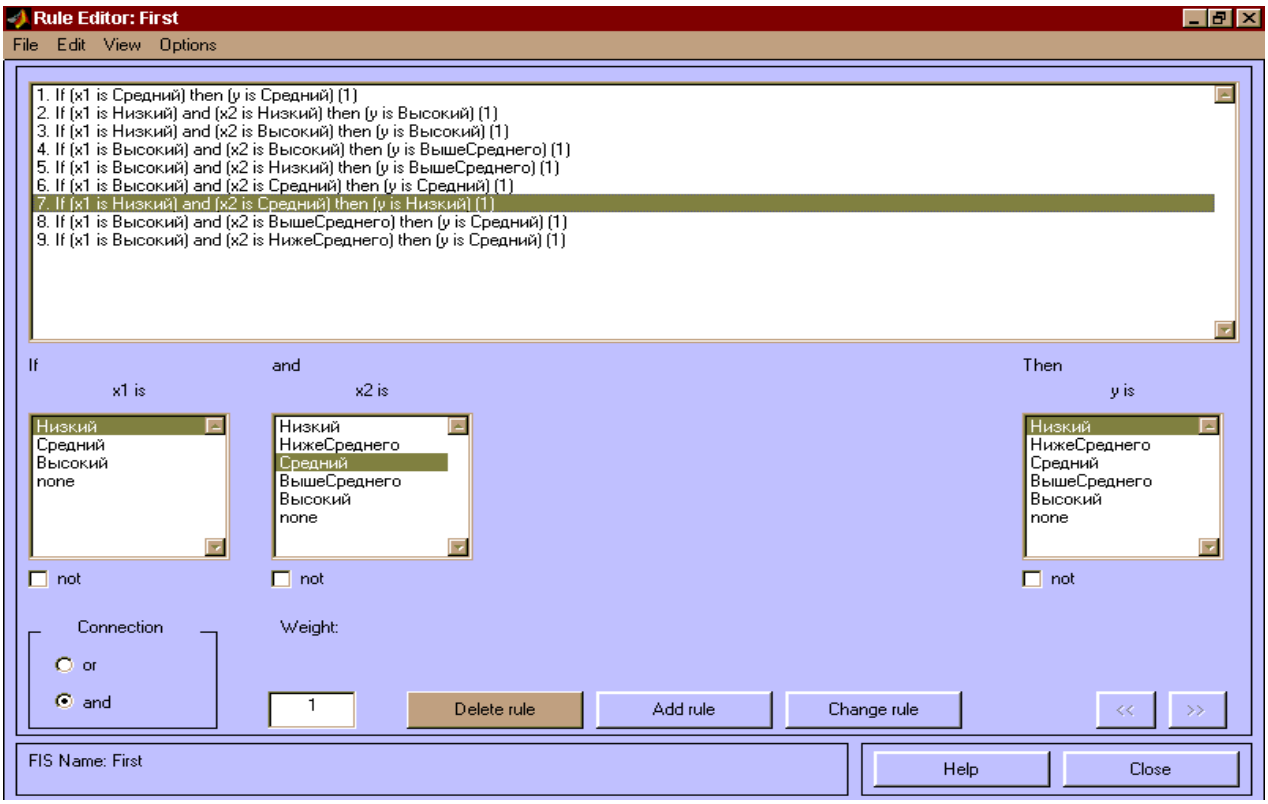


Рисунок 12 – Редактор базы правил

Шаг 17. Сохраним созданную систему. Для этого в меню «File» выбираем в подменю «Export» команду «To disk».

На рисунке 13 приведено окно визуализации нечеткого логического вывода. Это окно активизируется командой «View rules», меню «View». В поле «Input» указываются значения входных переменных, для которых выполняется логический вывод.

На рисунке 14 приведена поверхность «входы-выход», соответствующая синтезированной нечеткой системе. Для вывода этого окна необходимо использовать команду «View surface» из меню «View». Сравнивая поверхности на рисунке 7 и на рисунке 14 можно сделать вывод, что нечеткие правила достаточно хорошо описывают сложную нелинейную зависимость.

Отметим, что, настраивая функции принадлежности исследователь часто полагается на свою интуицию, что требует наличие опыта работы с системами нечеткого ввода. Более простой в настройке и использовании является модель Сугено.

Рассмотрим ее на примере данных таблицы 3. Создадим переменную «CP» и в Matlab (см. раздел **Ошибка! Источник ссылки не найден.**), заполнив первый столбец значениями концентраций.

Второй столбец области данных переменной заполним значениями плотности раствора (рисунок 15).

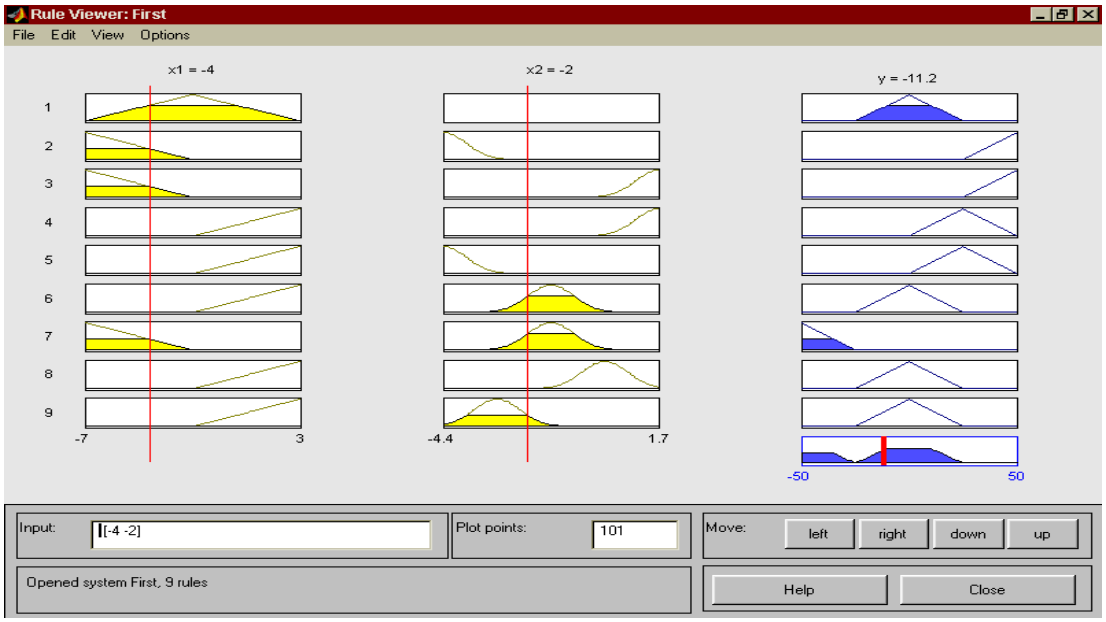


Рисунок 13 – визуализация нечеткого логического вывода правил

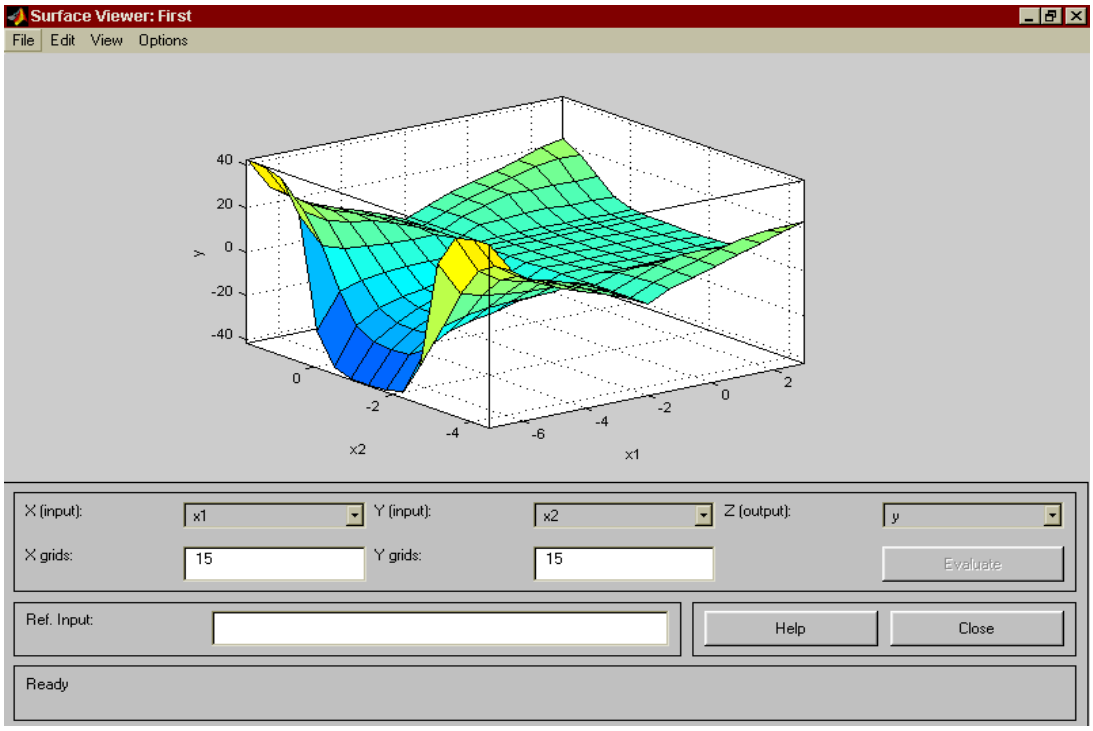


Рисунок 14 – Поверхность «входы-выход» в окне «SurfaceViwer»

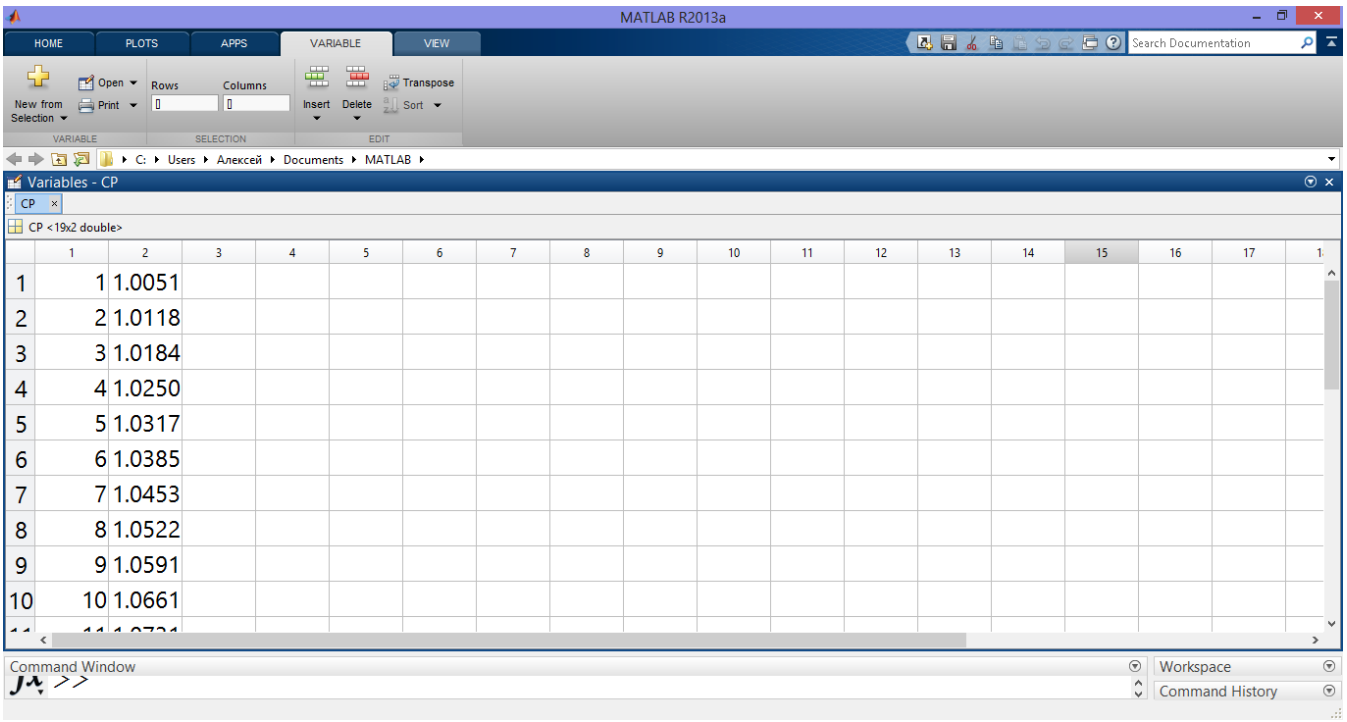


Рисунок 15 – Область данных переменной «CP»

Для загрузки основного fis-редактора напечатаем слова «fuzzy» в командной строке. Алгоритмом по умолчанию является алгоритм Мамдани. Сменим его на алгоритм Сугено, для этого выполним следующую последовательность действий: «File» – «New FIS» – «Sugeno».

Поскольку в нашем случае должна быть функция одного аргумента, то в структуре модели ничего менять не нужно. В противном случае, необходимо привести число входных воздействий и выходных результатов в соответствие с реальным набором данных.

Следующим шагом настроим функции принадлежности, используя алгоритм Сугено для их обучения. Для этого вызовем в меню «Edite» пункт «Anfis». Получим окно «Anfis Editor».

Выберем в блоке «Load data» в поле «Type:» – «Training», в поле «From:» – «worksp.» и нажмем кнопку «Load Data». В появившемся окне запроса введем название переменной «CP», хранящей исходные данные «вход-выход» и нажмем «Ok».

Далее, в области блоки «Generate FIS», отметим способ «Grid partition» и нажмем «Generate FIS». В появившемся окне запроса о структуре функций принадлежности оставим параметры по умолчанию (три функции принадлежности треугольной формы) и нажмем «Ok».

Перейдем к блоку «Train FIS», оставив его параметры по умолчанию и нажмем кнопку «Train Now».

После, в блоке «Test Fis» ставим «Training data» и нажимаем «Test Now», убеждаемся, что функции принадлежности настроены так. что система достаточно хорошо описывает точки экспериментальной выборки (рисунок 16).

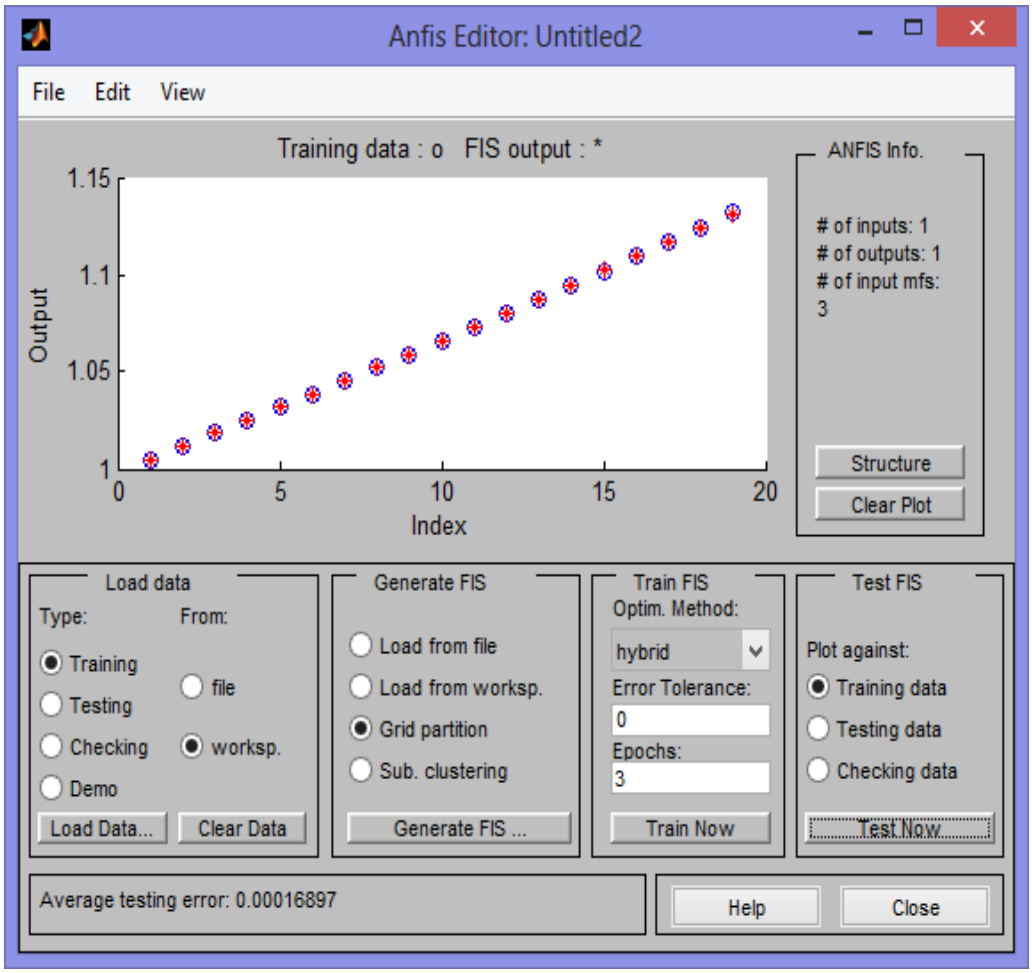


Рисунок 16 – Тестирование и настройка нечеткой модели

На рисунке 17 приведено окно визуализации нечеткого логического вывода. Это окно активизируется командой «View rules», меню «View». В поле «Input» указываются значения входных переменных, для которых выполняется логический вывод.

На рисунке 18 приведена поверхность «входы-выход», соответствующая синтезированной нечеткой системе.

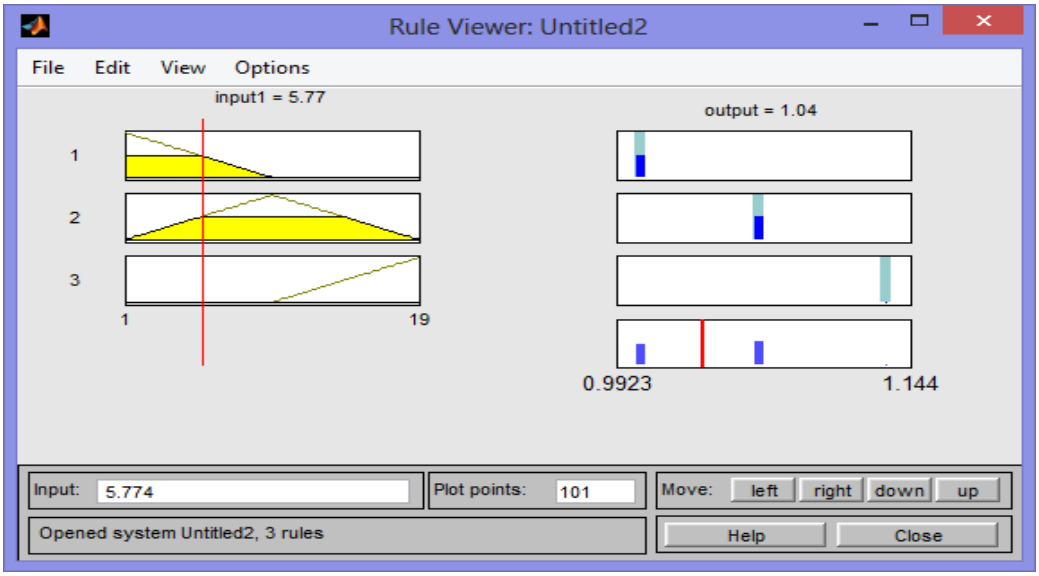


Рисунок 17 – Визуализация работы нечеткой модели

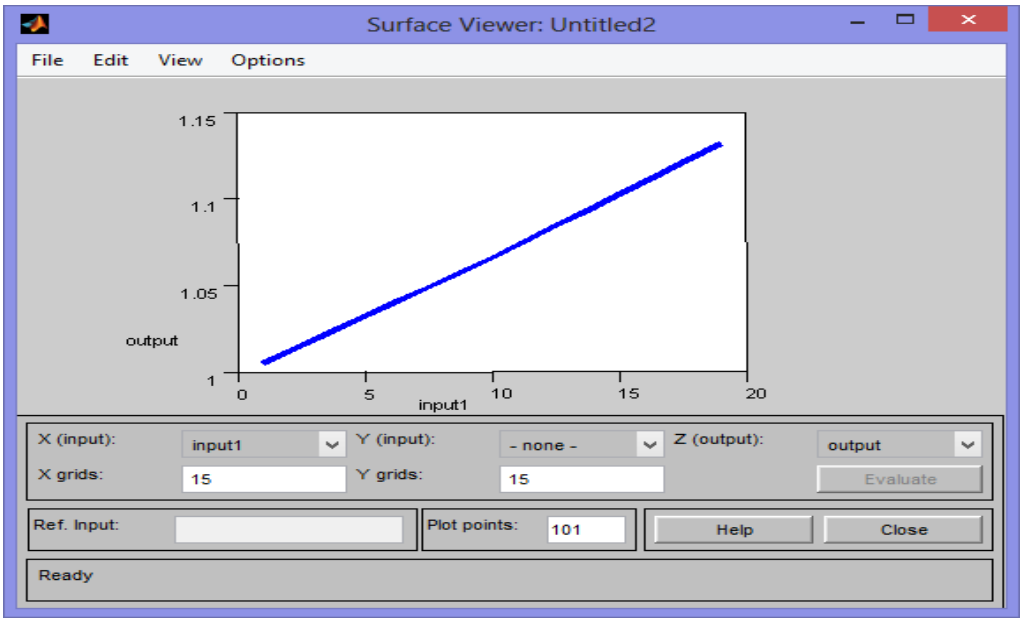


Рисунок 18 – Поверхность «входы-выход» в окне «SurfaceViwer» модели Сугено

Для вывода этого окна необходимо использовать команду «View surface» из меню «View». Таким образом можно сделать вывод, что нечеткие модели достаточно хорошо описывают как сложную нелинейную зависимость, так и достаточно простую и могут быть использованы для задач прогнозирования и аппроксимации.

2.8 Использование нейронных сетей для аппроксимации эмпирических данных функций

В ряде методов, созданных для реализации искусственного интеллекта, используются явные представления знаний и тщательно спроектированные алгоритмы перебора. Отличный от этого подход состоит в построении интеллектуальных программ с использованием моделей, имитирующих нейронные структуры в человеческом мозге или эволюцию разных альтернативных конфигураций, как это делается в генетических алгоритмах и искусственной жизни. Для того чтобы решать сложные и плохо формализуемые задачи и возникло направление, которое называется искусственные нейронные сети.

Искусственные нейронные сети состоят из нейроноподобных элементов, соединенных между собой в сеть. Существуют статические и динамические нейронные сети. В статических нейронных сетях изменение параметров системы происходит по некоторому алгоритму в процессе обучения. После обучения параметры сети не меняются. В динамических нейронных сетях отображение внешней информации, и ее обработка осуществляется в виде некоторого динамического процесса, то есть процесса, зависящего от времени.

Нейронные сети нашли применение практически во всех областях науки и техники. С использованием нейронных сетей успешно решаются многие проблемы бизнеса и финансов. Задачи управления, классификации, распознавания образов, прогнозирования, присущие практически всем прикладным областям, таким как медицина, военное дело, авиация и космос, строительство, все чаще решаются с применением нейросетевых технологий.

В настоящее время одним из самых эффективных и обоснованных методов облучения нейронных сетей является алгоритм обратного распространения ошибки, который применим к однонаправленным многослойным сетям. В многослойных нейронных сетях имеется множество скрытых нейронов, входы и выходы которых не являются входами и выходами нейронной сети, а соединяют нейроны внутри сети, то есть скрытые нейроны. Занумеруем выходы нейронной сети индексом j=1..n, а обучающие примеры индексом M=1..M_0. Тогда в качестве целевой функции можно выбрать функцию ошибки как сумму квадратов расстояний между реальными выходными состояниями y_{jM} нейронной сети, выдаваемых сетью на входных данных примеров, и правильными значениями функции d_{jM}, соответствующими этим примерам. Пусть $x = \{x_i\}$ – столбец входных значений, где i=1,2,..,n. Тогда $y = \{y_j\}$ – выходные значения, где j=1,2,…,m. В общем случае n≠m. Рассмотрим разность $y_{jM} - d_{jM}$, где d_{ji} – точное (правильное) значение из примера. Эта разность должна быть минимальна. Введем расстояния согласно евклидовой метрике, определив норму:

$$\|y - d\| = \sqrt{(y - d, y - d)^2} \tag{30}$$

Пусть целевая функция имеет вид:

$$E = \frac{1}{2}\sum_{j,M}(y_{j,M} - d_{j,M})^2 \tag{31}$$

Коэффициент ½ выбран из соображений более короткой записи последующих формул. Задача обучения нейронной сети состоит в том, чтобы найти такие коэффициенты $w_{\beta k}$, при которых достигается минимум $E(w)$ $(E \geq 0)$.

На рисунке 19 показана архитектура нейронной сети с прямой передачей сигнала.

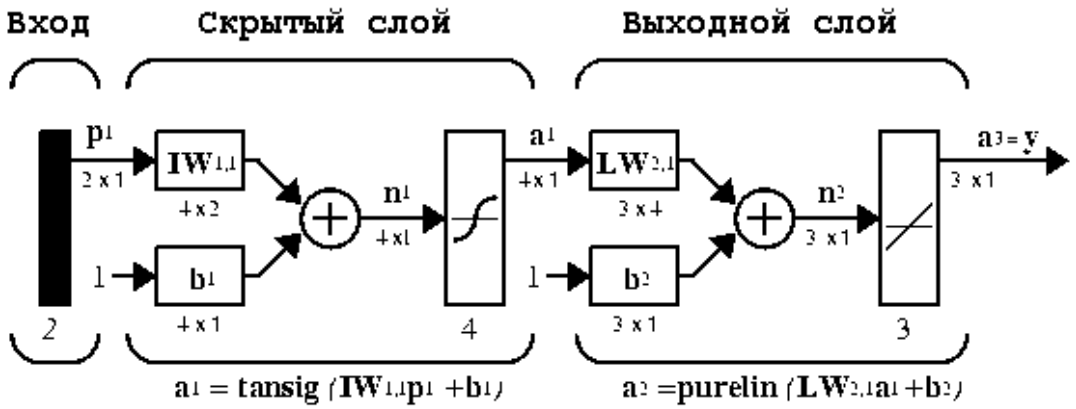


Рисунок 19 – Схема архитектуры нейронной сети с прямой передачей сигнала

Обучение сети обратного распространения требует выполнения следующих операций:

1) выбрать очередную обучающую пару из обучающего множества; подать входной вектор на вход сети;

2) вычислить выход сети;

3) вычислить разность между выходом сети и требуемым выходом (целевым вектором обучающей пары);

4) скорректировать веса сети так, чтобы минимизировать ошибку;

5) повторять шаги с 1 по 4 для каждого вектора обучающего множества до тех пор, пока ошибка на всем множестве не достигнет приемлемого уровня.

2.9 Пример апроксимации функции с помощью нейронной сети в Matlab

Рассмотрим случай, когда имеются данные приборов регистрации для кагого-либо производственного процесса или параметра процесса. Пусть имеются данные об изменении одного параметра в зависимости от другого (рисунок 20).

В примере, по графику были получены два массива, каждый из которых состоит из 15 значений. По горизонтальной оси – [0,10 0,31 0,51 0,72 0,93 1,14 1,34 1,55 1,76 1,96 2,17 2,38 2,59 2,79 3,00]. По вертикальной оси – [0,1010 0,3365 0,6551 1,1159 1,7632 2,5847 3,4686 4,2115 4,6152 4,6095 4,2887 3,8349 3,4160 3,1388 3,0603].

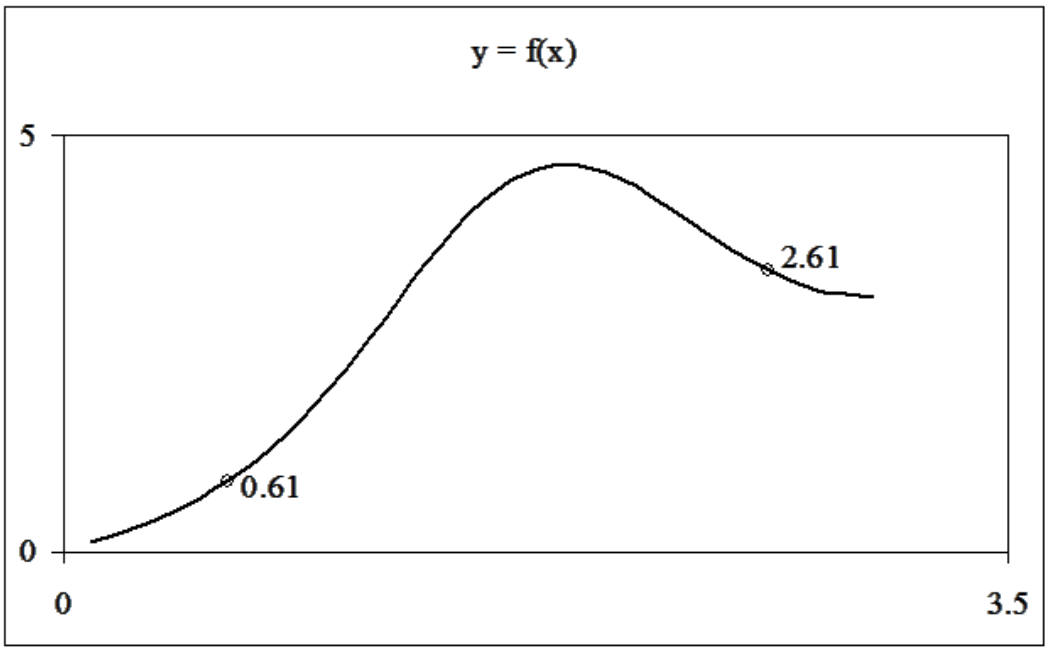


Рисунок 20 – Пример графической зависимости функции одной переменной

Реализуем апроксимацию в Matlab. Для этого создадим m-файл следующего содержания:

```
x=[0.10  0.31  0.51  0.72  0.93  1.14 ...
   1.34  1.55  1.76  1.96  2.17  2.38 ...
   2.59  2.79  3.00];
y=[0.1010 0.3365 0.6551 1.1159 1.7632 2.5847 ...
   3.4686 4.2115 4.6152 4.6095 4.2887 3.8349 ...
   3.4160 3.1388 3.0603];
net=newff([0 3],[5,1],{'tansig','purelin'},'trainbfg');
net.trainParam.epochs=300;
net.trainParam.show=50;
net.trainParam.goal=1,37e-2;
[net,tr]=train(net,x,y);
an=sim(net,x);
plot(x,y,'+r',x,an,'-g'); hold on;
xx=[0.61  2.61];
v=sim(net,xx)
```

plot(xx,v,'ob','MarkerSize',5,'LineWidth',2)

В результате выполнения программы получаются следующие результаты, отражённые на рисунке 21:

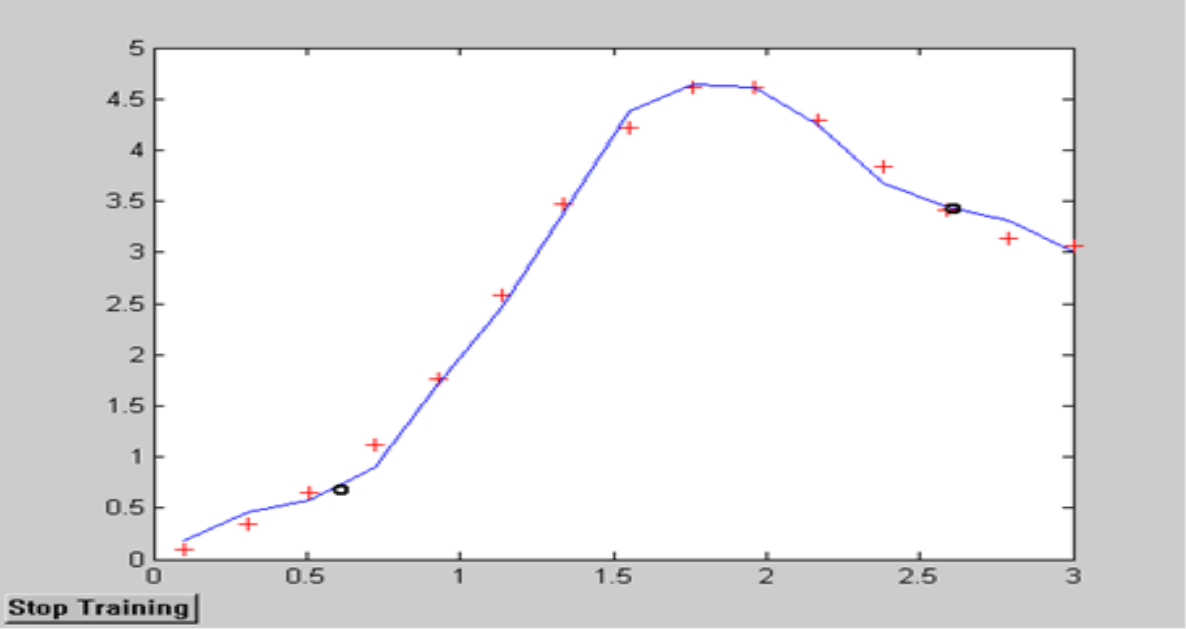


Рисунок 21 – Результаты моделирования сети: «+» - исходные данные; сплошная линия и символ «o» – результаты моделирования всей зависимости и в контрольных точках

Контрольные вопросы и задачи к разделу 2

1) Используя любые удобные для Вас средства найдите уравнение регрессии, связывающее два массива данных

X: {0; 5; 10; 15; 25}

У: {1;3,6;5,9;8,7;13,2}

2) Как оценивается точность линейной однопараметрической модели?

3) Как оценивается точность двухпараметрических линейных моделей?

4) Используя любые удобные для Вас средства найдите уравнение регрессии, связывающее два массива данных

X	1,0051	1,0385	1,0591	1,102	1,1318
У	0,103	0,635	0,972	1,685	2,192

5) Используя алгоритмы нейронных сетей или нечеткой логики постройте модель технологического процесса по следующим исходным данным таблицы 4.

Таблица 4 – Набор исходных параметров

№эксп	x_1	x_2	x_3	x_4	x_5	y	№эксп	x_1	x_2	x_3	x_4	x_5	y
1	234,2	34,20	0,23	20,0	0,5	12,32	11	269,87	69,21	0,41	20,4	5,5	17,21
2	239,6	36,13	0,24	20,0	1,0	13,43	12	270,87	69,25	0,43	20,4	6,0	16,43
3	241,3	39,20	0,25	20,0	1,5	14,45	13	268,10	68,32	0,35	20,3	6,5	17,50
4	252,5	41,98	0,26	20,0	2,0	15,65	14	272,98	71,06	0,51	20,5	7,0	14,23
5	257,9	45,67	0,27	20,3	2,5	16,87	15	281,21	73,76	0,53	20,4	7,5	11,65
6	266,3	49,12	0,28	20,3	3,0	17,45	16	289,54	75,87	0,57	20,4	8,0	11,24
7	267,2	52,98	0,29	20,3	3,5	17,56	17	290,12	79,21	0,61	20,5	8,5	10,67
8	267,5	67,87	0,30	20,4	4,0	18,32	18	292,54	80,10	0,54	20,5	9,0	9,80
9	267,9	68,54	0,32	20,4	4,5	17,34	19	294,98	82,05	0,52	20,5	9,5	9,03
10	268,1	68,32	0,35	20,4	5,0	17,32	20	300,02	85,98	0,53	20,5	10,0	8,54

6) Что можно сказать об исходных данных предыдущего задания, если известно, что при проведении параллельных опытов с объектом, его выход менялся:

y ={17,30 17,50 17,40 17,60 17,25}?

7) С какой целью используют нейросетевые алгоритмы? Как устроена нейронная сеть?

8) Перечислите основные этапы нечеткого вывода.

9) Чем отличаются и что общего в алгоритмах Мамдани и Сугэно?

10) Можно ли нейронную сеть, обученную по одному объекту, без изменения и переобучения использовать на всех объектах такого же типа и технических параметров?

11) Определите область применения регрессионных моделей и эмпирических моделей в целом.

12) Используя Matlab постройте аппроксимирующую кривую с использованием нейронной сети для функции

а) $y=2x+3x^3-4x^5+5\sin(20x)$ б) $y=8x^3+20$ в) $y=3(1-\exp(-x/3))$

3 Динамические модели химико-технологических процессов и оборудования

Обычно модели строятся на основе двух типов исходных данных:

– данные, характеризующие совокупность различных объектов в определенный момент (период) времени;

– данные, характеризующие один объект за ряд последовательных моментов (периодов) времени.

Модели, построенные по данным первого типа, называются пространственными (статическими) моделями. Они были разобраны в предыдущих разделах. Модели, построенные на основе второго типа данных, называются моделями временных рядов или динамическими. Временной ряд – совокупность значений какого-либо показателя за несколько последовательных моментов или периодов времени. Каждый уровень временного ряда формируется под воздействием большого числа факторов, которые условно можно подразделить на три группы:

1) факторы, формирующие тенденцию ряда;

2) факторы, формирующие циклические колебания ряда;

3) случайные факторы.

Очевидно, что реальные данные чаще всего содержат все три компоненты. Модель, в которой временной ряд представлен как сумма перечисленных компонент, называется аддитивной моделью временного ряда. Если же временной ряд представлен как их произведение, то такая модель называется мультипликативной. При наличии в временном ряде тенденции и циклических колебаний значения каждого последующего уровня ряда зависят от предыдущих. Корреляционную зависимость между последовательными уровнями временного ряда называют автокорреляцией уровней ряда. Количественно эту зависимость с помощью коэффициента корреляции между уровнями исходного временного ряда и уровнями этого ряда, сдвинутого на несколько шагов во времени. Этот показатель называется коэффициентом автокорреля-

ции r_k. Число периодов k, по которым происходит смещение временного ряда для вычисления коэффициента автокорреляции, называется лагом. Функция, характеризующая зависимость коэффициента автокорреляции от лага $r_k = r(k)$ называется автокорреляционной функцией, а ее график – коррелограммой. Коррелограмма позволяет исследовать структуру временного ряда, выявлять наличие его компонент. Коэффициенты автокорреляции со смещением (лагом) на k периодов находятся по формуле:

$$r_k = \frac{\overline{y_t \cdot y_{t+k}} - \overline{y_t} \cdot \overline{y_{t+k}}}{\sqrt{(\overline{y_t^2} - \overline{y_t}^2) \cdot (\overline{y_{t+k}^2} - \overline{y_{t+k}}^2)}} \quad (32)$$

Значимость каждого в отдельности коэффициента автокорреляции можно проверять с помощью критерия стандартной ошибки. Коэффициент автокорреляции можно считать значимым на уровне значимости $\alpha = 0{,}05$, если не выполняется неравенство:

$$-1{,}96 \cdot \frac{1}{\sqrt{n}} \leq r_k \leq 1{,}96 \cdot \frac{1}{\sqrt{n}}, \quad (33)$$

где n – число пар наблюдений временного ряда, используемые для расчета коэффициента;

k – лаг (смещение данных ряда).

Если рассчитанное значение автокорреляции попадает в этот интервал, то можно сделать вывод, что данные не показывают наличие автокорреляции k -го порядка. Знание автокорреляционной функции может оказать помощь при подборе и идентификации модели анализируемого временного ряда и статистической оценки его параметров. По коэффициенту автокорреляции можно судить о наличии линейной или близкой к линейной тенденции. Для некоторых временных рядов, имеющих сильную нелинейную тенденцию, например, параболу второго порядка или экспоненту, коэффициент автокорреляции уровней исходного ряда может приближаться к

нулю. Если наиболее высоким оказался коэффициент автокорреляции первого порядка, исследуемый ряд содержит только тенденцию. Если наиболее высоким оказался коэффициент автокорреляции порядка k, исследуемый ряд содержит циклические (сезонные) колебания с периодичностью в k моментов времени. Если ни один из коэффициентов автокорреляции не является значимым, можно сделать предположение относительно структуры этого ряда: либо ряд не содержит тенденции и циклических колебаний, либо ряд содержит сильную нелинейную тенденцию, для выявления которой нужно провести дополнительный анализ.

3.1 Передаточная функция. Преобразование Лапласа в решении дифференциальных уравнений

Реальные объекты не могут мгновенно изменять свое состояние, поэтому вместо статических моделей для их исследования используют динамические модели, которые описываются дифференциальными уравнениями, содержащими производные (скорости изменения сигналов). Такие модели могут быть получены из физических законов.

3.1.1 Понятие передаточной функции

При составлении уравнений динамики исследуемого элемента системы прежде всего необходимо выявить физический закон, определяющий его поведение во времени. Обычно это закон сохранения (вещества или энергии), который устанавливает невозможность мгновенного перехода физического объекта из одного состояния в другое, поскольку для этого потребовалась бы бесконечно большая скорость изменения состояния, что физически нереализуемо.

В динамическом моделировании часто используют модели вида «Вход-выход», в которых выходной сигнал можно представить, как результат действия некоторого

оператора на вход модели. Для линейных моделей такой оператор можно записать следующим образом.

Пусть модель объекта задана линейным дифференциальным уравнением второго порядка, связывающим вход x(t) и выход y(t):

$$b_2 \cdot \frac{d^2y(t)}{dt^2} + b_1 \cdot \frac{dy(t)}{dt} + b_0 \cdot y(t) = a_1 \cdot \frac{dx(t)}{dt} + a_0 \cdot x(t), \quad (34)$$

где a_i и b_i – некоторые постоянные коэффициенты уравнения модели.

Введем оператор дифференцирования p = d/dt, который действует на сигнал x(t) по правилу: $px(t) = dx(t)/dt$. Обратите внимание, что запись $px(t)$ обозначает не умножение оператора p на x(t), а действие этого оператора, то есть дифференцирование x(t). Тогда производная второго порядка по переменной y(t) запишется как $p^2y(t)$. Запишем уравнение (34) в принятой нами операторной форме:

$$b_2 \cdot p^2y(t) + b_1 \cdot py(t) + b_0 \cdot y(t) = a_1 \cdot px(t) + a_0 \cdot x(t) \quad (35)$$

Абстрагировавшись от смысла оператора p, выделим за скобки y(t) в левой части уравнения и x(t) в правой части уравнения. После чего приведем уравнение к виду $y = f(x)$.

$$y(t) \cdot (b_2 \cdot p^2 + b_1 \cdot p + b_0) = (a_1 \cdot p + a_0) \cdot x(t), \quad (36)$$

$$y(t) = \frac{(a_1 \cdot p + a_0)}{(b_2 \cdot p^2 + b_1 \cdot p + b_0)} \cdot x(t) \quad (37)$$

Обозначим

$$W(p) = \frac{(a_1 \cdot p + a_0)}{(b_2 \cdot p^2 + b_1 \cdot p + b_0)}, \quad (38)$$

получим запись уравнения в виде

$$y(t) = W(p) \cdot x(t), \quad (39)$$

где запись $W(p) \cdot x(t)$ означает не умножение, а действие сложного оператора W(p) на сигнал возмущения x(t).

Таким образом уравнение (39) представляет собой удобную символическую запись исходного уравнения второго порядка. Оператор W(p) называется передаточной функцией объекта. Она полностью описывает связи между выходом и входом объекта при нулевых начальных условиях, но не учитывает его внутреннее устройство.

3.1.2 Преобразование Лапласа

Одна из основных задач, которые возникают при математическом моделировании объектов – вычисление выхода системы при известном входе. Для ее реализации нужно решать дифференциальные уравнения, часто достаточно высоких порядков. Чтобы упростить процедуру, математики придумали преобразование, которое позволило заменить решение дифференциальных уравнений алгебраическими вычислениями, то есть, операциями с полиномами (многочленами) и рациональными функциями.

Для функции f(t) вводится преобразование Лапласа, которое обозначается как L{f(t)}:

$$F(s) = L\{f(t)\} = \int_0^\infty f(t) \cdot e^{-s \cdot t} dt. \quad (40)$$

Функция F(s) называется изображением для функции f (t) (оригинала). Здесь s – это комплексная переменная, которая выбирается так, чтобы интеграл (40) сходился.

Обратное преобразование Лапласа $L^{-1}\{F(s)\}$ позволяет вычислить оригинал f(t) по известному изображению F(s):

$$f(t) = L^{-1}\{F(s)\} = \frac{1}{(2\cdot\pi\cdot j)} \cdot \int_{\sigma-j\cdot\infty}^{\sigma-j\cdot\infty} F(s) \cdot e^{s\cdot t} ds, \quad (41)$$

где $j = \sqrt{-1}$, а постоянная σ выбирается так, чтобы интеграл сходился.

На практике чаще всего используют готовые таблицы, по которым можно сразу определить изображение по оригиналу и наоборот.

Преобразование Лапласа имеет несколько замечательных свойств, например, преобразование Лапласа производной функции f(t):

$$L\left\{\frac{df(t)}{dt}\right\} = s \cdot F(s) - f(0), \quad (42)$$

или при нулевых начальных условиях изображение производной равно изображению самой функции, умноженному на s. Аналогично для построения изображения i -ой производной нужно умножить изображение функции на s^i (это также справедливо только при нулевых начальных условиях).

Рассмотрим снова уравнение (35) и применим к левой и правой частям преобразование Лапласа, считая, что все начальные условия нулевые. Получается уравнение в изображениях, связывающее преобразования Лапласа входа X (s) и выхода Y(s):

$$b_2 \cdot s^2 \cdot y(t) + b_1 \cdot s \cdot y(t) + b_0 \cdot y(t) = a_1 \cdot s \cdot x(t) + a_0 \cdot x(t). \quad (43)$$

Выразим из уравнения изображение функции выходного сигала Y(s):

$$Y(s) = \frac{(a_1\cdot s + a_0)}{(b_2\cdot s^2 + b_1\cdot s + b_0)} \cdot X(s), \quad (44)$$

или

$$Y(s) = W(s) \cdot X(s), \quad (45)$$

где W(s) – передаточная функция, записанная в виде функции от комплексной переменной s, а не от оператора дифференцирования.

Таким образом, при нулевых начальных условиях изображение выхода линейного объекта вычисляется как произведение его передаточной функции на изображение входного сигнала.

Из (45) следует, что передаточная функция равна отношению изображений по Лапласу выхода и входа при нулевых начальных условиях.

Рассмотрим пример использования преобразования Лапласа для вычисления выхода системы при известном входном сигнале. Пусть объект управления описывается уравнением первого порядка:

$$T \cdot \frac{dy(t)}{dt} + y(t) = k \cdot x(t) \qquad (46)$$

и на его вход поступает единичный ступенчатый сигнал x(t) = 1(t) . Требуется найти сигнал выхода y(t).

Решим эту задачу с помощью передаточных функций и изображений сигналов по Лапласу. Чтобы найти изображение выхода, нужно знать изображение входного сигнала X (s) и передаточную функцию звена W(s). Изображение входа находим по табличным данным (таблицы преобразования Лапласа для типовых функций), а передаточную функцию – из (46), повторяя приведенные выше рассуждения:

$$X(s) = \frac{1}{s}; W(s) = \frac{k}{T \cdot s + 1}.$$

Отсюда, изображение выхода

$$Y(s) = W(s) \cdot X(s) = \frac{k}{T \cdot s + 1} \cdot \frac{1}{s} = \frac{k}{s^2 + \frac{1}{T} \cdot s}. \qquad (47)$$

Выходной сигнал y(t) найдет, используя обратное преобразование Лапласа от Y(s), для чего снова обратимся к таблицам преобразования Лапласа для типовых функций, получим:

$$y(t) = L^{-1}\{Y(s)\} = L^{-1}\left\{\frac{k}{s^2+\frac{1}{T}\cdot s}\right\} = k - k \cdot e^{-\frac{t}{T}}. \quad (48)$$

Таким способом можно вычислять реакцию системы на известный входной сигнал без прямого решения дифференциального уравнения.

3.2 Линеаризация нелинейных динамических объектов

Во многих случаях более или менее точные модели представляют собой нелинейные дифференциальные уравнения, поэтому для того, чтобы применить теорию линейных систем, требуется линеаризация. При этом применяется почти та же методика, что и для алгебраических уравнений.

Представим себе бак с водой (рисунок 22). В нижней части бака просверлено отверстие, через которое вытекает вода. Площадь сечения бака обозначим через S, а площадь сечения отверстия – через S_0.

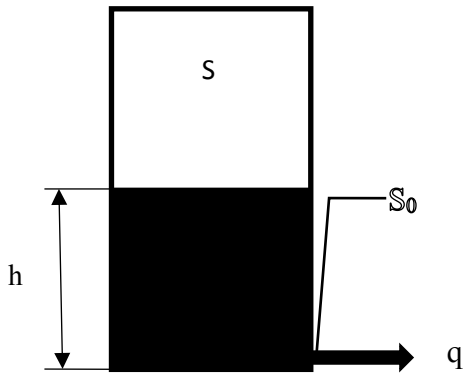


Рисунок 22 – Схематичное изображение параметров модели свободного истечения жидкости из вертикальной цилиндрической емкости

Построим модель, которая связывает уровень воды в баке h (в метрах) и расход вытекающей воды q (в м3/с). Эту связь можно найти с помощью закона Бернулли, который в данном случае принимает вид

$$\rho \cdot g \cdot h = \rho \cdot \frac{v^2}{2}, \tag{49}$$

где ρ – плотность жидкости (в кг/м3),

g ≈ 9,81м/с^2 – ускорение свободного падения,

v – скорость вытекания жидкости (в м/с).

Отсюда получаем

$$v = \sqrt{2 \cdot g \cdot h} \tag{50}$$

Учитывая, что расход воды вычисляется как

$$q = S_0 \cdot v, \tag{51}$$

находим

$$q = \alpha \cdot \sqrt{h}, \tag{52}$$

где α – постоянный коэффициент равный $\alpha = S_0 \cdot \sqrt{2 \cdot g}$.

Это статическая модель, потому что она не содержит производных, характеризующих изменение сигналов во времени. Статическая модель описывает установившееся состояние (статический режим), когда в баке поддерживается постоянный уровень воды и поток вытекающей воды тоже постоянный.

Очевидно, что полученная модель – нелинейная, поскольку содержит h^{0,5}. Линеаризовать ее – значит приближенно заменить уравнение (52) линейным уравнением

q = k·h , где k – некоторый коэффициент. Как его выбрать? На этот вопрос нет однозначного ответа.

Предположим, что уровень воды изменяется в интервале от 0 до 1 м. Тогда один из вариантов – вычислить коэффициент как угол наклона отрезка, соединяющего точки кривой q =α $h^{0,5}$ на концах этого интервала. Для определенности далее везде принимаем α = 1, тогда получаем k =1.

Конечно, эта модель очень грубая и дает большую ошибку, особенно для уровней в диапазоне от 0,1 до 0,6. Чтобы уменьшить ошибку, можно попробовать несколько изменить k (например, увеличив его до 1,2), однако точность приближения по-прежнему будет невысока, хотя и чуть-чуть лучше, чем в первом случае.

Теперь предположим, что обычно уровень мало изменяется вблизи среднего значения h = 0,5 м. В этом случае можно применить другой подход. Заметим, что в этой области кривая q =α $h^{0,5}$ почти совпадает с касательной в точке $(0{,}5;\ \frac{\sqrt{2}}{2})$, угол наклона которой равен производной

$$k = \frac{dq}{dh}_{\lfloor h=0,5} = \frac{1}{2\cdot\sqrt{h}}_{\lfloor h=0,5} = \frac{\sqrt{2}}{2} \tag{53}$$

Касательная – это прямая с наклоном k, проходящая через точку $(0{,}5;\ \frac{\sqrt{2}}{2})$, ее уравнение имеет вид q = = k·h + b. Свободный член b определим из равенства:

$$\frac{\sqrt{2}}{2} = k \cdot h + b = \frac{\sqrt{2}}{2} \cdot 0{,}5 + b, \tag{54}$$

$$b = \frac{\sqrt{2}}{4}. \tag{55}$$

Таким образом получаем модель вида:

$$q = \frac{\sqrt{2}}{2} \cdot h + \frac{\sqrt{2}}{4}\,. \tag{56}$$

Уравнение (56) линейное, однако модель – нелинейна, так как для нее не выполняется принцип умножения на константу:

$$f(const \cdot x) = const \cdot f(x). \quad (57)$$

Для того, чтобы получить линейную модель, нужно записать уравнения в отклонениях от рабочей точки $(0{,}5;\ \frac{\sqrt{2}}{2})$, в которой мы определяли наклон касательной. Обозначим $q = q_0 + \Delta q, h = h_0 + \Delta h$. Получим

$$q_0 + \Delta q = \frac{\sqrt{2}}{2} \cdot h_0 + \frac{\sqrt{2}}{2} \cdot \Delta h + \frac{\sqrt{2}}{4}, \quad (58)$$

но

$$q_0 = \frac{\sqrt{2}}{2} \cdot h_0 + \frac{\sqrt{2}}{4}, \quad (59)$$

тогда линейная модель, записанная в отклонениях от рабочей точки будет выглядеть

$$\Delta q = \frac{\sqrt{2}}{2} \cdot \Delta h. \quad (60)$$

Приближенная модель (60) точнее всего соответствует объекту вблизи рабочей точки, а при больших отклонениях от нее ошибка может значительно возрастать. Заметим, что от выбора рабочей точки завися и коэффициенты k и b линеаризованного уравнения модели.

Отметим, что полученная нами модель бака относится к классу статических моделей, так как описывает только установившиеся режимы работы емкости, не учитывая переходных процессов, возникающих при изменении входного уровня воздействия.

Модель, только что построенная для бака, не совсем правильная, потому что не учитывает, что уровень в баке изменяется – уменьшается по мере вытекания воды.

Кроме того, предположим, что для поддержания уровня используется насос, который подкачивает воду в бак, его расход обозначим через Q. Для такого объекта входом является расход Q, а выходом – изменение уровня h (рисунок 23).

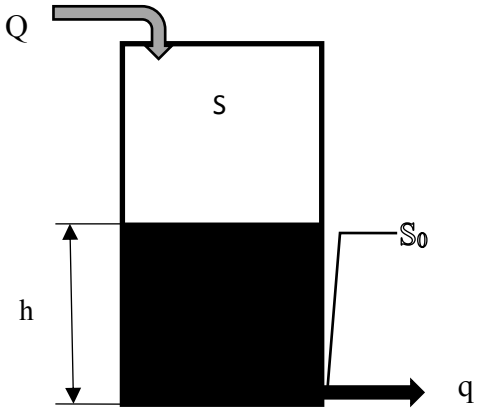


Рисунок 23 – Схематичное изображение параметров модели вертикальной цилиндрической емкости

Предположим, что в течение маленького интервала Δt расходы Q и q можно считать постоянным. За это время объем воды, добавленной в бак насосом, равен Q·Δt, а объем «ушедшей» воды – q·Δt. Учитывая, что площадь сечения бака равна S, получаем изменение уровня:

$$\Delta h = \frac{Q-q}{S} \cdot \Delta t. \qquad (61)$$

Переходя к пределу при $\Delta t \rightarrow 0$ получаем дифференциальное уравнение:

$$\frac{dh}{dt} = \frac{1}{S} \cdot [Q(t0 - q(t)]. \qquad (62)$$

Модель (62) учитывает, что уровень жидкости в баке и расходы жидкости изменяются во времени. Зависимость расхода вытекающей жидкости q(t) от уровня жидкости в баке h(t) нелинейна и найдена ранее (см. уравнения (49) – (52)). Подставим (52) в (62) и раскроем скобки в правой части уравнения, получим:

$$\frac{dh}{dt} = \frac{1}{S} \cdot Q(t) - \frac{\alpha}{S} \cdot \sqrt{h(t)}. \quad (63)$$

Выберем рабочую точку с координатами (Q_0; h_0) и предположим, что параметры расход подачи и уровень жидкости изменяются в окрестностях рабочей точки с малыми отклонениями $\Delta Q, \Delta h$. Для простоты записи не будем далее явно указывать зависимость параметров Q и h от времени.

Для линеаризации динамической модели будем использовать разложение функции в ряд Тейлора:

$$f(x,y) = f(x_0,y_0) + \frac{\partial f(x_0,y_0)}{\partial x} \cdot \Delta x + \frac{\partial f(x_0,y_0)}{\partial y} \cdot \Delta y + F(x,y), \quad (64)$$

где F(x, y) – частные производные более высоких порядков.

При малых значениях Δx и Δy можно считать, что функция F(x, y) ничтожно мала. Пренебрегая ей. получим:

$$f(x,y) \approx f(x_0,y_0) + \frac{\partial f(x_0,y_0)}{\partial x} \cdot \Delta x + \frac{\partial f(x_0,y_0)}{\partial y} \cdot \Delta y \quad (65)$$

Применив формулу (65) к правой части уравнения (63), получим:

$$\frac{1}{S} \cdot Q - \frac{\alpha}{S} \cdot \sqrt{h} \approx \frac{1}{S} \cdot Q_0 - \frac{\alpha}{S} \cdot \sqrt{h_0} + \frac{1}{S} \cdot \Delta Q - \frac{\alpha}{2 \cdot S \cdot \sqrt{h_0}} \cdot \Delta h. \quad (66)$$

Подставим $Q = Q_0 + \Delta Q$ и $h = h_0 + \Delta h$, а также учтем, что

$$\frac{d(h_0+\Delta h)}{dt} = \frac{d\Delta h}{dt}, \quad (67)$$

$$\frac{dh_0}{dt} = \frac{1}{S} \cdot Q_0 - \frac{\alpha}{S} \cdot \sqrt{h_0} = 0 \quad (68)$$

получим линеаризованное и записанное в отклонениях от рабочей точки уравнение динамической модели бака:

$$\frac{d\Delta h}{dt} + \frac{\alpha}{2 \cdot S \cdot \sqrt{h}} \cdot \Delta h \approx \frac{1}{S} \cdot \Delta Q \qquad (69)$$

В данном случае, коэффициенты линеаризации также будут зависеть от выбора рабочей точки, а модель будет справедлива только при малых отклонениях параметров от рабочих.

3.3 Типовые возмущения, применяемые при исследовании динамических объектов. Функция отклика

Один из методов построения моделей «вход-выход» – определение реакции объекта на некоторый стандартный сигнал. Один из простейших сигналов – так называемый «единичный скачок» (единичный ступенчатый сигнал или функция Хэвисайда), то есть мгновенное изменение входного сигнала с 0 до 1 в момент t = 0. Формально этот сигнал определяется так:

$$1(t) = \begin{cases} 0, & t \leq 0 \\ 1, & t \geq 0 \end{cases}. \qquad (70)$$

Реакция объекта на единичный скачок называется переходной функцией и обозначается h(t) (рисунок 24).

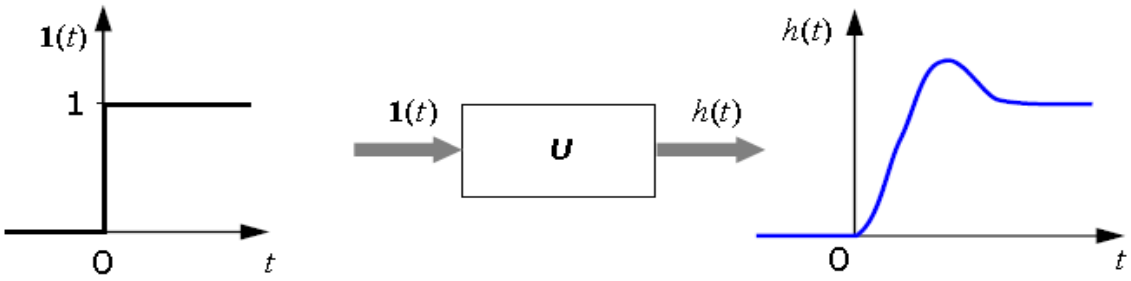


Рисунок 24 – Графическое изображение единичного скачка и отклика объекта

При этом предполагается, что объект в начальный момент находится в состоянии покоя, то есть, имеет нулевые начальные условия. Это значит, что все его переменные состояния равны нулю и внутренняя энергия также нулевая.

Если начальные условия ненулевые, то для построения сигнала выхода при любом входе нужно использовать дифференциальные уравнения объекта. Это значит, что переходная характеристика дает меньше информации, чем исходные уравнения.

Заметим, что ступенчатый сигнал легко получить на практике, поэтому переходную характеристику можно снять экспериментально.

В качестве тестового сигнала можно, в принципе, использовать любой сигнал. Например, можно изучать реакцию объекта на прямоугольный импульс. Вопрос в том, чтобы определить некоторый стандартный вид этого импульса. На рисунке 25 (а-в) показаны три импульса, имеющих одинаковые площади. Для простоты будем считать, что эта площадь равна единице.

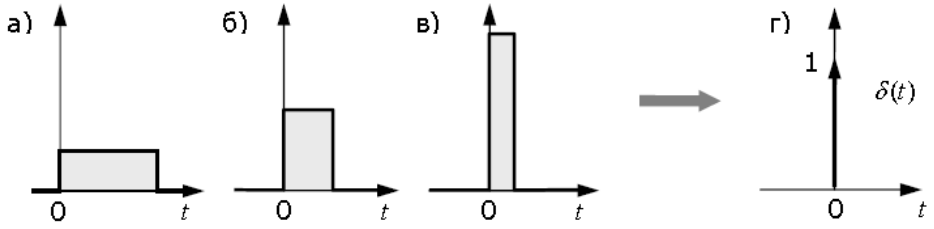


Рисунок 25 – Графическое изображение прямоугольных импульсных сигналов равной площади

Что будет, если мы будем уменьшать ширину импульса, сохраняя его площадь? Очевидно, что высота импульса будет расти и в пределе (когда ширина стремится к нулю) станет бесконечной. Таким образом, мы получили еще один классический тестовый сигнал – единичный импульс или дельта-функцию Дирака δ(t). Это идеальный (невозможный в реальности) сигнал, который равен нулю во всех точках, кроме t = 0, где он уходит в бесконечность, причем его площадь (интеграл по всей оси времени) равна единице:

$$\delta(t) = \begin{cases} \infty, & t = 0 \\ 0, & t \neq 0 \end{cases}, \int_{-\infty}^{+\infty} \delta(t)dt = 1. \quad (71)$$

Поскольку бесконечный импульс невозможно нарисовать, на графике он изображается стрелкой, высота которой равна единице (рисунок 26).

Иногда определяют дельта-функцию как производную от единичного ступенчатого сигнала 1(t). Действительно, эта производная равна нулю при всех значениях t, кроме нуля, где она обращается в бесконечность.

Реакция системы на единичный импульс (дельта-функцию) называется импульсной характеристикой и обозначается w(t).

Импульсная характеристика, так же, как и переходная характеристика, определяется при нулевых начальных условиях, то есть, объект должен находиться в состоянии покоя. Рассматривая дельта-функцию как предельный случай прямоугольного сигнала единичной площади, можно найти связь между переходной функцией и импульсной характеристикой.

Импульсная характеристика равна производной от переходной функции. Наоборот, переходная функция – это интеграл от импульсной характеристики на интервале от 0 до t.

Другое название импульсной характеристики – весовая функция. Это название связано с тем, что для произвольного входного сигнала x(t) выход системы y(t) при нулевых начальных условиях вычисляется как интеграл

$$y(t) = \int_{-\infty}^{t} x(\tau) \cdot w(t-\tau)d\tau = \int_{0}^{\infty} x(t-\tau) \cdot w(\tau)d\tau \quad (72)$$

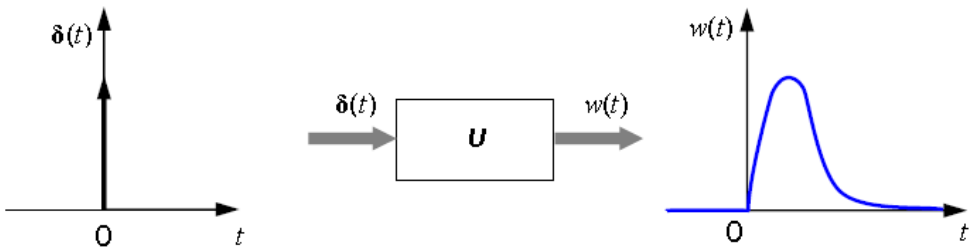


Рисунок 26 – Графическое изображение единичного импульса и отклика объекта

Здесь функция w(t) как бы «взвешивает» входной сигнал x(t) в подынтегральном выражении. Заметим, что импульсная характеристика дает неполную информацию об объекте, поскольку не учитывает ненулевые начальные условия. В отличие от ступенчатого сигнала, мгновенный импульс бесконечной величины невозможно получить на реальном устройстве, поэтому снять импульсную характеристику системы экспериментально без искажений не удается.

Еще один популярный эталонный сигнал – гармонический (синус, косинус), например:

$$x(t) = \sin \omega \cdot t,$$

где ω – угловая частота (в радианах в секунду).

Можно показать, что при таком входе на выходе линейной модели в установившемся режиме (при больших t) будет синус той же частоты, но с другой амплитудой A и сдвигом фазы φ:

$$y(t) = A(\omega) \cdot sin(\omega t + \varphi(\omega)).$$

Для каждой частоты входного сигнала будет своя амплитуда и свой сдвиг фазы. Чтобы определить по графику фазовый сдвиг φ, нужно найти расстояние Δt по оси времени между соответствующими точками синусоид (например, точками пересечения с осью t или вершинами). Если Δt умножить на частоту ω, получаем сдвиг фазы φ (в радианах).

На рисунке 27 показан случай φ > 0 (опережение по фазе), когда выход сдвинут «влево» по оси времени относительно входа, то есть, «идет раньше» входного.

Модель практически любого объекта можно охарактеризовать частотными характеристиками. Рассмотрим подробнее каждую из них.

Амплитудная характеристика (АЧХ) – зависимость отношения амплитуд выходного сигнала и входного от частоты входного гармонического сигнала.

Фазовая частотная характеристика (ФЧХ) – зависимость фазового сдвига между входным и выходным сигналами от частоты. ФЧХ показывает, какое отставание или опережение выходного сигнала по фазе создает элемент при различных частотах.

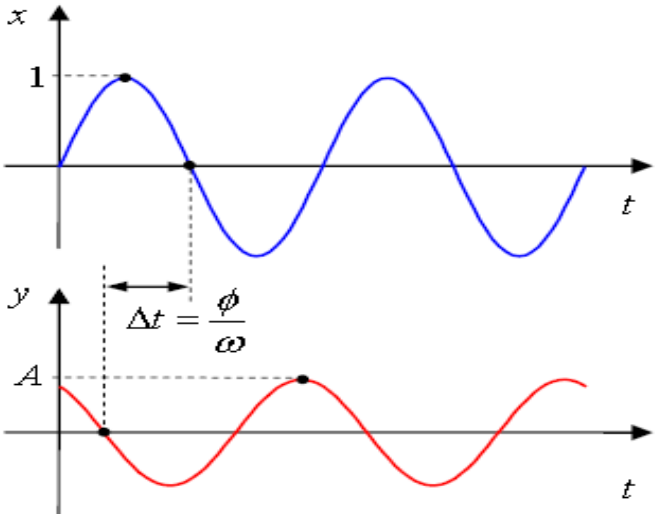


Рисунок 27 – Графическое определение фазового сдвига

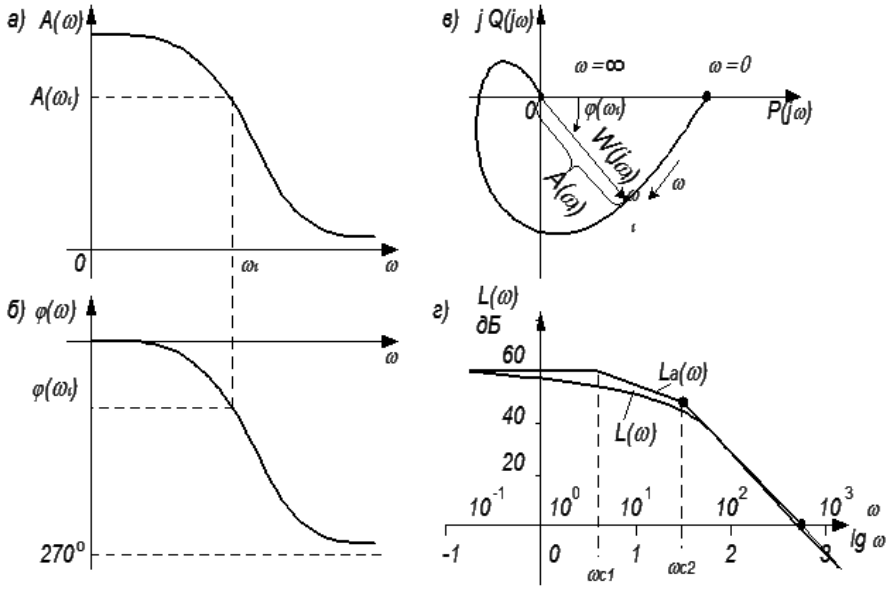
Амплитудную и фазовую характеристики можно объединить в одну общую – амплитудно-фазовую частотную характеристику (АФЧХ). АФЧХ представляет собой функцию комплексного переменного $j \cdot \omega$:

$$W(j \cdot \omega) = A(\omega) \cdot e^{j \cdot \varphi(\omega)}, \quad (73)$$

где A(ω) – модуль функции; φ(ω) – аргумент функции.

Каждому фиксированному значению частоты ω_i соответствует комплексное число $W(j\omega_i)$, которое на комплексной плоскости можно изобразить вектором, имеющим длину $A(\omega_i)$ и угол поворота $\varphi(\omega_i)$ (рисунок 28). Отрицательные значения $\varphi(\omega_i)$, соответствующие отставанию выходного сигнала от входного, принято отсчитывать по часовой стрелке от положительного направления действительной оси.

При изменении частоты от нуля до бесконечности вектор $W(j\omega_i)$ поворачивается вокруг начала координат, при этом одновременно изменяется длина вектора. Кривая, которую при этом опишет конец вектора, и есть АФЧХ. Каждой точке характеристики соответствует определенное значение частоты.



а – амплитудная; б – фазовая; в – амплитудно-фазовая; г – логарифмическая

Рисунок 28 – Частотные характеристики

Таким образом, частотная передаточная функция $W(j\omega)$ представляет собой комплексное число, модуль которого равен отношению амплитуды выходной величины к амплитуде входной, а аргумент – сдвигу фаз выходной величины по отношению к входной.

3.4 Типовые передаточные функции

Современные производственные системы состоят из элементов различной физической природы, конструктивного исполнения, источников энергии и т. д. Однако динамические свойства этих элементов часто можно описать одним и тем же дифференциальным уравнением (ДУ). Положив в основу классификации динамические

свойства, обычно выделяют следующие звенья: усилительное, инерционные, колебательное, интегрирующее, дифференцирующее.

Усилительное звено

Оператор преобразования равен k:

$$y = k \cdot x \tag{74}$$

Инерционные (апериодические) звенья первого порядка, описываются ДУ вида:

$$T \cdot \frac{dy}{dt} + y = k \cdot x, \tag{75}$$

второго порядка:

$$T^2 \cdot \frac{d^2y}{d^2t} + 2 \cdot \xi \cdot T \cdot \frac{dy}{dt} + y = k \cdot x, \text{при } \xi > 1, \tag{76}$$

где k,T, ξ – некоторые постоянные уравнения.

Колебательное звено описывается ДУ такого же вида как инерционное звено второго порядка, но при $0 < \xi < 1$.

В статическом режиме (при равенстве нулю всех производных) все приведенные звенья имеют уравнение, аналогичное усилительному звену, что свидетельствует о наличии линейной связи между входной величиной x и выходной величиной y в статике. Поэтому все рассмотренные звенья относятся к классу статических.

Интегрирующее звено описывается выражением:

$$\frac{dy}{dt} = k \cdot x. \tag{77}$$

Здесь выходная величина y будет изменяться до тех пор, пока входная величинах не станет равной нулю.

Дифференцирующее звено описывается ДУ вида:

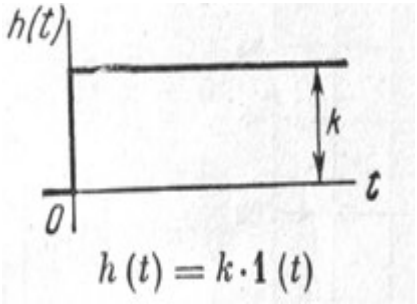
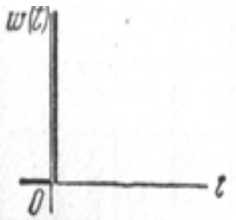
$$y = k \cdot \frac{dx}{dt}. \qquad (78)$$

Последние два звена не имеют однозначной связи между входными и выходными величинами в статике, поэтому относятся к классу астатических.

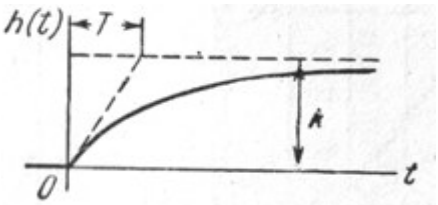
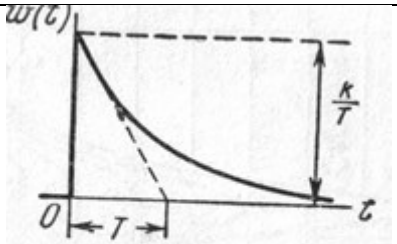
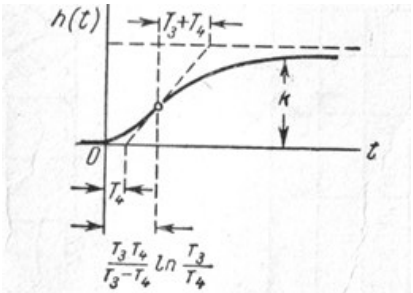
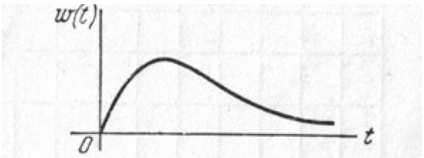
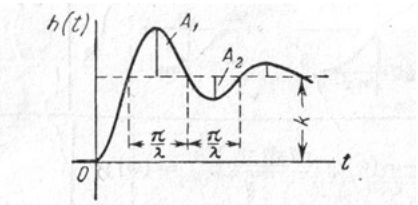
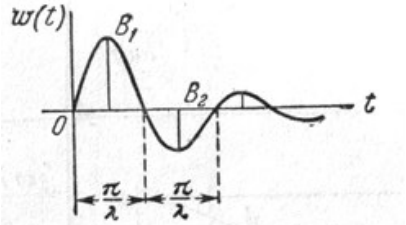
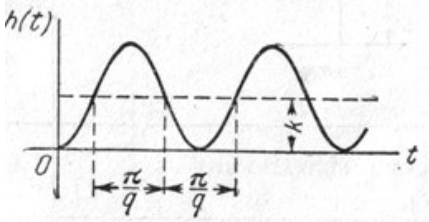
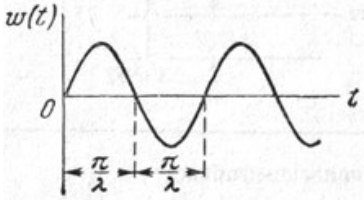
Реальные звенья могут описываться уравнением и выше второго порядка, но формально это описание можно заменить системой уравнений, каждое из которых имеет порядок не выше второго, и таким образом, представить реальное звено в виде звеньев, упомянутых ранее. Поэтому такие звенья обычно называются типовыми.

Характеристики статических типовых звеньев представлены в таблице 5.

Таблица 5 – Временные характеристики типовых звеньев

Наименование и передаточная функция	**Переходная функция**	**Весовая функция**
1	**2**	**3**
Безынерционное звено (усилительное) $W(p) = k$	 $h(t) = k \cdot 1(t)$	 $w(t) = k \cdot \delta(t)$

Продолжение таблицы 5

1	2	3
Апериодическое 1-го порядка $W(p) = \frac{k}{1 + Tp}$	 $h(t) = k\left(1 - e^{-\frac{t}{T}}\right) \cdot 1(t)$	 $w(t) = \frac{k}{T} e^{-\frac{t}{T}} \cdot 1(t)$
Апериодическое 2-го порядка $W(p) = \frac{k}{1 + T_1 p + T_2^2 p^2} = \frac{k}{(1 + T_3 p)(1 + T_4 p)}$, $T_{3,4} = \frac{T_1}{2} \pm \sqrt{\frac{T_1^2}{4} - T_2^2}$ $(T_1 > 2T_2;\quad T_3 > T_4)$	 $h(t) = k\left(1 - \frac{T_3}{T_3 - T_4} e^{-\frac{t}{T_3}} + \frac{T_4}{T_3 - T_4} e^{-\frac{t}{T_4}}\right) \cdot 1(t)$	 $w(t) = \frac{k}{T_3 - T_4} \left(e^{-\frac{t}{T_3}} - e^{-\frac{t}{T_4}}\right) \cdot 1(t)$
Колебательное звено $W(p) = \frac{k}{1 + 2\zeta T p + T^2 p^2} = \frac{k}{1 + \frac{2\zeta}{q} p + \frac{p^2}{q^2}}$, $q = \frac{1}{T}$	 $\gamma = \frac{\lambda}{\pi} \ln \frac{A_1}{A_2}$, $\gamma = \zeta q$, $\lambda = q\sqrt{1 - \zeta^2}$, $h(t) = k\left[1 - e^{-\gamma t}\left(\cos \lambda t + \frac{\gamma}{\lambda} \sin \lambda t\right)\right] \cdot 1(t)$	 $\gamma = \frac{\lambda}{\pi} \ln \frac{B_1}{B_2}$, $q = \sqrt{\gamma^2 + \lambda^2}$, $\zeta = \frac{\gamma}{\sqrt{\gamma^2 + \lambda^2}}$, $w(t) = \frac{kq^2}{\lambda} e^{-\gamma t} \sin \lambda t \cdot 1(t)$
Консервативное звено $W(p) = \frac{k}{1 + T^2 p^2} = \frac{k}{1 + \frac{p^2}{q^2}}$, $q = \frac{1}{T}$	 $h(t) = k(1 - \cos qt) \cdot 1(t)$	 $w(t) = kq \sin qt \cdot 1(t)$

Таким образом, зная временные свойства типовых моделей можно по известному отклику реального объекта на стандартный сигнал предположить форму дифференциального уравнения и найти его коэффициенты, то есть составить математическое описание динамических свойств объекта или процесса.

3.5 Модель работы напорного бака в Matlab

Рассмотрим реализацию модели напорного бака, описанного в разделе 3.2. Воспользовавшись формулами (49) - (63) построим модель в Simulink. Для этого создадим новую модель: «File» – «New»- «Model».

В открывшемся окне из стандартных блоков библиотеки компонентов («View» – «Library Browser») создадим следующую схему, отображенную на рисунке 29:

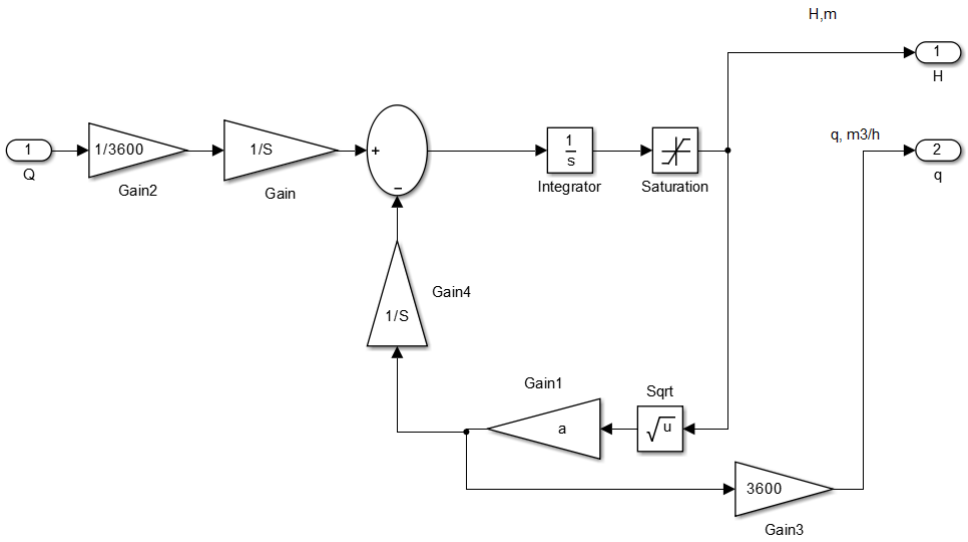


Рисунок 29 – Схема подсистемы расчета модели бака

При построении использованы: пять блоков «Gain», один интегратор, блок насыщения «Saturation», блок, выполняющий операцию извлечения корня «Sqrt», Один сумматор (блок «Sum»), один порт входа «In1», два выходных порта «Out1».

Блок «Gain2» переводит расход из $м^3/ч$ в $м^3/с$, потому в его настройках, в поле «Gain» необходимо установить значение «1/3600». Блок «Gain3» выполняет обратный

пересчет, а, следовательно, значение поля «Gain» устанавливается равным «3600». Блоки «Gain» и «Gain 4» настраиваются на значение коэффициента усиления «1/S» (см. формулы (49) – (63) п.3.2). У оставшегося блока «Gain» устанавливаем коэффициент усиления равный «a». Блок извлечения корня оставляем без изменения параметров по умолчанию, порты ввода-вывода переименовываем для наглядности обозначений так, как показано на рисунке 29. При необходимости на линиях связи делаем поясняющие надписи.

В настройках блока «Saturation» вносим следующие изменения: в поле «Upper limit» устанавливаем значение «H», в поле «Lower limit» устанавливаем значение «0».

Далее, курсором мыши выделяем всю схему и по правому щелчку мыши в контекстном меню выбираем «Create Subsystem from Selection», тем самым объединяем всю схему в одну компактную подсистему.

Дополняем модель стандартными блоками до вида, показанного на рисунке 30.

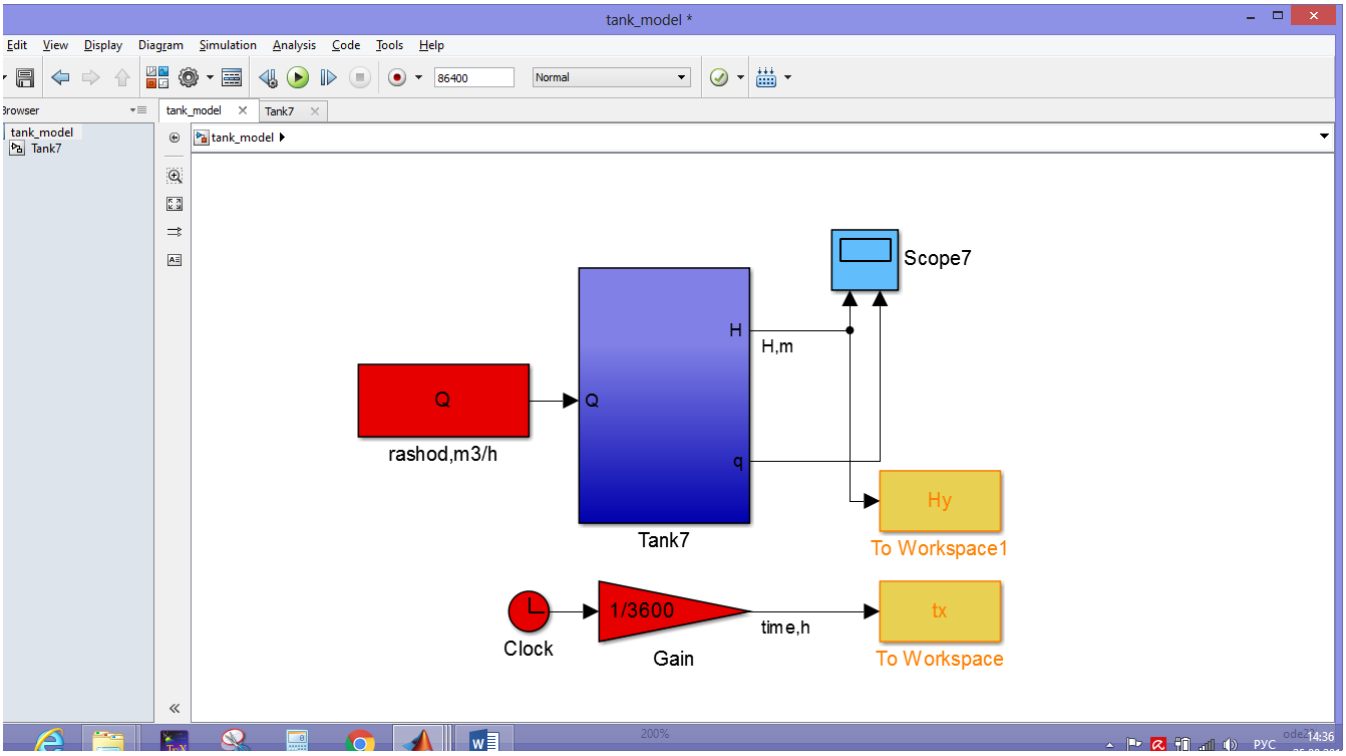


Рисунок 30 – Схема модели бака

Для этого используем стандартные блоки: «Gain», «Clock», «Constant», «To Workspace», «Scope». В настройках усилителя «Gain» устанавливаем значение «1/3600» для вывода времени, выраженного в часах, в настройках блока «Constant» устанавливаем значение «Q» в поле «Constant value:», в настройках бока «To Workspace1» устанавливаем имя переменной для экспорта «Hy» в поле «Variable name:», в блоке «To Workspace» в аналогичном поле – «tx». Графопостроитель «Scope» настраиваем по своему усмотрению.

Модель бака построена, однако запуск симуляции приведет к возникновению ошибки – мы использовали неопределенные ранее переменные «Q», «S», «H», «a» в настройках блоков, вместо подстановки числовых значений.

Объявим эти переменные и проведем предварительный расчет параметров модели. Данные манипуляции можно выполнить в командной строке окна «Workspace», однако, для автоматического запуска данных команд мы будем использовать процедуру встраивания функции в саму модель.

Откроем инспектор модели, вызвав меню «Tools» – «Model Explorer». В области «Model Properties:» активируем вкладку «Callbacks». В списке «Model Callbacks» выбираем «Preload Fcn» и справа, в поле ввода добавляем следующий код (рисунок 31):

```
%модель бака с водой.
% Ввод исходных данных
g=9.81; %m/s^2
V1=100; %m3
D1=10; %m
Q=100; %m3/h
do=80; %mm
%Расчет параметров модели
So=(pi*(do/1000)^2)/4;
S=(pi*D1^2)/4;
a=So*sqrt(2*g);
```

H=V1/S;

Qs=Q/3600;

Ho=(Qs/a)^2;

Наш бак задан следующими параметрами:

- объем: 100 м^3;

- диаметр: 10 м;

- диаметр отверстия истечения жидкости: 80 мм.

Расход подачи жидкости в бак составляет 100 м^3/ч.

При необходимости, параметры модели можно менять как используя инспектор модели, так и непосредственно после запуска в редакторе переменных окна «Workspace».

После выполнения данных процедур, создадим функцию по графическому отображению результатов работы модели. В инспекторе модели в правой части выберем поле «Stop Fcn» и введем следующий код, который будет выводить график изменения уровня жидкости в баке от времени по завершении симуляции:

plot(tx,Hy)

title('Уровень жидкости в баке');

xlabel('время, ч');

ylabel('Уровень,м');

grid on

Сохраняем модель и перезапускаем её. Результат работы модели показан на рисунках (32) – (33).

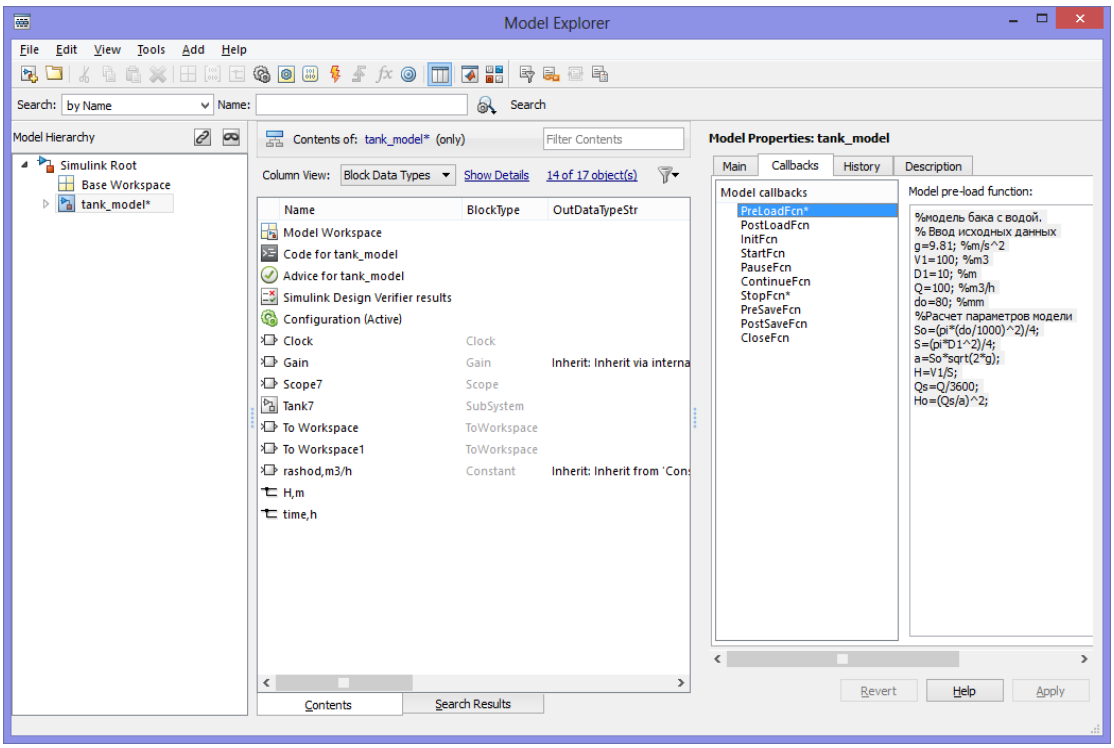


Рисунок 31 – Окно инспектора модели бака

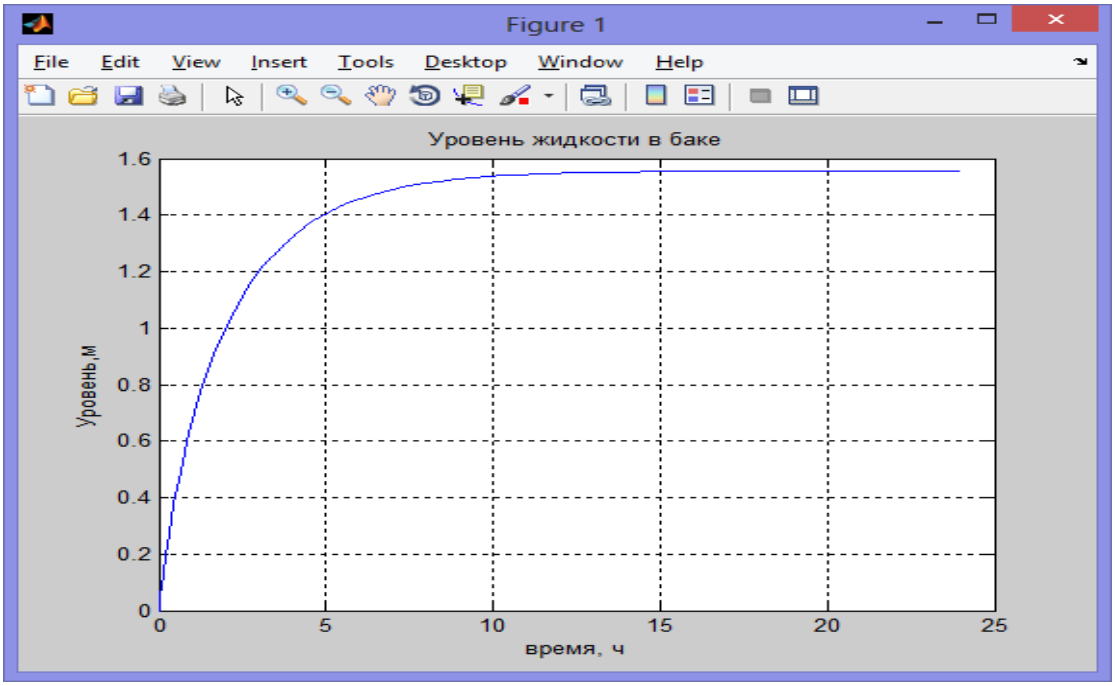


Рисунок 32 – Результат работы модели

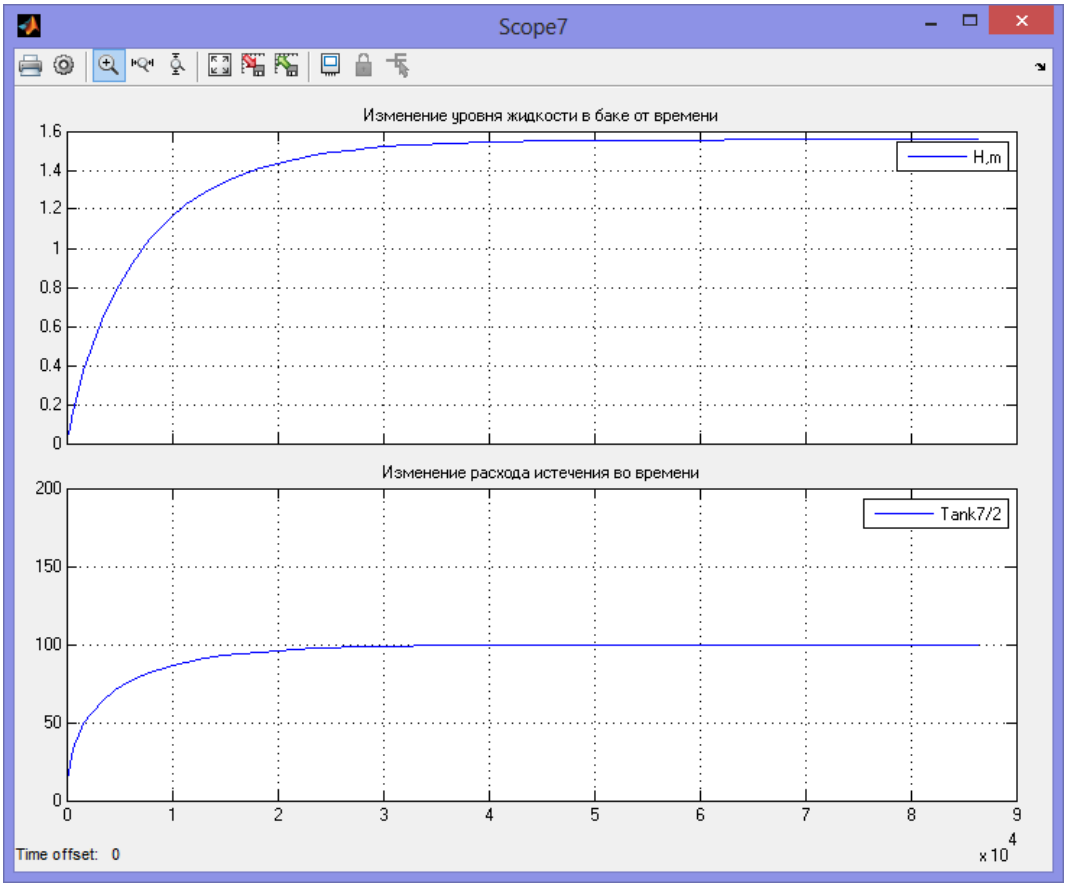


Рисунок 33 – Результат работы модели, блок «Scope» (время в секундах)

3.6 Модель осаждения полидисперсной смеси в Matlab

Одними из наиболее распространенных методов разделения дисперсных систем являются их отстаивание под действием силы тяжести или помещение в центробежное поле. Эти процессы получили название седиментация. В их основе лежит разность плотностей между дисперсными частицами и средой, в которой они находятся.

Для расчета широкого круга аппаратов, таких как отстойники и центрифуги, необходимо знать основных закономерности седиментационных процессов.

3.6.1 Теоретические основы седиментационных процессов

Законы осаждения частиц достаточно хорошо изучены для разбавленных дисперсных систем, концентрация которых невелика. При этом, из-за большого расстояния столкновения между частицами происходят весьма редко, и их взаимное влияние незначительно. Поэтому закономерности осаждения таких систем могут быть получены путем изучения процесса движения единичной частицы. Рассмотрим частицу, находящуюся в объеме дисперсионной среды. На нее действуют сила тяжести, направленная вниз, и сила Архимеда, направленная вверх. Сила тяжести равна

$$F_g = m_ч\, g, \tag{79}$$

где $m_ч$ – масса частицы, кг.

Масса частицы может быть определена через ее объем и плотность

$$m_ч = \rho_ч\, v_ч, \tag{80}$$

где $\rho_ч$ и $v_ч$ – – плотность частицы, кг/м^3 ; объем частицы, м^3.

Тогда сила тяжести будет вычисляться по следующей формуле

$$F_g = \rho_ч\, v_ч g \tag{81}$$

В свою очередь, сила Архимеда определяется выражением

$$F_a = \rho_{ср} v_ч\, g, \tag{82}$$

где $\rho_{ср}$ – плотность среды, кг/м^3.

Равнодействующая сила, называемая седиментационной силой, равна

$$F_{\text{сед}} = F_g - F_a \tag{83}$$

Подставляя выражения для сил тяжести и Архимеда в (83), получаем

$$F_{\text{сед}} = (\rho_{\text{ч}} - \rho_{\text{ср}}) v_{\text{ч}} g \tag{84}$$

Если плотность частиц больше плотности жидкости, то $F_{\text{сед}} > 0$, и частица осаждается. Этот процесс называется прямой седиментацией. Если плотность частиц меньше плотности жидкости, то $F_{\text{сед}} < 0$, и частица всплывает. Этот процесс называется обратной седиментацией. Когда частица под действием седиментационной силы приобретает скорость, наличие дисперсионной среды приводит в появлению силы трения, направленной противоположно скорости движения. Сила трения в общем случае может быть определена с помощью уравнения

$$F_{\text{тр}} = \zeta s \left(\frac{\rho_{\text{ср}} u^2}{2}\right), \tag{85}$$

где ζ, s, u – коэффициент сопротивления среды; площадь поперечного сечения частицы, м^2; скорость частицы, м/с.

В зависимости от параметров частицы и среды, а также скорости движения, обтекание может происходить в различных режимах. Каждому режиму соответствует свое значение коэффициента сопротивления ζ, зависящее от числа Рейнольдса

$$Re = \frac{u\, d\, \rho_{ср}}{\mu}, \tag{86}$$

где μ – динамическая вязкость среды, Па·с;

d – диаметр частицы, м.

При $Re \leq 2$ движение имеет ламинарный характер и зависимость между ζ и Re имеет вид

$$\zeta = 24\, Re^{-1} \tag{87}$$

При промежуточных значениях $(2 < Re \leq 500)$ в расчетах можно применять формулу Аллена

$$\zeta = 18{,}5 Re^{-0{,}6} \tag{88}$$

При турбулентном режиме движения, когда $Re > 500$, коэффициент сопротивления имеет постоянное значение

$$\zeta = 0{,}44 \tag{89}$$

Если частица имеет неправильную форму, то сопротивление со стороны среды возрастает. Коэффициент сопротивления для таких частиц может быть определен по формуле

$$\zeta' = k\zeta, \tag{90}$$

где k – поправочный коэффициент (коэффициент формы), значение которого больше единицы.

Таким образом, в первом приближении, суммарная сила, действующая на частицу в процессе ее движения, равна

$$F = F_{\text{сед}} - F_{\text{тр}} = \left(\rho_{\text{ч}} - \rho_{\text{ср}}\right) v_{\text{ч}} g - Bu, \tag{91}$$

где B – коэффициент трения.

В начальный период времени из-за малой скорости движения сила сопротивления среды также мала, и частица движется ускоренно. С ростом скорости в определенный момент времени сила трения возрастает настолько, что становится равной силе седиментации. При этом сумма всех сил, действующих на частицу, становится равной нулю. С этого момента времени частица движется с постоянной скоростью, равной

$$u = \frac{(\rho_ч - \rho_{ср})v_ч g}{B} \quad (92)$$

Для частиц сферической формы сила трения может быть вычислена по закону Стокса

$$F_{тр} = 3\pi d\mu u \quad (93)$$

Скорость осаждения сферических частиц при этом равна

$$u = \frac{g(\rho_ч - \rho_{ср})d^2}{18\mu} \quad (94)$$

Из выражения (94) видно, что скорость осаждения прямо пропорциональна разности между плотностями частиц и среды. Чем меньше размер частиц, тем меньше скорость их осаждения. Для очень мелких частиц скорость осаждения настолько мала, что их движению начинают препятствовать процессы конвекции жидкости и броуновского движения.

3.6.2 Седиментационные кривые

Представленные ранее закономерности позволяют построить зависимости массы осажденных частиц от времени, называемые седиментационными кривыми. Рассмотрим дисперсную систему, содержащую частицы одного размера, плотность

которых больше плотности среды. В начальный момент времени все частицы находятся во взвешенном состоянии и равномерно распределены по всему объему жидкости. В этот момент масса осажденных частиц равна нулю. С течением времени частицы осаждаются, и верхний слой жидкости очищается от них. При этом граница, разделяющая чистую жидкость от жидкости с частицами, все время равномерно перемещается вниз. Вследствие равномерности движения масса осажденных частиц линейно растет с течением времени t

$$M = \frac{M_0 t}{\tau}, \tag{95}$$

где M_0 – суммарная масса частиц, кг

τ – время полного осаждения частиц, с

t – варемя с начала опыта, с

График зависимости массы осажденных частиц для данного случая представлен на рисунке 34.

Рассмотрим теперь осаждение полидисперсной системы. В данном случае частицы каждой фракции будут осаждаться со своей скоростью. Масса осажденных частиц i-й фракции может быть определена следующим образом

$$M_i = \frac{M_{0i} t}{\tau_i}, \tag{96}$$

где M_{0i} – суммарная масса частиц i-й фракции, кг.

По истечении времени τ_i осаждение частиц данной фракции прекращается. Суммарная масса осевших частиц определяется путем суммирования

$$M_\Sigma = \sum_i M_i \tag{97}$$

Графики зависимости массы осажденных частиц от времени представлены на рисунке 35.

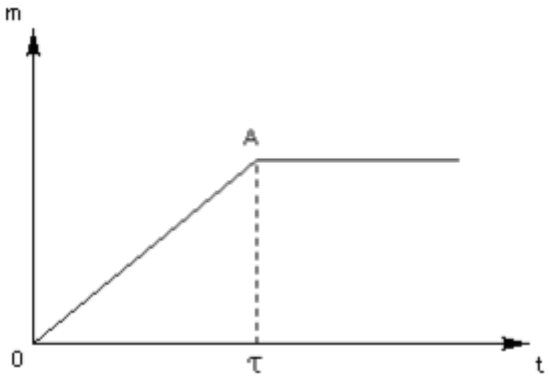


Рисунок 34 – Кривая осаждения монодисперсной системы

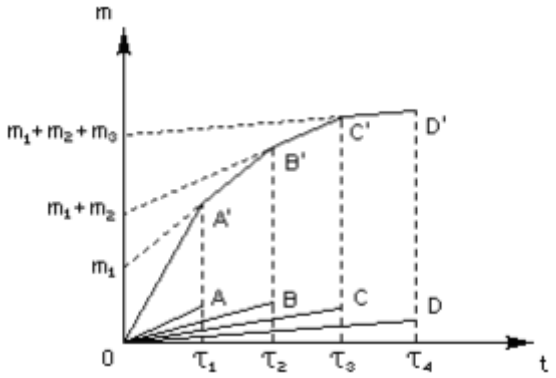


Рисунок 35 – Кривая осаждения монодисперсной системы

Реальные дисперсные системы, содержащие частицы самых различных размеров, имеют не ступенчатые, а гладкие кривые осаждения.

3.6.3 Седиментационный анализ дисперсного состава частиц

Если известна кривая седиментации дисперсной системы (например, полученная экспериментально), то с ее помощью можно построить распределение частиц по размерам. Если система содержит конечное число размеров частиц, то кривая седиментации имеет вид ломаной линии. Количество фракций, содержащихся в дисперсной системе, равно количеству изломов. Время τ_i, соответствующее каждому излому,

является временем осаждения одной из фракций частиц. Зная время осаждения фракции τ_i и высоту уровня жидкости h, можно определить скорость осаждения частиц каждой фракции по формуле

$$u_i = \frac{h}{\tau_i} \tag{98}$$

Зная скорость осаждения и свойства частиц и среды, можно определить размер частиц фракции по формуле

$$d_i = \sqrt{\frac{18\mu u_i}{g(\rho_ч - \rho_{ср})}} \tag{99}$$

Продолжая каждый линейный участок на графике седиментации до оси массы и измеряя интервалы масс на оси, можно определить суммарные массы частиц каждой из фракций M_i (рисунок 35).

Масса одной частицы каждой фракции равна

$$m_i = \frac{\rho_ч \pi d^3}{6} \tag{100}$$

Деля массу фракции на массу частицы, получаем количество частиц в фракции

$$N_i = \frac{M_i}{m_i} \tag{101}$$

Общее количество частиц в дисперсной системе будет равно

$$N_0 = \sum_i N_i \tag{102}$$

При этом доля фракции от общего количества частиц равна

$$x_i = \frac{N_i}{N_0} \tag{103}$$

Таким образом, определив значения d_i и x_i для всех фракций дисперсной системы, получаем распределение частиц по размерам.

3.6.4 Седиментационно-диффузионное равновесие

Все вышесказанное относится к дисперсным системам, содержащим частицы достаточно больших размеров. В системах с очень мелкими частицами существенную роль начинают играть процессы диффузии за счет броуновского движения частиц. Как известно, диффузионные процессы стремятся равномерно распределить частицы по всему объему среды. Когда происходит осаждение частиц под действием силы тяжести, их концентрация, n, в нижних слоях становится выше концентрации в верхних слоях. Поэтому возникает обратный диффузионный поток частиц, направленный вверх. Плотность потока частиц при этом равна

$$J_{\text{диф}} = -D\frac{dn}{dh} = -\frac{kT}{B}\frac{dn}{dh}, \tag{104}$$

где D – коэффициент диффузии;

h - высота;

k – постоянная Больцмана;

T – абсолютная температура.

Она представляет собой количество частиц, проходящих через единицу поверхности в единицу времени. Плотность потока частиц, обусловленная процессами седиментации, равна

$$J_{\text{сед}} = un = \frac{(\rho_{\text{ч}} - \rho_{\text{ср}})\nu g n}{B} \tag{105}$$

В определенный момент времени наступает равновесие между данными процессами, и потоки выравниваются

$$J_{\text{диф}} = J_{\text{сед}} \tag{106}$$

Подставляя (104) и (105) в (106), получаем

$$-kT\frac{dn}{dh} = (\rho_{\text{ч}} - \rho_{\text{ср}})vgn \tag{107}$$

Проводя разделение переменных, получим

$$\frac{dn}{n} = -\frac{(\rho_{\text{ч}}-\rho_{\text{ср}})vg}{kT}dh \tag{108}$$

Интегрируя по всей высоте, получаем

$$ln\frac{n}{n_0} = -\frac{(\rho_{\text{ч}}-\rho_{\text{ср}})vg}{kT}h, \tag{109}$$

где n_0 – концентрация частиц на нулевой высоте, м$^{-3}$.

Таким образом, зависимость равновесной концентрации частиц от высоты имеет вид

$$n = n_0 \exp\left(-\frac{(\rho_{\text{ч}}-\rho_{\text{ср}})vg}{kT}h\right) \tag{110}$$

Уравнение (110) называется гипсометрическим законом. Согласно этому закону наибольшая концентрация частиц достигается у дна сосуда с дисперсной системой и уменьшается по высоте по экспоненциальному закону.

3.6.5 Реализация модели в Matlab

Рассмотрим реализацию модели седиментации частиц на примере следующей задачи. Приготовлена смесь глинистых частиц следующего состава (таблица 6). Общая масса каждой фракции составляла 1 г.

Таблица 6 – Характеристика полидисперсной смеси

Средний диаметр частицы фракции, d_i, мкм	0,25	Средний диаметр частицы фракции, d_i, мкм	4,25
	0,75		6,75
	1,00		8,25
	1,25		12,75
	1,50		14,25
	1,75		16,25
	2,25		20,00
	2,75		25,00
	3,75		-

Необходимо смоделировать осаждение данной смеси в термостатическом цилиндре высотой 100 мм в среде с вязкостью 0,03865 Па·с и плотностью 1159 кг/м^3 при температуре осаждения (каменноугольная смола), плотность частиц глины составляет 1600 кг/м^3.

Для более точного решения, высоту цилиндра разобьём на 10 слоев, толщину каждого обозначим «Hsloia». Диаметр частицы обозначим «d», плотность частицы – «pch», плотность среды – «psmol», вызкость среды – «mu».

Реализуем формулы (94) для вычисления скорости осаждения фракции и (98) для вычисления времени полного осаждения фракции в слое в среде Simulink пакета Matlab. Вычисления будем проводить в подсистеме, назовем её «Skorost» (рисунок 36).

Для расчета массы частиц, оставшихся в слое и массы частиц, осевших за время моделирования, создадим подсистему «Massa_v_sloe» (рисунок 37) и реализуем в ней расчет осаждения частиц по формуле (96), полагая, что в начале процесса частицы распределены равномерно в каждом слое, а по прохождении высоты слоя частицы попадают на следующий слой (или оседают на дно цилиндра в последнем слое).

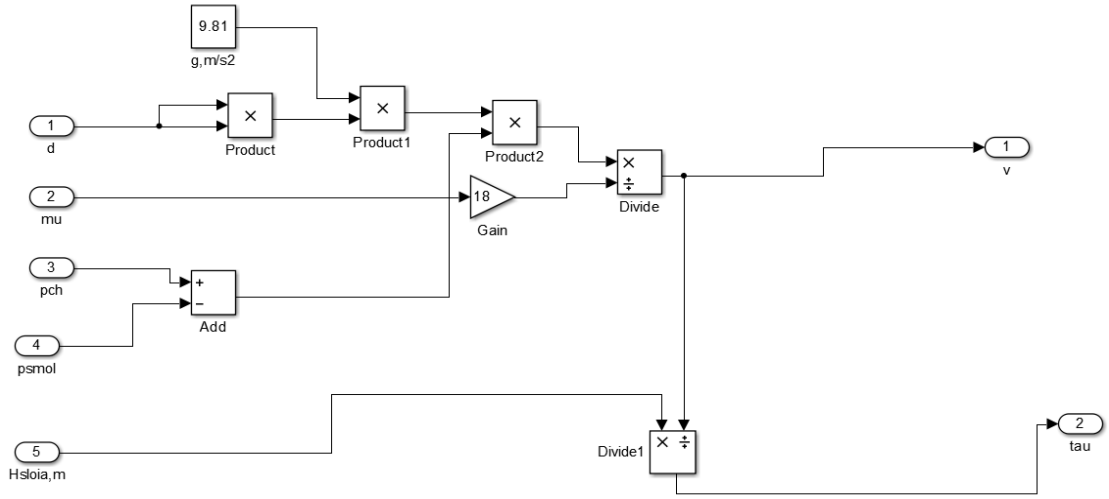


Рисунок 36 – Структура подсистемы «Skorost»

Для простоты, будем полагать, что в каждом слое частицы сосредоточены на его верхней границе, потому точность модели будет зависеть от частоты разбиения высоты цилиндра на слои. Чем большее количество слоев моделируется, тем точнее расчет.

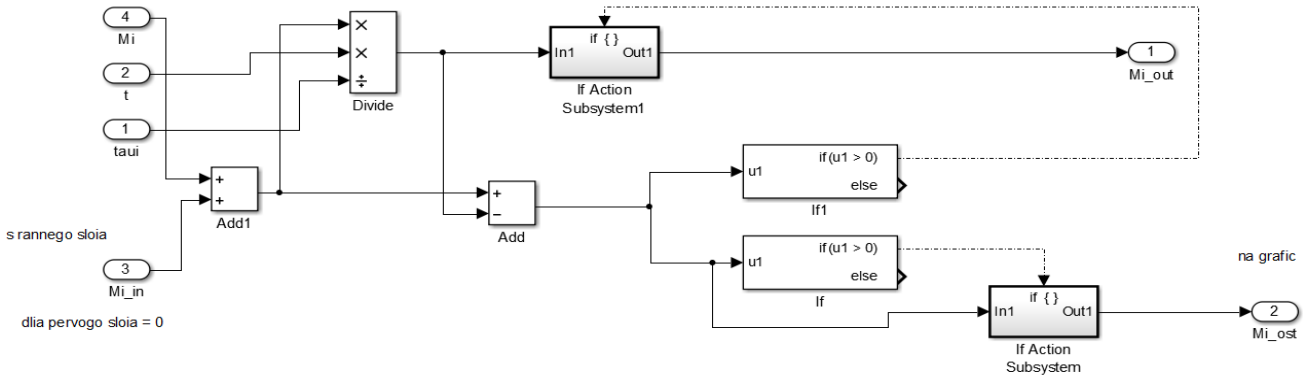


Рисунок 37 – Структура подсистемы «Massa_v_sloe»

Далее, скопируем и вставим 10 раз (по количеству слоев) наши подсистемы, соединим, как показано на рисунке 38, объединив в новую подсистему «d1» (рисунок 39).

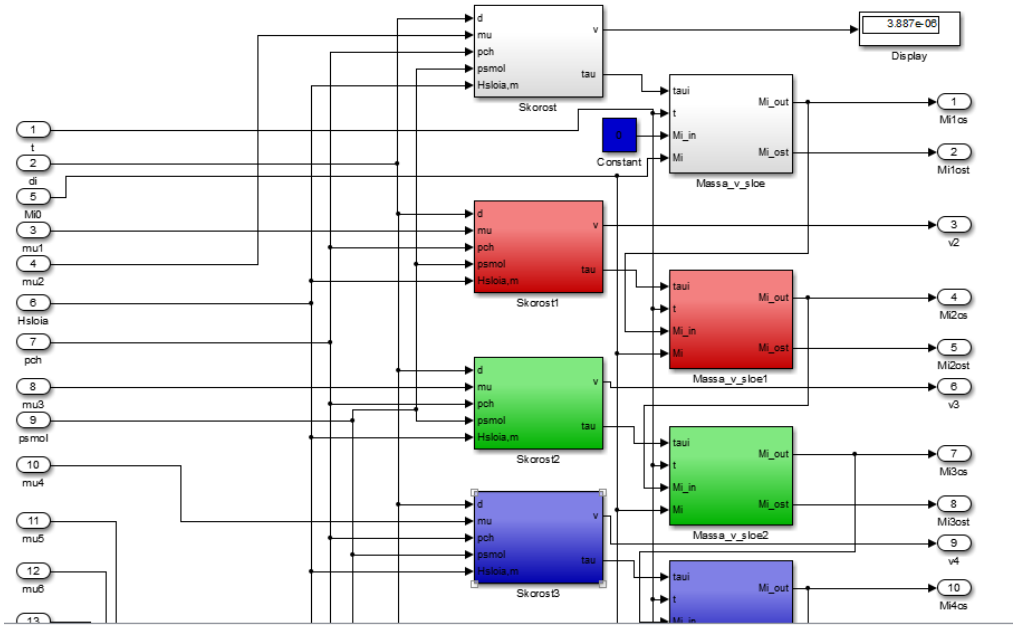


Рисунок 38 – Фрагмент подсистемы «d1»

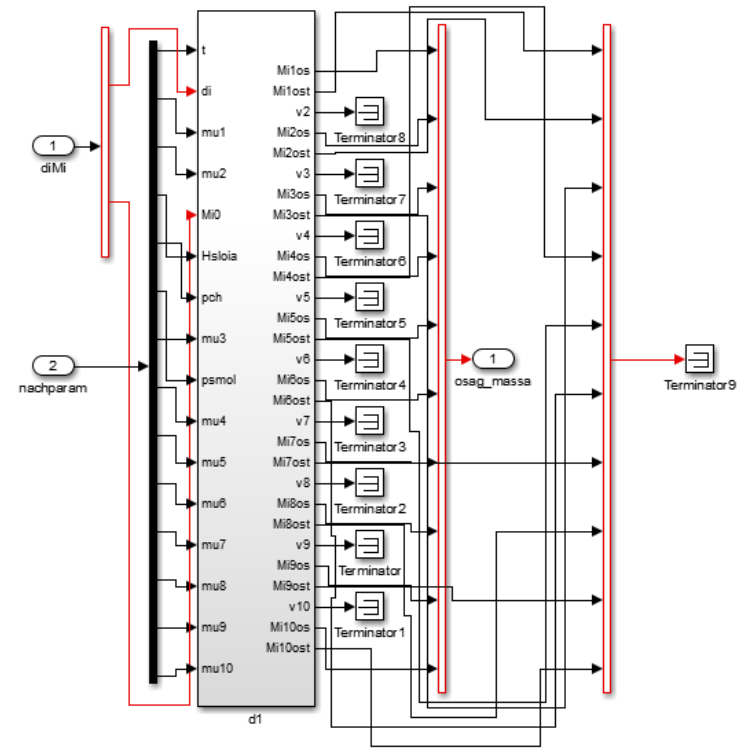


Рисунок 39 – Соединение подсистемы «d1»

Объединим, получившуюся структуру в ещё одну подсистемы, назовем её также «d1». Поскольку в нашей смеси представлены 17 фракций глины, а каждая подсистема «d1» моделирует осаждение одной фракции, нам понадобится 17 таких подсистем, причем выходы подсистем «d» необходимо суммировать, для чего на схеме необходимо поместить сумматор с 17 входами. Фрагмент схемы подключения элементов подсистем «d» представлен на рисунке 40.

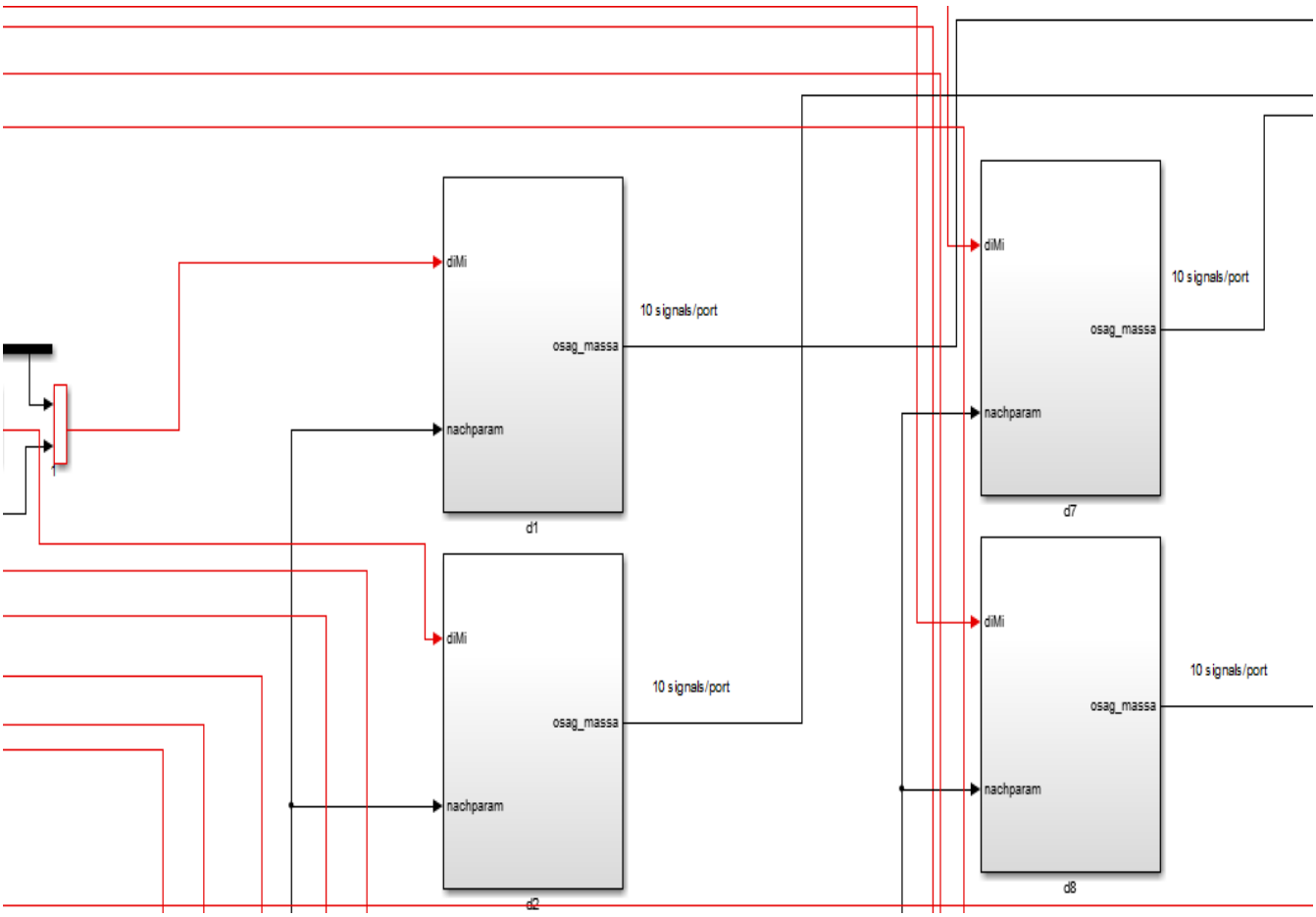


Рисунок 40 – Фрагмент подключения подсистем «d»

Порты «dMi» предназначены для передачи связки параметров «di» и «mi» по шине, порты «nachparam» – для передачи исходных данных, например вязкости, плотностей, времени моделирования.

Блоки формирования шин представлены на рисунках (41)-(42).

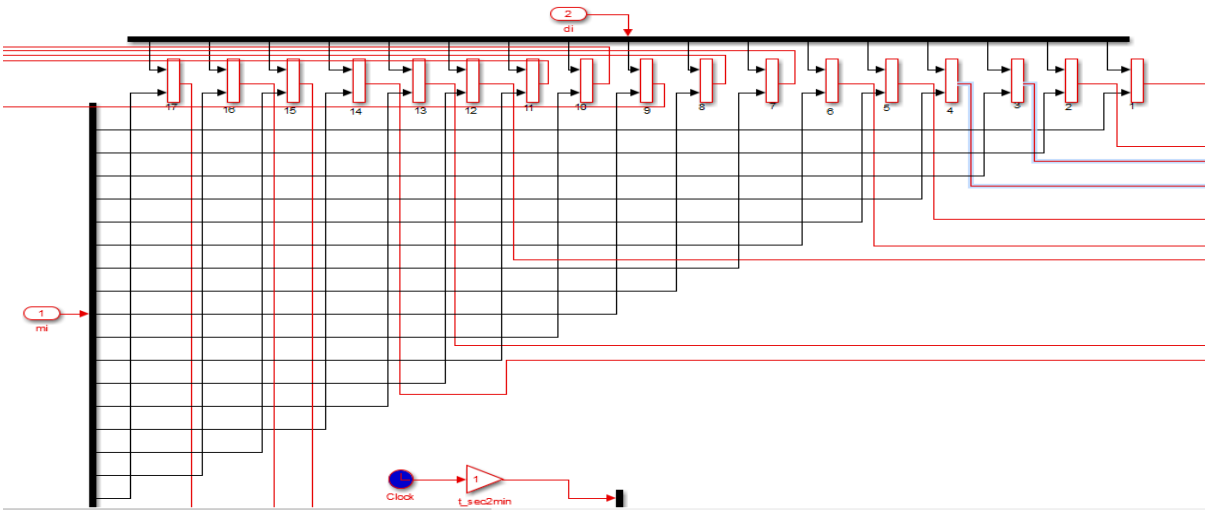


Рисунок 41 – Фрагмент участка формирования шин для портов «dMi»

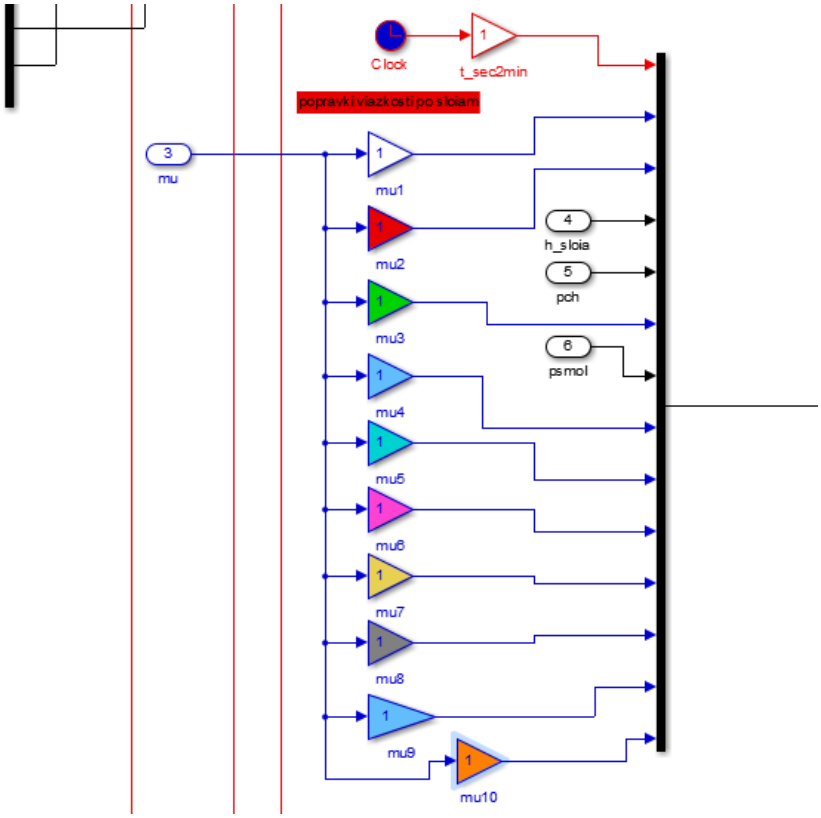


Рисунок 42 – Фрагмент участка формирования шин для портов «nachparam»

Объединим получившиеся соединения в подсистему, назовем её «osagdenie» (рисунок 43).

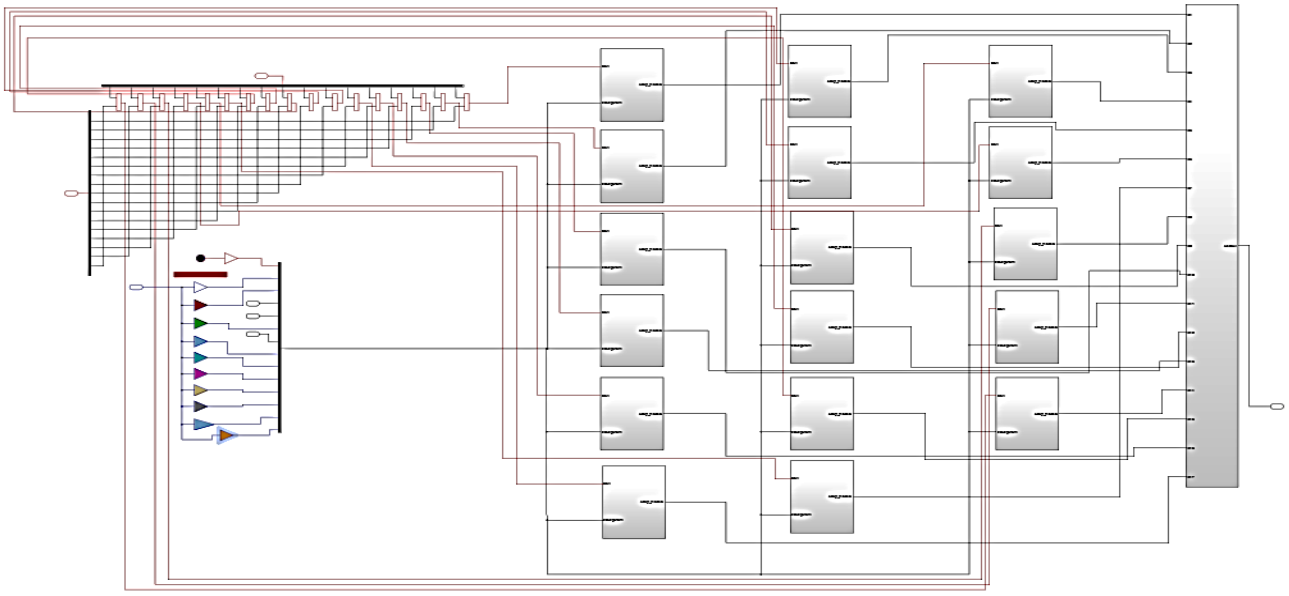


Рисунок 43 – Подсистема «osagdenie»

Сформируем блоки задания исходных данных и перевода величин с систему СИ, а также создадим средства вывода результатов моделирования в графической форме (рисунок 44).

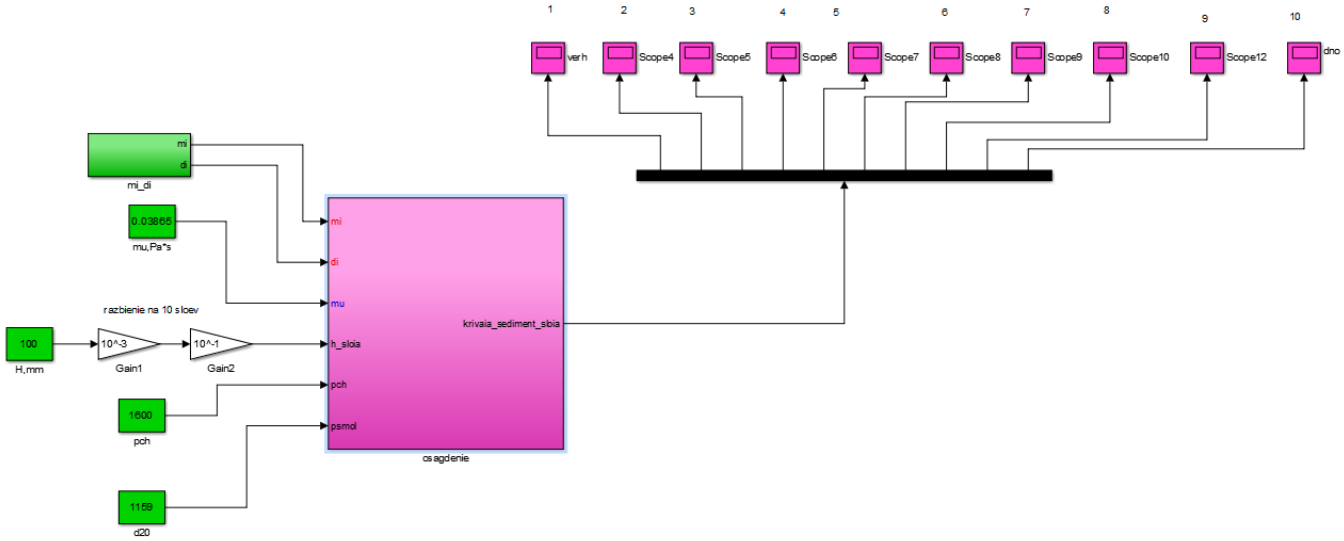


Рисунок 44 – Итоговая схема модели

Подсистема «mi_di» отвечает за формирования шин исходных данных, представленных в таблице 6 и представлена на рисунке 45.

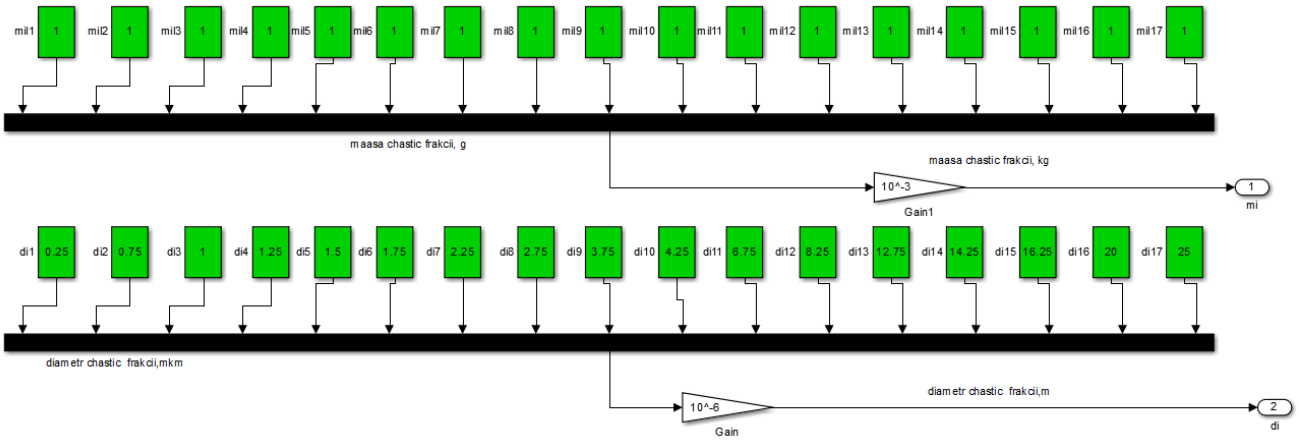


Рисунок 45 – Подсистема «mi_di» формирования исходных данных

Каждый из десяти осциллографов (рисунок 44) показывает кривую осаждения для случая, когда чашка весов седиментометра опущена на соответствующую глубину слоя (1- слой 10 мм, 2- слой 20 мм и т.д.).

Результат моделирования для случая, когда чашка весов установлена на глубине 40 мм от поверхности раздела фаз и имеет диаметр, равный диаметру цилиндра, в котором происходит осаждение, представлен на рисунке 46.

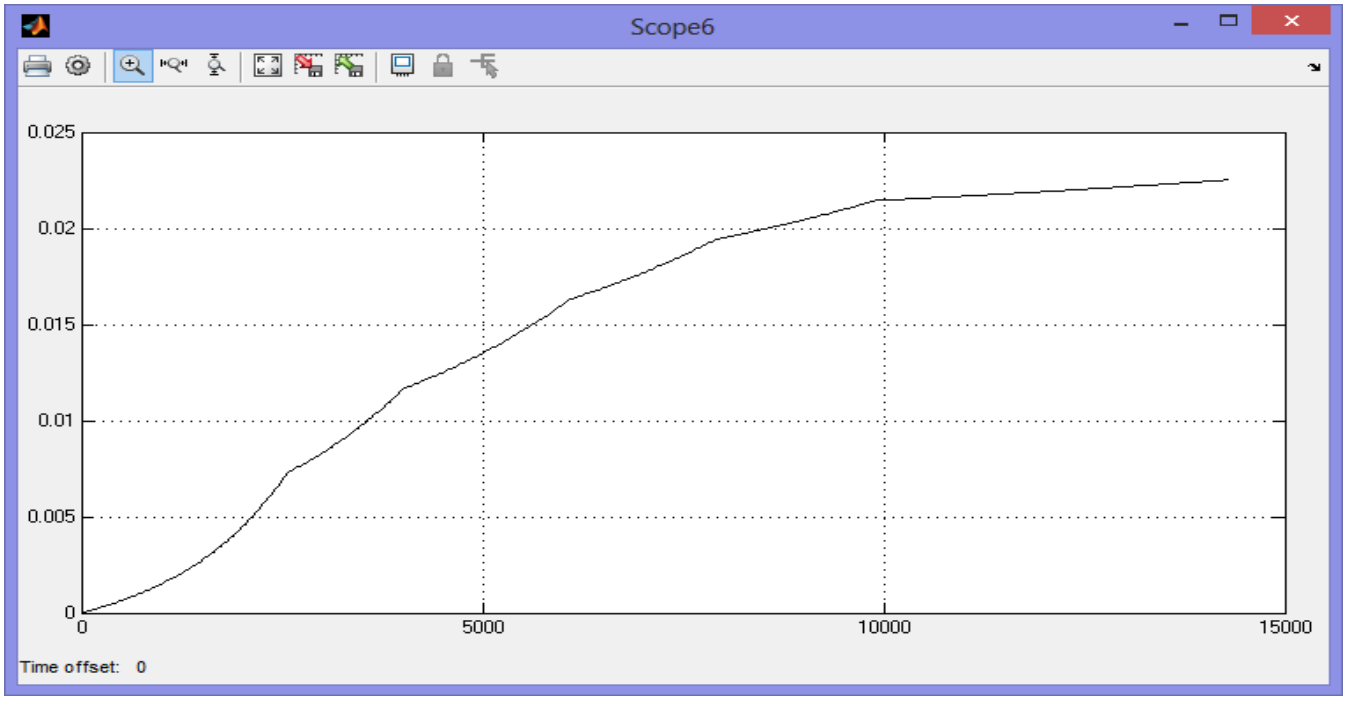


Рисунок 46 – Кривая осаждения, слой 40 мм

3.7 Пример решения тепловой задачи в Matlab

В расчетах температурного состояния заготовки часто требуется определить момент ее готовности или максимальный градиент температур по сечению в процессе термообработки, рекомендовать режим термообработки с учётом допустимых температурных напряжений, от точности такого расчета зависит качество термообработки, угар металла, расход топлива на нагрев, производительность агрегата. Аналитические расчеты провести не всегда удается, а тем более просчитать несколько вариантов задача довольно трудоемкая, связанная с большим объемом вычислений. В таких случаях математическая модель процесса позволяет существенно сократить трудозатраты и решить задачу.

Решение задачи определения температурного и теплового состояния заготовки методом конечных разностей не ново, но в данной методике предложен новый, оригинальный алгоритм позволяющий, путем использования стандартных блоков объектно-ориентированного языка программирования Matlab, описывающих тот или иной динамический процесс, представить задачу теплообмена в виде структурной схемы отражающей процессы теплообмена в пространстве и времени. Методика представляется более привлекательной потому, что не требует глубоких специальных знаний в области программирования, но очень наглядно представляет динамику происходящих при нагреве процессов с высокой точностью и, вполне годиться, для инженерного использования при решении указанных выше производственных задач.

Рассмотрение способов построения и основных свойств разностных схем начнем с задачи теплопроводности, возникающей при расчете симметричного нагрева бесконечной пластины толщиной 2 d (дельта). В этом случае в каждый момент времени изменение температуры в пространстве T(y, t) происходит лишь в направлении оси y, перпендикулярной поверхности пластины. Используя свойство симметрии температурного поля, поместим начало координат y = 0 в точку, лежащую в средней плоскости пластины, и выберем в качестве расчетной области G интервал

$0 < y < d$, соответствующий половине толщины пластины. Тогда уравнение теплопроводности, описывающее нагрев пластины, принимает вид:

$$c\frac{dT}{dt} = \frac{d}{dy}\left(\lambda\frac{dT_0}{dy}\right), \tag{111}$$

Предполагая, что в начальный момент времени тело является равномерно прогретым до температуры $T_н$, запишем начальное условие:

$$T(y,0) = T_н \quad 0 < y < \delta \tag{112}$$

Граничное условие при y=0 является следствием симметрии температурного поля. Следовательно:

$$\frac{dT}{dy} = 0 \quad y = 0 \tag{113}$$

На поверхности пластины будем считать заданным линейное граничное условие 3 рода, соответствующее постоянной температуре окружающей среды T_0 и постоянному, не зависящему от температуры, коэффициенту теплоотдачи α (альфа) [Вт/(м^2 ·K)].

$$\lambda\frac{dT}{dy} = \alpha \bullet (T_0 - T) \quad y = \delta \tag{114}$$

При записи двух последних соотношений учтено, что при нагреве тела внешний тепловой поток имеет направление противоположное оси y.

Рассмотрим сначала применение метода конечных разностей для решения линейной задачи теплопроводности, предполагая, что теплофизические характеристики тела c и λ не зависят от температуры. В этом случае уравнение упрощается и принимает следующий вид:

$$\frac{dT}{dt} = a\frac{d^2T}{dy^2},\ 0 < y < \delta, \qquad (115)$$

где $a = \lambda / c$ коэффициент температуропроводности $м^2/с$.

3.7.1 Построение разностных схем

Основная идея метода конечных разностей (метода сеток) заключается в том, что непрерывная область изменения пространственной переменной $0 < y < \delta$ заменяется конечной совокупностью дискретно расположенных узловых точек $Y_1, Y_2, ..., Y_n, Y_{n+1},$. При равномерном расположении этих точек на отрезке [0,d] их координаты равны $Y_i=(i-1)V_y$ при i =1, ..., n, n+1, где расстояние между соседними точками (шаг по координате) $\Delta y= d/n$. Аналогично, вместо непрерывного изменения температурного поля во времени рассматриваются значения температуры в фиксированные моменты времени $t_k=k\ \Delta t$, k=1, 2, ..., где Δt - интервал между двумя последовательными моментами времени (шаг по времени). В плоскости (y, t) совокупность узловых точек с координатами (y_i,t_k) образует прямоугольную сетку, изображенную на рисунке 47 ниже, и расчет температурного поля T(y,t) сводится к отысканию сеточной функции T_i^k, приближенно характеризующей температуру тела в узловых точках.

При замене непрерывной функции T(y,t) дискретной сеточной функцией T_i^k необходимо заменить дифференциальное уравнение теплопроводности с соответствующими краевыми условиями, системой алгебраических (разностных) уравнений, связывающих значения сеточной функции в соседних узловых точках. Такая система алгебраических уравнений, являющаяся приближенной математической моделью процесса теплопроводности, называется разностной схемой решения исходной краевой задачи.

Перейдем к построению разностных схем для линеиной одномерной задачи теплопроводности. Используем для этого метод баланса, причем будем исходить не из готовых дифференциальных соотношений, а непосредственно из законов

сохранения энергии и переноса тепла, примененных к дискретному температурному полю.

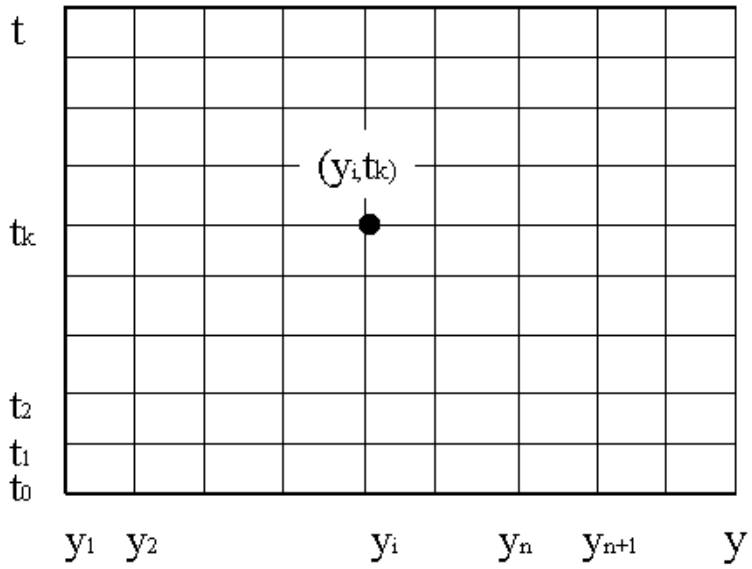


Рисунок 47 – Прямоугольная сетка в плоскости (y, t)

Рассмотрим элементарный слой $y_{i-1}/2 < y < y_{i+1}/2$ толщиной Δy, соответствующий некоторому i-тому внутреннему узлу и запишем для него уравнение теплового баланса при переходе от k-того к k+1 -му моменту времени. В расчете на единицу площади поперечного сечения пластины получим:

$$C \cdot \Delta y\left(T_i^{k+1} - T_i^k\right) = \left(q_{i+1}/2 - q_{i-1}/2\right) \cdot \Delta t\,, \qquad (116)$$

где $(q_{i+1}/2)$ -плотность теплового потока, входящего в i -тый слой со стороны соседнего правого узла;

$a(q_{i-1}/2)$ -плотность теплового потока, выходящего из i-того слоя и передаваемого соседнему левому узлу.

Правая часть уравнения выражает количество тепла, аккумулированного i-тым элементарным слоем (i-тым узлом) в течение интервала времени Δt. Левая часть

выражает изменение энтальпии элементарного слоя при изменении его температуры от (T_i^k) и (T_i^{k+1}).Для получения замкнутой системы разностных уравнений относительно сеточных значений температур нужно связать плотности тепловых потоков $(q_{i+1}/2)$ и $(q_{i-1}/2)$ с температурами в соответствующих узловых точках.Для этого используем дискретныи аналог закона Фурье.

$$q_{i+1}/2 = -\lambda \frac{T_{i+1} - T_i}{\Delta y}, q_{i-1}/2 = -\lambda \frac{T_i - T_{i-1}}{\Delta y}. \qquad (117)$$

При конкретизации выражений следует указать, какому моменту времени соответствуют температуры $T_{i+1,} T_i$ и T_{i-1}. Возможность различных ответов на этот вопрос и является главной причиной многообразия разностных схем решения исходной задачи теплопроводности.

Если плотности тепловых потоков вычисляются по температурам в предыдущий, k-тый момент времени, т.е.

$$q_{i+1}/2 = -\lambda \frac{T_{i+1}^k - T_i^k}{\Delta y}, q_{i-1}/2 = -\lambda \frac{T_i^k - T_{i-1}^k}{\Delta y} \qquad (118)$$

то в результате получается так называемая явная разностная схема.

Если в выражениях фигурируют температуры в последующий k+1 -ый момент времени, т.е.

$$q_{i+1}/2 = -\lambda \frac{T_{i+1}^{k+1} - T_i^{k+1}}{\Delta y}, q_{i-1}/2 = -\lambda \frac{T_i^{k+1} - T_{i-1}^{k+1}}{\Delta y} \qquad (119)$$

то получающаяся разностная схема называется чисто неявной.

3.7.2 Явная разностная схема

Подставим выражение (118) в (116) для внутрених узлов и введем обозначение $f = a \cdot \Delta t / \Delta y$, где a-коэффициент температуропроводности. После элементарных преобразований получим систему алгебраических уравнений

$$\frac{T_i^{k+1} - T_i^k}{\Delta t} = a \cdot \frac{T_{i-1}^k - 2T_i^k + T_{i+1}^k}{\Delta y^2}, \ i=2,3 \ \ldots n, \qquad (120)$$

являющуюся разностным аналогом дифференциального уравнения теплопроводности. Из приведенных уравнении следует,что

$$T_i^{k+1} = f \cdot T_{i-1}^k + (1-2f)\, T_i^k + f\, T_i^{k+1}, \ i=2,3 \ \ldots n, \qquad (121)$$

т.е. в каждый k+1 -ый момент времени новые значения температуры T_i^{k+1} определяются тремя ее значениями $T_{i-1}^k, T_i^k, T_{i+1}^k$ в предыдущий, k-тый момент времени. Это положение иллюстрируется шаблоном, изображенным на рис.2, который указывает совокупность точек (i, k), используемых при записи разностных уравнений во внутренних узлах.

Четырехточечный шаблон, соответствующий явной разностной схеме представлен на рисунке 48.

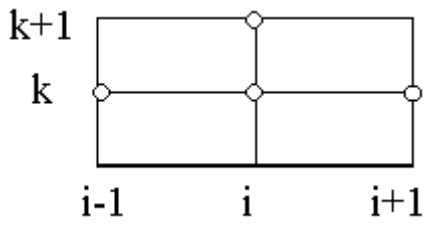


Рисунок 48 – Четырехточечный шаблон явной разностной схемы

Разностное уравнение для левого граничного узла (i=1) соответствующее граничному условию в центре пластины, получим, записав уравнение теплового баланса для крайнего левого элементарного слоя толщиной Δy/2.

$$\frac{1}{2}c\cdot\Delta y\frac{T_1^{k+1}-T_1^k}{\Delta t}=\frac{T_2^k-T_1^k}{\Delta y}, \qquad (122)$$

или

$$T_i^{k+1}=(1-2f)T_1^k+2f\,T_2^k, \qquad (123)$$

при этом, для правого граничного узла (i=n+1) следует записать разностное уравнение, соответствующее одному из граничных условий на поверхности пластины.

При граничном условии II или III рода используем уравнение теплового баланса для крайнего элементарного слоя толщиной Δy/2.

При записи разностного уравнения, учтем, что плотность внешнего теплового потока на k-ом шаге по времени равна $\alpha\cdot\left(T_0-T_{n-1}^k\right)$, получим

$$\frac{1}{2}c\cdot\Delta y\frac{T_{n+1}^{k+1}-T_{n+1}^k}{\Delta t}=\alpha\cdot\left(T_0-T_{n+1}^k\right)-\lambda\frac{T_{n+1}^k-T_n^k}{\Delta y} \qquad (124)$$

или

$$T_{n+1}^{k+1}=2f\cdot T_n^k+[1-2f(1+b)]T_{n+1}^k+2\cdot f\cdot b\cdot T_0, \qquad (125)$$

где $b=\alpha\cdot\Delta y/\lambda$.

Соотношения (121), (123), и (125) показывают, что в граничных узлах так же, как и во внутренних, новые значения температуры полностью определяются ее значениями в предыдущий момент времени.

Таким образом, зная исходное распределение температуры (при k=0), вытекающее из начального условия (112)

$$T_i^0 = T_н,\ i=1,2,...,n+1, \qquad (126)$$

используя приведенные разностные уравнения, можно, последовательно переходя от k-того к k+1 -му моменту времени, произвести расчет дискретного температурного поля.

Отметим, что особенностью полученной явной разностнои схемы является то, что она распадается на отдельные уравнения, решение которых производится независимо друг от друга, причем вычисление новых значений температур в каждый момент времени производится по явным формулам.

Рассмотрим решение примера.

Динамика решения задачи расчета распределения температуры по сечению сляба, нагреваемого в течение 480 с в печи скоростного конвективного нагрева. Половина толщины сляба Δy = 0,08 м, начальная температура $T_н$=1100 К.

Температура газа T_0=2000 К, коэффициэнт теплоотдачи α =350 Вт/(м·К).

Коэффициэнт теплопроводности λ и температуропроводности a для стали соответственно равны: λ=28 Вт/(м^2 К) и a =6,4 ·10 $^{-6}$ м^2 /с.

В данном случае, задача теплопроводности описывается уравнениями (115), (112), (113) и (114).

Применив, для решения рассматриваемой задачи, явную разностную схему составим программу расчета.

Программа предусматривает ввод значений исходных данных и проведение вычислений по явным формулам (121), (123), (125) и вывод на экран дисплея значений температур, соответствующих заданному моменту времени.

Решать на ЭВМ уравнение с частными производными можно только после его преобразования в эквивалентную систему обыкновенных дифференциальных

уравнений по методу конечных разностей. Для этого геометрическую координату нагреваемой пластины (толщину сляба) разбивают на отрезки конечной длины Δy.

Производная (115) аппроксимируется центральными разностями по уравнению

$$\frac{dT_i}{dt} = \frac{a}{\Delta y^2} T_{i+1} - \frac{2a}{\Delta y^2} T_i + \frac{a}{\Delta y^2} T_{i-1} \qquad (127)$$

на всех отрезках координаты y, кроме левого и правого граничных узлов.

Для левого граничного узла в центре пластины производная (115) аппроксимируется разностью по уравнению

$$\frac{dT_i}{dt} = \frac{2a}{\Delta y^2} T_{i-1} - \frac{2a}{\Delta y^2} T_i . \qquad (128)$$

Для правого граничного узла на поверхности пластины производная (115) аппроксимируется разностью по уравнению

$$\frac{dT_i}{dt} = \frac{2\alpha a}{\lambda \Delta y} T_{i-1} - \frac{2\alpha a}{\lambda \Delta y} T_i - \frac{2a}{\Delta y^2} T_i + \frac{2a}{\Delta y^2} T_{i+1}, \qquad (129)$$

где Δy - толщина слоя разбиения пластины,

T_i -температура на i-том участке разбиения пластины.

Для выбраного примера выбираем сетку из 13 узловых точек,тогда для12 -ти участков одинаковой толщины вместо исходного уравнения с частными производными запишем эквивалентную схему из 13 -ти дифференциальных уравнений первого порядка:

$$1. \frac{dT_1}{dt} = \frac{2\alpha a}{\lambda \Delta y} T_0 - \frac{2\alpha a}{\lambda \Delta y} T_1 - \frac{2a}{\Delta y^2} T_1 + \frac{2a}{\Delta y^2} T_2$$

$$2.\ \frac{dT_2}{dt} = \frac{a}{\Delta y^2}T_3 - \frac{2a}{\Delta y^2}T_2 + \frac{a}{\Delta y^2}T_1$$

$$3.\ \frac{dT_3}{dt} = \frac{a}{\Delta y^2}T_4 - \frac{2a}{\Delta y^2}T_3 + \frac{a}{\Delta y^2}T_2$$

4.....

5. …….

6. …….

…….

$$12.\ \frac{dT_{12}}{dt} = \frac{a}{\Delta y^2}T_{13} - \frac{2a}{\Delta y^2}T_{12} + \frac{a}{\Delta y^2}T_{11}$$

$$13.\ \frac{dT_{13}}{dt} = \frac{2a}{\Delta y^2}T_{12} - \frac{2a}{\Delta y^2}T_{13}$$

Начальные условия системы, вводятся на соответствующие интеграторы, с помощью которых решаются указанные выше дифференциальные уравнения. На выходе каждого интегратора, представляющего узловую точку сеточной области модели нагрева (или охлаждения) в результате расчёта будем иметь динамическую кривую температурного состояния заготовки в данной точке.

Перед началом решения системы уравнений целесообразно произвести замену перменной t, исходя из равенства $t=\theta/A1$. При этом, значительная часть коэффициентов в уравнениях становится равной единице. Этот прием фактически означает введение масштаба времени, численно равного $1/A1$, однако замена независимой переменной t в исходных уравнениях позволяет изменить коэффициенты передачи на сумматорах или параметры на интеграторах упрощая математическую модель процесса.

Решение этих уравнений можно реализовать используя всего три- пять типовых блоков имеющихся в инструментальном средстве визуального моделирования Simulink, входящего в состав популярного математического пакета Matlab, который

широко используется студентами ВУЗов, обучающихся по технической специальности. С его помощью можно легко реализовать решение дифференциальных уравнений разностной схемы.

При построении модели будем использовать типовые блоки стандартной библиотеки компонентов программы (рисунок 49).

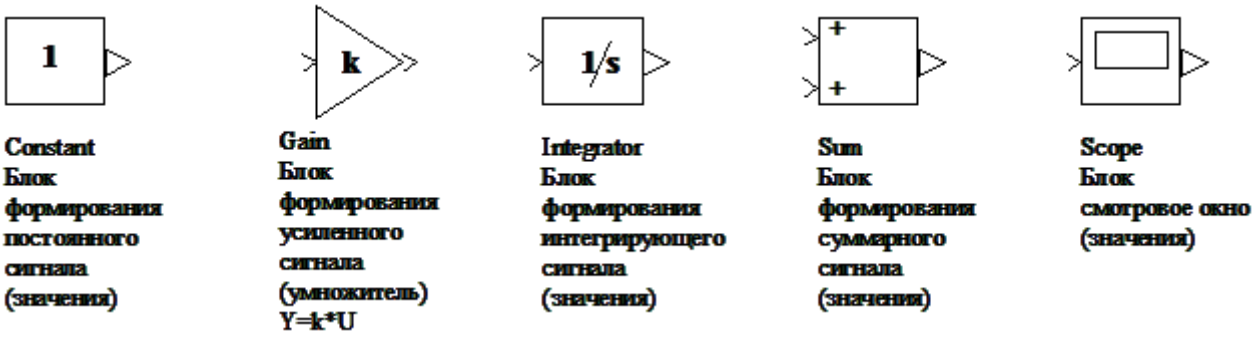


Рисунок 49 – Используемые блоки для построения разностной схемы

Структурная схема,для решения системы уравнений представлена ниже на рисунке 50.

С помощью блока **Constant** будем задавать температуру окружающей среды и температуру заготовки в начале нагрева.

С помощью блока **Gain** будем задавать условия теплообмена между окружающей средой и заготовкой, а также условия теплообмена внутри заготовки между точками сеточной области.

Продолжительность моделирования задаётся установкой параметров на панели меню SIMULINK, там же осуществляется выбор одного из методов интегрирования. Подачей расчетного сигнала T1, T2 и т.д. на вход блока **Scope** обеспечивается получение графического представления динамики нагрева заготовки.

Данная схема может быть реализована для расчета нагрева заготовок в термических печах, слитков в нагревательных колодцах, методических печах, необходимо, только пересчитать коэффициенты и увеличить или уменьшить число интегрирующих блоков, задать начальные и граничные условия. При задании шага по времени и

общей продолжительности нагрева, необходимо учитывать масштаб времени получающийся при расчёте $\theta = A1 \cdot t$. В нашем примере $\theta = 0.144 \cdot 480 = 69.12$.

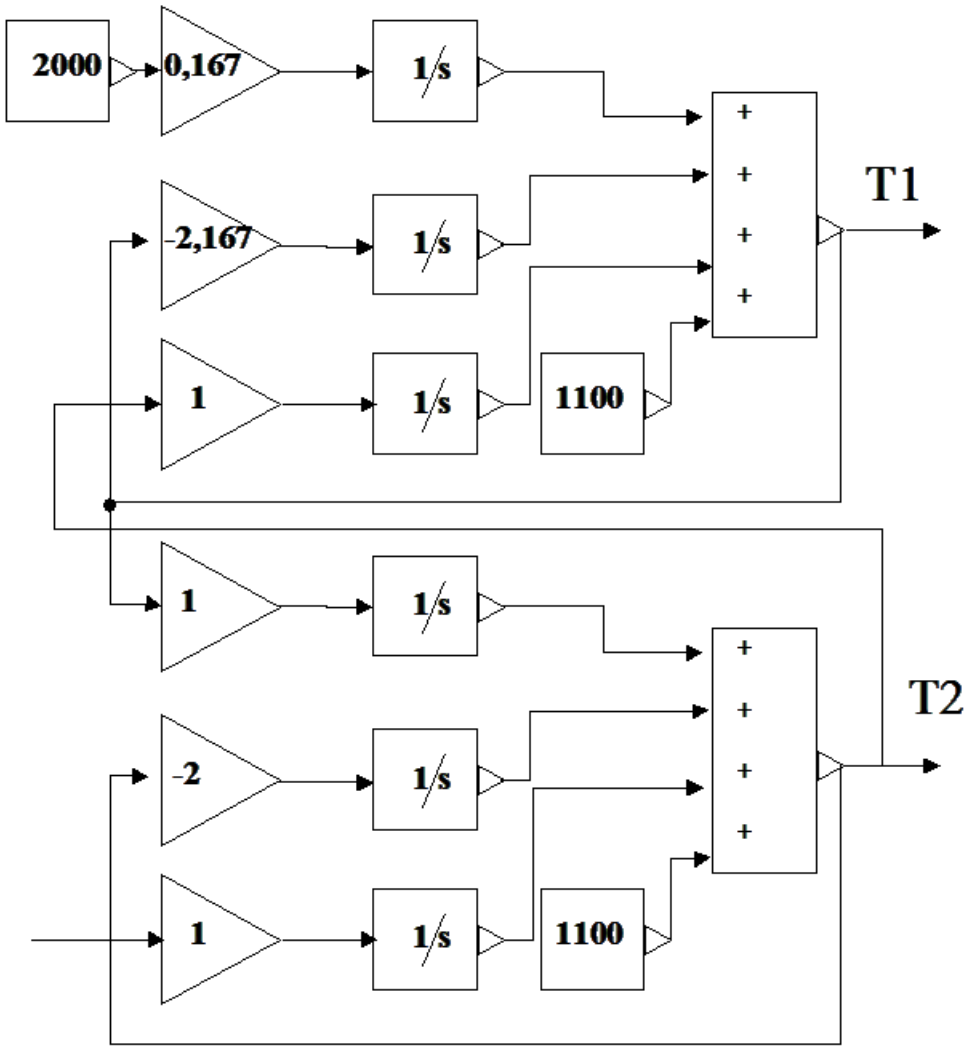


Рисунок 50 – Участок блок-схемы для решения системы дифференциальных уравнений

В случае использования громоздких схем целесообразно объединять элементы и участки схемы со сходными функциями в блоки-подсистемы, сворачивая схему до компактных размеров. Для этого выделяют соответствующие элементы схемы и в контекстном меню (вызываемом по правому щелчку мыши) выбирают пункт «Create subsystem». Кроме того, разобранная модель позволяет рассчитывать не только

нагрев заготовки, но и охлаждение. В схеме, при необходимости можно предусмотреть переключение режимов нагрева и охлаждения образца. необходимо отметить, что данная модель также идеализирована, поскольку при её построении было сделано предположение о том, что длина и ширина заготовки много больше её толщины. Для большей адекватности модели при наличии возможности сравнения расчетных и экспериментальных данным вводят поправочные коэффициенты. Кроме того, в процессе моделирования можно также использовать как приведенное время, так и реальное.

Отметим, что графическое средство scope не обладает гибкостью настроек, потому во многих случаях удобнее экспортировать расчетные значения в массив-переменную и воспользоваться командой plot(x, y).

Ниже показан пример решения тепловой задачи при охлаждении и нагреве заготовки при её разбиении на 21 слой, причем расчет производится сначала для охлаждения заготовки в течении 9,17 часа, а затем заготовка охлаждается при более низких температурах ещё 0,75 часа, а после её подают в зону больших температур.

Этапы построения, а также схемы всех подсистем показаны на рисунках (51) – (58).

Отметим, что звено Scope отрисовывает графический ход изменения температур в приведенном времени. При необходимости представления данных в трехмерном виде в координатах (время, расстояние слоя от поверхности, температура) необходимо использовать возможности программирования в Simulink по событию (в данном случае целесообразно назначить выполнение кода, как событие, происходящее по окончанию симуляции). Для наглядности временную координату приведем в масштабах реального времени, в часах от начала процесса теплопередачи.

Изменение температур во времени будем сохранять в переменной Ti при помощи блока «Signal To Workspace», подключенному к шине вывода результатов.

Рассчитанное реальное время также будем экспортировать в переменную treal в созданной нами подсистеме «realtime» используя вышеуказанный блок «Signal To Workspace».

Для вывода трехмерного графического представления результата выполним следующие операции: в окне Simulink откроем меню «Tools – Model Explorer». В правой части открывшегося окна инспектора моделей перейдем на вкладку «Callbacks», в поле «Model callbacks» выберем событие «StopFcn».

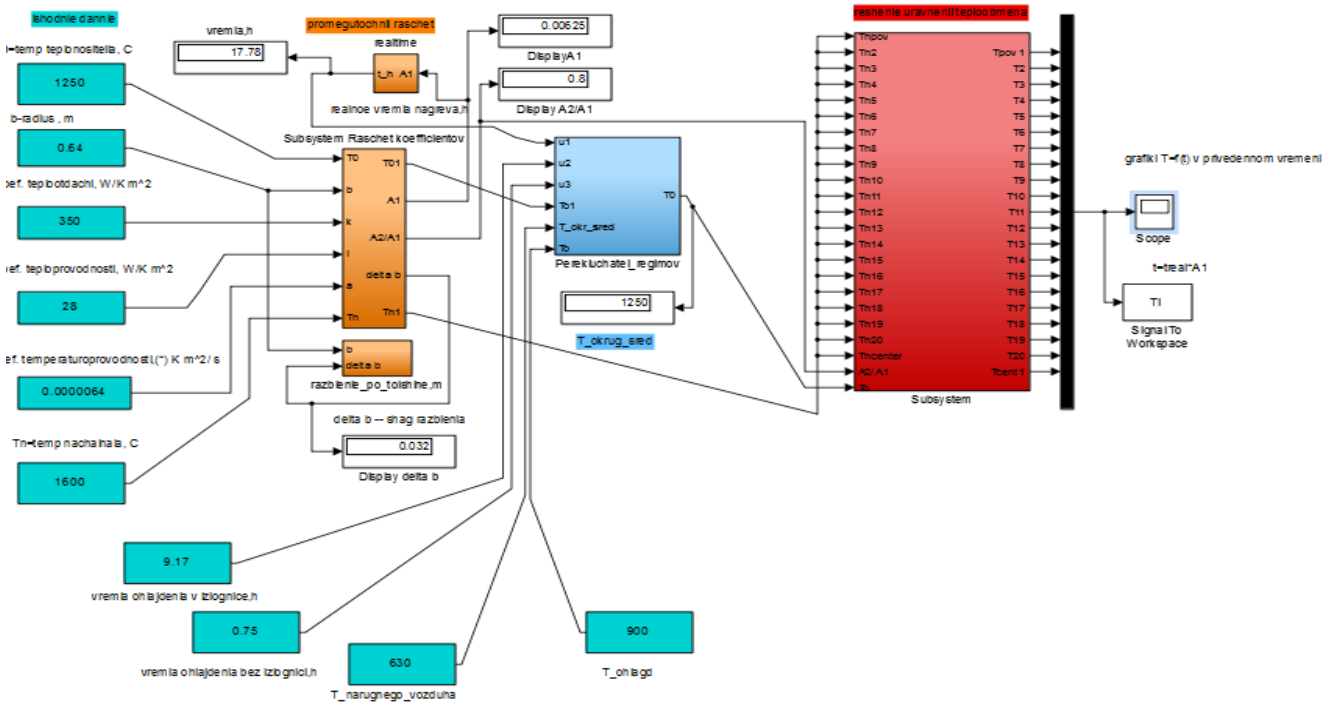


Рисунок 51 – Структурная схема модели

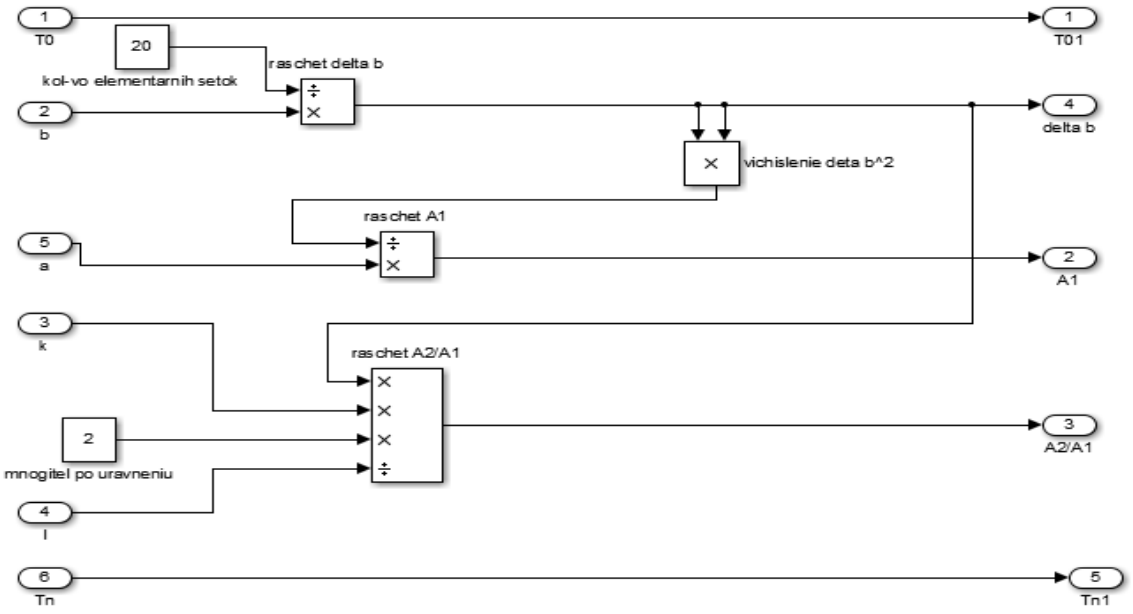


Рисунок 52 – Схема подсистемы расчета коэффициентов и параметров уравнения

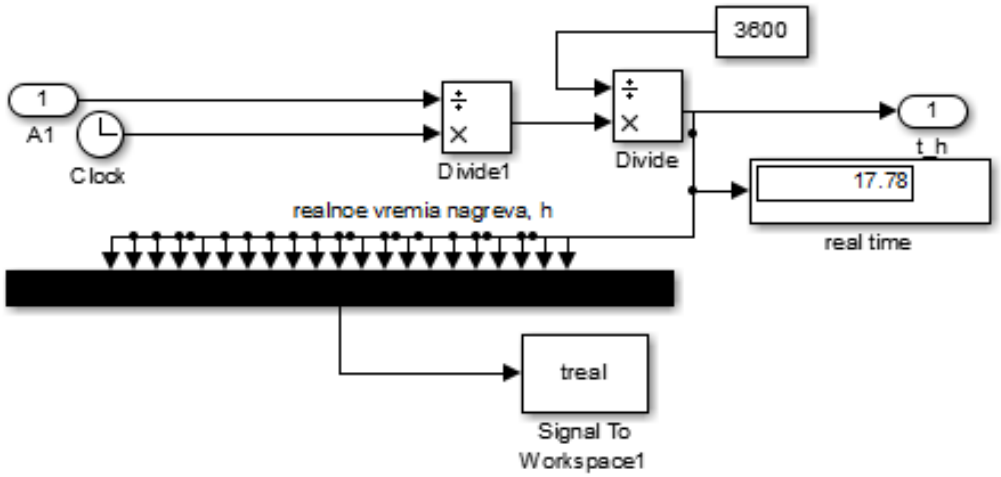


Рисунок 53 – Схема подсистемы расчета реального времени (в часах от начала процесса)

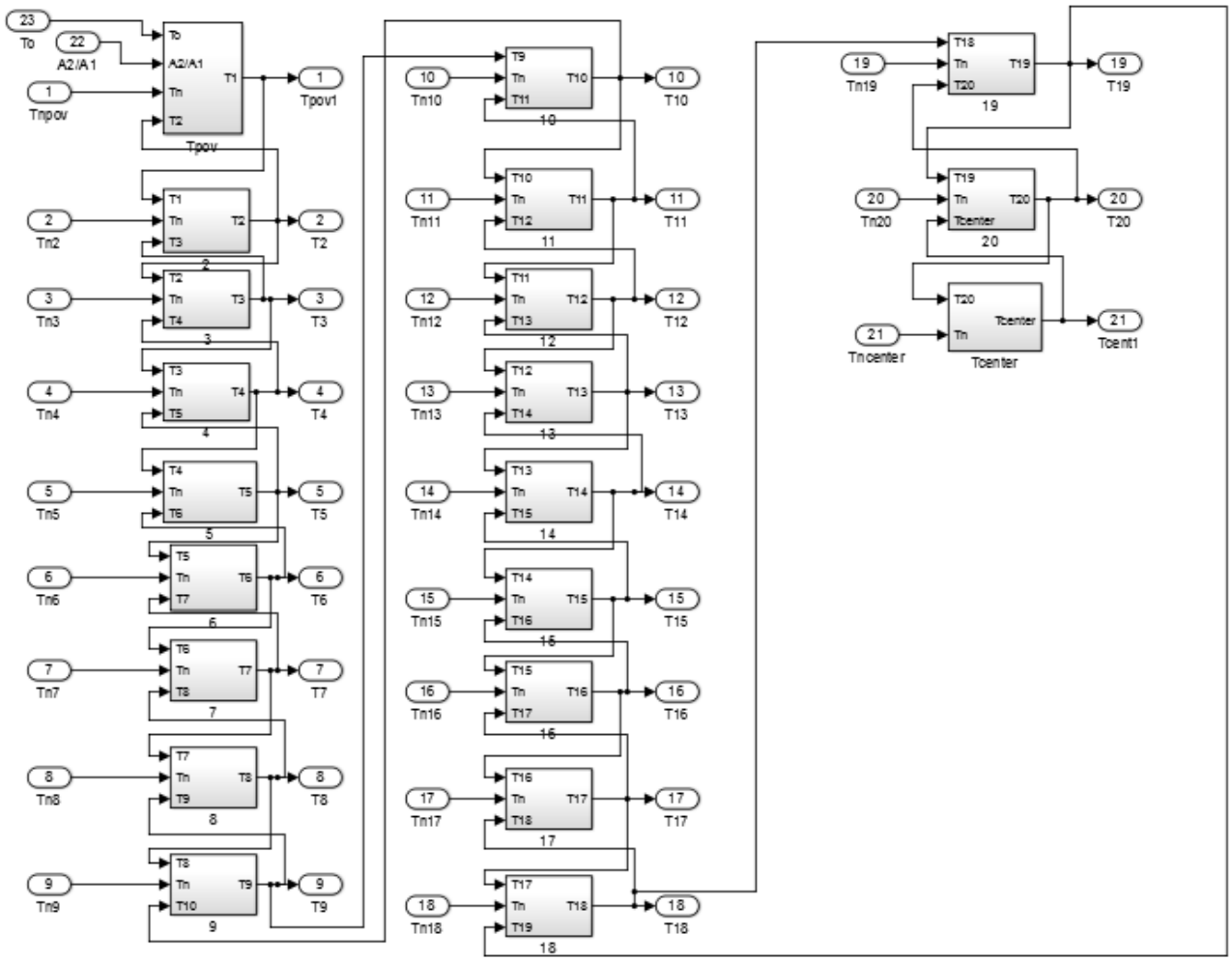


Рисунок 54 – Схема подсистемы расчета разностных уравнений при разбиении полутолщины заготовки на 21 слой

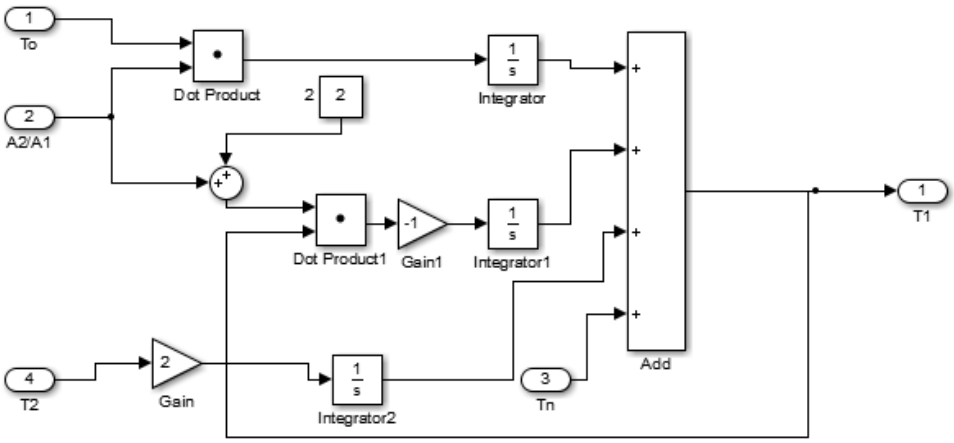


Рисунок 55 – Схема подсистемы расчета разностного уравнения для внешнего слоя

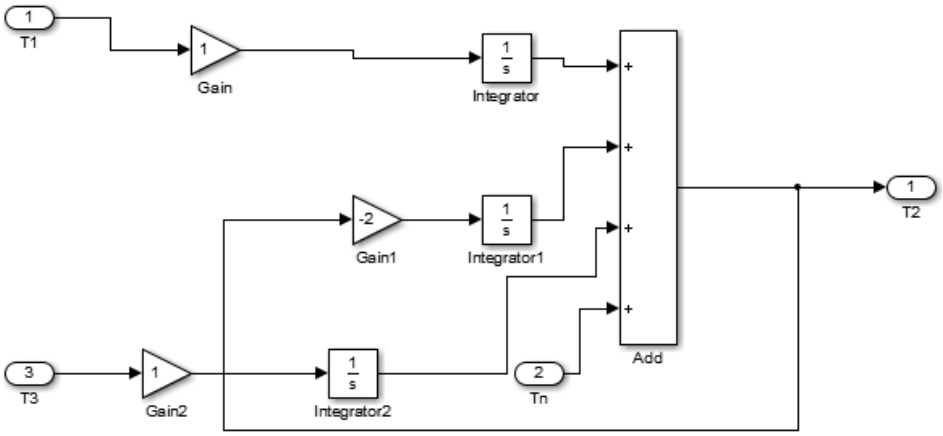


Рисунок 56 – Схема подсистемы расчета разностного уравнения для промежуточного слоя (в данном случае: второго)

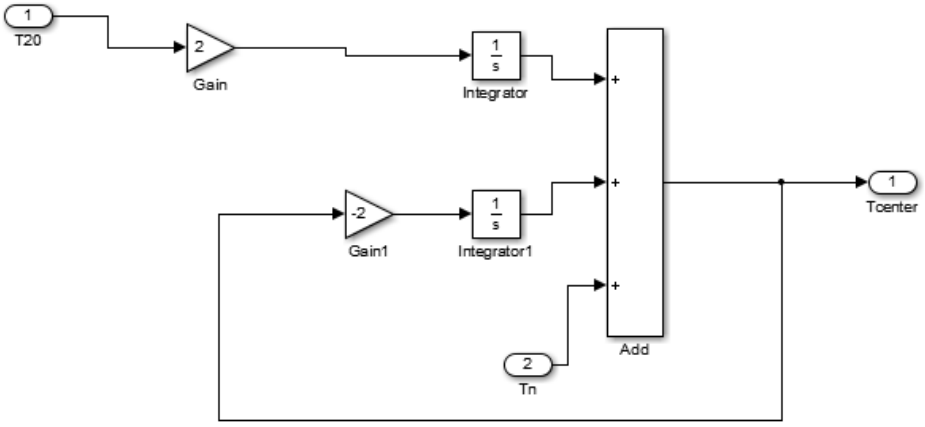


Рисунок 57 – Схема подсистемы расчета разностного уравнения для последнего, внутреннего слоя

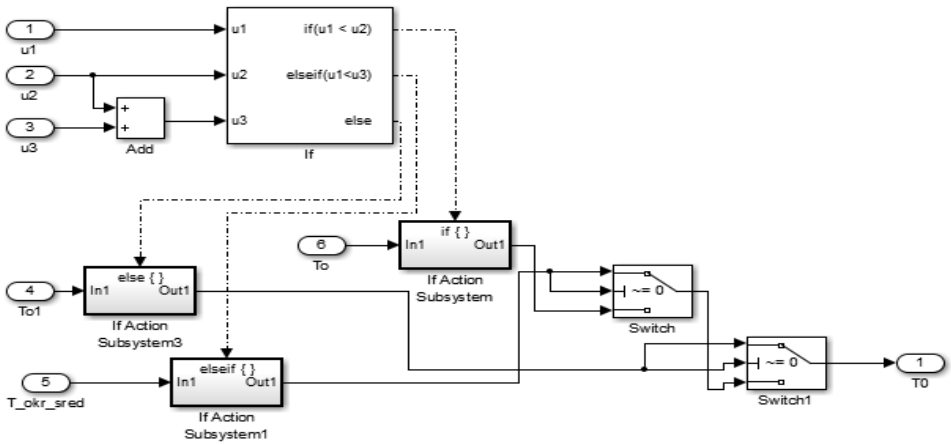


Рисунок 58 – Схема подсистемы переключения режимов охлаждения-нагрева заготовки по заданной продолжительности

Справа от поля события «StopFcn» в поле «Simulation stop function:» введем следующий код:

```
figure(1);
plot3(treal,sloi,Ti);
title('nagrev obrazca, T(treal,sloi)')
xlabel('vremia nagreva, h')
ylabel('rasstoianie ot centra, m')
zlabel('temperatura, C')
figure(2);
tri = delaunay(treal,sloi);
trisurf(tri,treal,sloi,Ti)
xlabel('vremia nagreva, h')
ylabel('rasstoianie ot centra, m')
zlabel('temperatura, C')
clear
```

Результат выполнения кода, а также кривые распределения температур по слоям заготовки в модельном времени представлены на рисунках (59), (60) и (61).

Отметим, что в случае использования трехмерного представления данных возможен поворот системы координат на произвольный угол, включение – отключение

отображения координатной сетки, масштабирование кривых, а также редактирование подписей и поясняющих надписей.

Заметим, что использование Simulink даже без знаний команд программирования, используемых в Matlab позволяет получить простое и наглядное отображение результата моделирования в виде двумерных кривых. В случае необходимости отображения зависимостей параметров друг от друга, без учета временного фактора, выходные данные заводят в блок графопостроения «XYGraph».

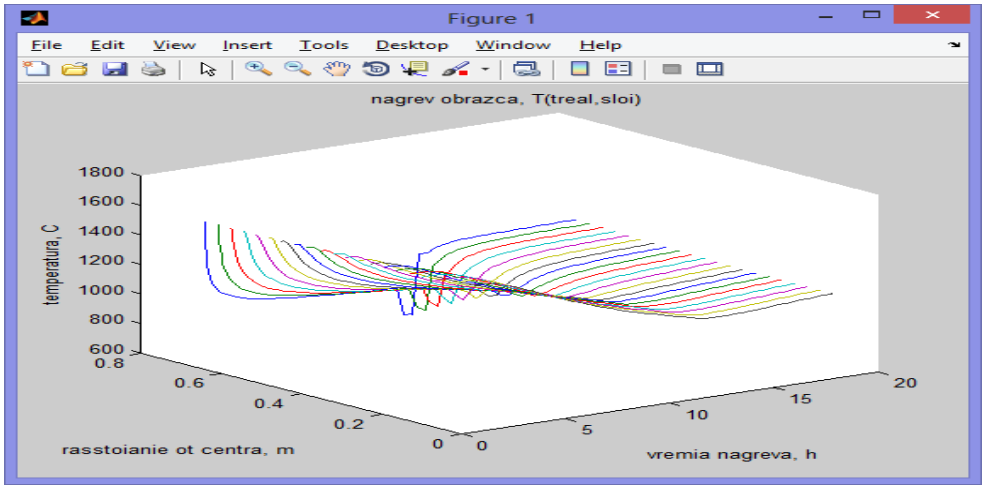


Рисунок 59 – Трехмерный график процесса

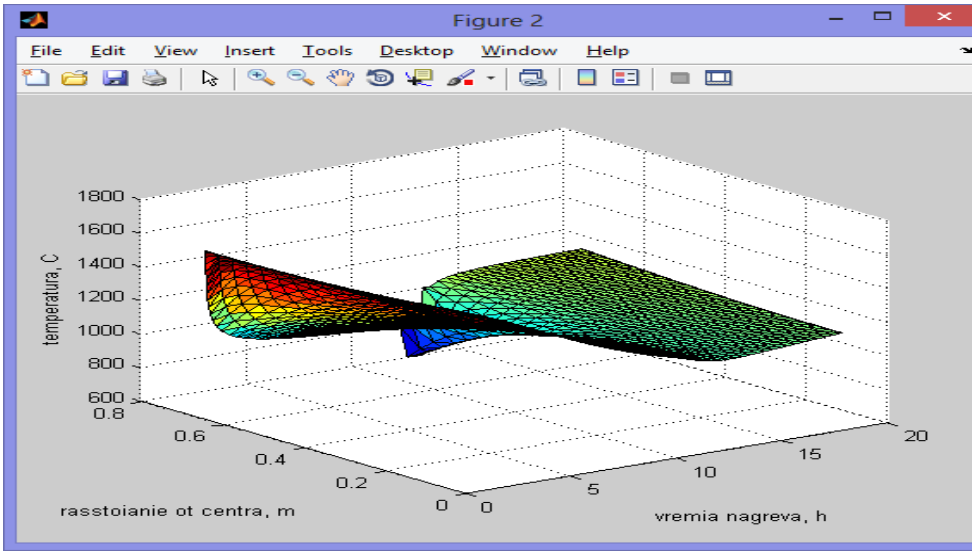


Рисунок 60 – Представление результата в виде трехмерной поверхности

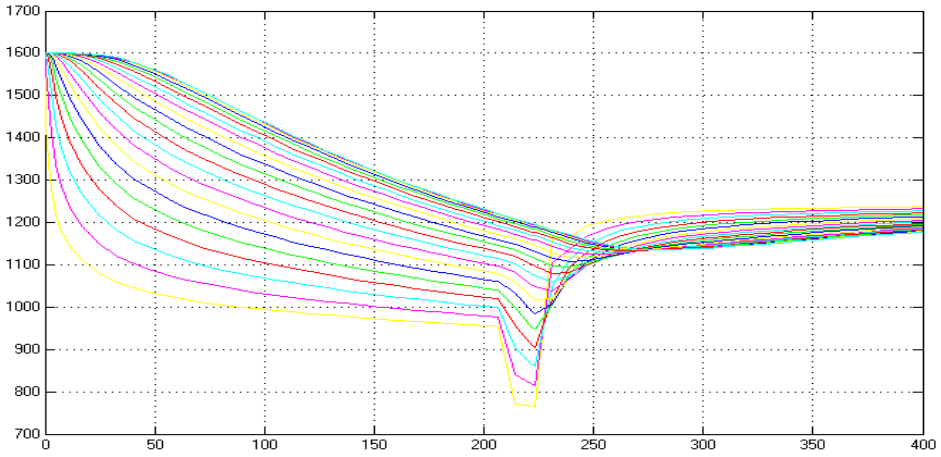


Рисунок 61 – Кривые изменения температур в приведённом времени

Контрольные вопросы и задания к разделу 3

1) Что такое передаточная функция и как зная передаточную функцию объекта получить уравнение переходного процесса?

2) Постройте линеаризованную модель наполнения/опорожнения цилиндрической напорной емкости и сравните её переходную функцию с аналогичной для нелианеризованной модели. Параметры обеих моделей следующие: диаметр бака: 5 м; высота бака: 7 м; расход подачи: 60 м^3/ч.

Жидкость: 98 %- ая серная кислота.

3) Постройте модель осаждения полидисперсной системы показанной в п. 3.6 и получите кривую седиментации частиц для случая, когда чашка седиментометра установлена на глубине 4 см от уровня налива цилиндра. Какая масса частиц осядет к моменту времени 2 часа?

4) Постройте графическое отображение расхода истечения во времени по данным задания 2, если расход подачи изменяется по закону:

а) $Q = 60 + 10\ sin\ (0{,}1t)$, м^3/ч.

б) $Q = 60 + 2\ sin\ (0{,}01t)$, м^3/ч., $t < 2$ ч;

$Q = 60 + 5\ sin\ (0{,}01t)$, м^3/ч, $t \geq 2$ ч.

5) Что такое явная и неявная разностная схема? Как получить и использовать разностные уравнения при моделировании тепловых процессов?

6) Постройте модель охлаждения заготовки по данным, использованным в п 3.7. Смоделируйте ситуацию, когда через два часа после начала эксперимента заготовку помещают в среду с температурой минус 20 °С и до конца эксперимента более не перемещают заготовку.

7) По результатам выполнения предыдущего задания получите массивы значений выходной величины (по слоям) и аппроксимируйте их с помощью нейронных сетей.

8) Какие упрощения использовались при построении разностных уравнений в решении тепловой задачи п.3.7? Для чего используют процедуру линеаризации нелинейных зависимостей? Опишите методы линеаризации динамических нелинейных зависимостей.

9) Для чего необходимо знать характеристики типовых динамических звеньев? Что такое АЧХ и ФЧХ? Можно ли по экспериментальных данным построить АЧХ?

10) Как по ЛАЧХ или АЧХ определить передаточную функцию объекта, а также восстановить уравнение динамики объекта?

11) Для всех примеров, приведенных в п.3.5 - 3.7 дайте объяснение назначения каждого типового блока, использованного при построении модели в Simulink.

14) Перечислите и охарактеризуйте типовые возмущения (воздействия), применяемые при изучении свойств объектов или процессов в химических технологиях.

4 Модели разрушения кускового материала кокса

Как известно, качество кокса оценивают по результатам испытания проб товарного кокса, отбираемых непосредственно перед отправкой потребителю. Это либо перед погрузкой в железнодорожные вагоны или хопперы, либо в месте перегрузки с конвейера коксового цеха на конвейер доменного цеха (при конвейерной подаче к доменным печам). При любом виде транспорта в момент погрузки и далее в процессе транспортировки происходит разрушение кокса и изменение его качественных характеристик. Это дополняется отсевом мелких классов перед подачей в скип доменной печи. Отсев мелочи выполняет самостоятельную роль в преобразовании свойств кокса, не связанную с закономерностями разрушения. Вследствие этого, зависимость между свойствами кокса, которые установлены в момент передачи потребителю и в момент поступления в скип доменной печи, нарушается. Однако на работу доменных печей оказывают влияние свойства кокса, находящегося уже в скипе.

Качество скипового кокса всегда было и остаётся предметом внимания и интереса доменщиков.

Отбор проб скипового кокса производят вручную в сложных условиях, ввиду чего проба, как правило, бывает недостаточно представительна и не отражает средних характеристик кокса даже за смену. В то же время и состав и свойства скипового кокса могут быть определены расчётным путём.

Ежесуточно в доменном цехе, как правило, фиксируется количество «провала» на грохоте перед скипом (так называемые потери от измельчения). Периодически, с целью недопущения потерь с коксовой мелочью доменного кокса класса >25 мм проверяют эффективность грохочения по рассеву подрешетного продукта. Результаты проверок фиксируются в виде процента «провала2 (от использованного кокса) и его состава (так называемые «потери от измельчения»).

Состав скипового кокса представляет собой состав кокса перед грохотом, установленным перед скипом (назовем его «предскиповый») после исключения потерь от измельчения.

Гранулометрический состав предскипового кокса можно вычислить аналитически. Для этого требуется данные о его составе в состояниях: товарной поставки, загрузки в барабан для стандартного испытания и после испытания (после 100 оборотов барабана). Кроме того, необходимо знание так называемых «барабанных эквивалентов» разрушения на тракте коксоподачи. Барабанным эквивалентом называют число оборотов барабана, при котором воспроизводится остаток на определенном сите, равный достигаемому остатку при разрушении кокса равных прочности и состава на каком-либо участке технологического процесса. Численные значения этих эквивалентов определяют по изменению состава товарного кокса в процессе транспортирования до состояния предскипового. При этом используют характеристики прочностных свойств кокса – константы прочности; их находят по изменениям суммарных остатков индивидуально на каждом сите в результате стандартного испытания в барабане.

В зависимости от числа перегрузок с конвейера на конвейер, высоты перепадов и наличия в перегрузочных желобах различных устройств (так называемых «коксовых подушек», отбойных плит и т.д.) соотношение между истирающими и дробящими воздействиями, равно как и величина их будут различными. В случае железнодорожной поставки кокса различия вызываются условиями погрузки кокса в вагоны и разгрузки их. Оказывает влияние и дальность пути перевозки. Однако для каждого предприятия и вида транспортирования суммарные разрушающие воздействия и характер их остаются практически постоянными. Они не зависят от качества кокса.

Разрушающее воздействие на кокс в испытательном барабане отличается от воздействий на него в процессе транспортирования. При идентификации разрушения это учитывается благодаря разной величине барабанных эквивалентов для кокса разных пределов крупности.

Различие в качестве кокса у потребителя обусловлено его свойствами, выявляемыми у поставщика. Оно проявляется в известной мере стандартными показателями качества и константами разрушения кокса, индивидуальными для каждого класса крупности. Установив однажды величины барабанных эквивалентов для предскипового кокса, можно впоследствии по данным стандартного испытания товарного кокса

вычислить состав предскипового, а зная состав и количество «провала» на грохоте перед скипом, расчетным путем найти состав скипового кокса.

4.1 Математическая модель процесса разрушения кокса и принцип использования условных эквивалентов величины механической нагрузки (УЭВМН)

Кокс, получаемый в камерах коксования при нагревании смеси измельченных углей до 1000°С без доступа воздуха, представляет собой после охлаждения массу кусков различных размеров и свойств. При подготовке для использования в металлургических процессах он претерпевает преобразования, связанные с его частичным разрушением и отсевом мелочи. Знание закономерностей этих преобразований позволяет в определенной мере управлять происходящими изменениями, добиваясь улучшения в конечном итоге технико-экономических показателей металлургических производств.

Основные преобразования, происходящие с коксом, связаны с процессом разрушения насыпной массы, деградацией среднего размера кусков и обусловлены механической нагрузкой, которую воспринимает кокс. Естественно, что в первую очередь разрушаются наиболее крупные куски. Они наименее однородны по текстуре и вероятность содержания в них крупных трещин наиболее высока.

Предложив непрерывность разрушающих воздействий, процесс уменьшения в общей массе доли кусков, превышающих некоторый размер i, описывается дифференциальным уравнением (130)

$$- C\frac{[C]_i}{dA} - K_i[C]_i, \qquad (130)$$

где K_i – коэффициент пропорциональности для кусков более размера i.

Отрицательный знак указывает на уменьшение той части насыпной массы кокса, которую мы рассматриваем.

В соответствии с существующей теорией распространения трещин, крупные трещины развиваются интенсивней. По мере их реализации процесс диспергирования замедляется. Предпосылкой снижения скорости давления кусков на части и образования мелочи за счёт их истирания является специфика текстуры кусков кокса, получаемого в коксовых печах. Это связанно с направленным тепловым потоком и различным временем пребывания уже образовавшейся коксовой структуры в зоне высоких температур. В результате этого в кусках проявляется направленная неоднородность по прочности пористого материала.

Изложенное выше говорит о том, что коэффициент пропорциональности в уравнении (130) зависит от уже выполненной работы, т.е.

$$K_i = f(A) \tag{131}$$

Если разрушить кокс в металлическом барабане, то, как показали опыты, изменение прочностных свойств кокса в зависимости от числа оборотов барабана происходит по кривой, аналогичной по форме кривой, описывающей степенную функцию. Это дало основание для предположения зависимости К от А в виде степенной функции, а поскольку значения К уменьшается, показатель степени – число отрицательное.

Изменение К вызвано неоднородностью кусков кокса, поэтому можно записать:

$$K_i = K_{0i} A^{-t0i}, \tag{132}$$

где K_{0i}- коэффициент пропорциональности для идеально однородных кусков;

T_{0i}- показатель, зависящий от уровня неоднородности кокса

Подставив выражение (132) в (130), получим:

$$\frac{-d[C]_i}{dA} - K_{0i}A^{-t0i}[C]_i \quad (133)$$

После разделения переменных, интегрирования, определения и подстановки постоянной, получили:

$$[C_n]i = [C_0] \cdot exp\left(-m_i n^{ti}\right), \quad (134)$$

где m_i и n^{ti}- показатели, характеризующие интенсивность убывания в общей массе кокса количества кусков, %, превышающих размер i, то есть суммарного остатка на сите с размером ячеек i мм.

Установлено, что m_i и n^{ti} постоянны при описании процесса разрушения конкретного кокса и поэтому они названы константами. Группой уравнений типа (134), отличающихся предельным размером кусков рассматриваемой массы (i), может быть определено диспергирование в процессе разрушения. Если разрушение происходит в барабане, то число оборотов барабана n непосредственно связано со временем протекания процесса. Поэтому уравнение (134) может быть названо уравнением, описывающим кинетику процесса разрушения.

Имея результаты разрушения кокса на двух уровнях, n_1 и n_2, можно вычислить константу t_i:

$$t_i = \frac{\frac{lnln[C_0]i}{[C_{n2}]i} - lnln[C_0]i/[C_{n1}]i}{ln(n_2/n_1)}, \quad (135)$$

где $[C_0]i, [C_{n1}]i, [C_{n2}]i$ - суммарный остаток, %, на сите i в коксе исходном, после n_2 и n_1 оборотов барабана, соответственно.

Зная t_i, можно найти m_i по уравнению:

$$m_i = n^{-t_i} \cdot \mathrm{Ln}\frac{[C_0]i}{[C_n]i} \quad (136)$$

Определив значение констант процесса, можно по уравнению (134) вычислить суммарные остатки на ситах на разных ступенях разрушения n. Далее, зная $[C]i$, можно вычислить содержание любого класса крупности в общей массе по соотношениям типа (137) – (139):

$$[C]_{60-80} = [C]_{60} - [C]_{80}, \quad (137)$$

$$[C]_{25-40} = [C]_{25} - [C]_{40}, \quad (138)$$

$$[C]_{0-10} = 100 - [C]_{10} \quad (139)$$

Адекватность приведённых выше уравнений реальному процессу разрушения доказана на большом количестве экспериментального материала. При этом доказано, что уравнения приемлемы для кокса слоевого способа производства различного состава угольной шихты, условий коксования и способа тушения, при разрушении методом наложения некоторого число равных воздействий, независимо от их интенсивности и характера до весьма большой глубины. Например, в малом стандартном барабане, называемом также Микум-барабан при количестве оборотов до 2000.

Обобщающую картину диспергации насыпной массы даёт изменение среднего диаметра кусков. Как известно, диаметр – характеристика тела шаровой формы. Когда используется такой параметр для тел, далёких по форме от шара, применяют вычисление по так называемому определяющему свойству. Это значит, что находят диаметр гипотетического шара, который имеет такое же свойство (поверхность, объём или другое), как у характеризуемого тела. В случае характеристики среднего размера кусков чаше других используется просто среднестатистический размер, вычисляемый по формуле (140):

$$D_{\text{ср}} = \sum \frac{a_i d_i}{100}, \quad (140)$$

где d_i - средний размер класса (например, для класса 60-80 мм это 70 мм);

a_i – количество этого класса в общей массе, %.

Для описания процесса истирания кокса используется гармоническим диаметром, D^r, так как он связан с поверхностью, а истираемость может быть охарактеризована приращением поверхности на единицу работы. D^r вычисляется по формуле:

$$D^r = \frac{100}{\sum ai/d_i} \tag{141}$$

Динамика уменьшения среднего диаметра кокса при разрушении описывается формулой:

$$D_n = D_0 \ exp(-\delta \cdot n^x), \tag{142}$$

где D_n и D_0 - средние диаметры кусков в исходной насыпной массе и после n числа воздействий, соответственно;

δ и x– константы процесса изменения диаметров при разрушении.

При этом для различия зависимости при использовании средневзвешенного и гармонического диаметров в последнем случае вместо δ пишем Δ, и вместо x пишем x', то есть:

$$D_n^г = D_0^г \cdot exp(-\Delta \cdot n^{x'}) \tag{143}$$

Численная характеристика показателя x или x' находится по формуле, аналогичной (135):

$$x \ = \frac{lnln[D_0]i/[D_{n_2}]i - lnln[D_0]i/\ [D_{n_1}]i}{ln(n_2/n_1)}, \tag{144}$$

$$x' = \frac{lnln[D_0^r]i/[D_{n_2}^r]i - lnln[D_0^r]i/[D_{n_1}^r]i}{ln(n_2/n_1)} \quad (145)$$

Аналогично формуле (136) находятся константы Δ иδ.

Например,

$$\delta = n^{-x} \cdot ln\frac{D_0}{D_n}, \quad (146)$$

$$\Delta = n^{-x'} \cdot ln\frac{D_0^r}{D_n^r}, \quad (147)$$

Анализ большого числа экспериментальных данных показал, что для определённого способа разрушения и отмеченной крупности кусков (i), константы t_i изменяются в небольшом диапазоне. Если принять их постоянными величинами, то в некоторой степени это корректируется при вычислении величины m_i, и, следовательно, описание процесса разрушения, до глубины 200-300 оборотов Микум – барабана, сохраняет адекватность. Аналогично может быть упрощено и описание процесса изменения средних диаметров.

Ход процесса разрушения при его реализации в различных условиях представляет интерес во многих случаях, и он неоднократно исследовался прямым экспериментом. Разработка математической модели преобразования состава кокса создала предпосылку для аналитического исследования процесса разрушения кокса, имеющего место в практике его транспортирования потребителям и подготовке к использованию. Понятно, что при перегрузках с транспортёра на транспортёр или при погрузке в железнодорожные вагоны и других операциях кокс разрушается не так, как это имеет место при его испытании. В то же время, зная динамику разрушения кокса в каком-либо испытательном барабане и преобразования насыпной массы при реальном процессе, можно найти такое число оборотов барабана, которое воспроизводит изменение фракций i. Такое число оборотов называют эквивалентным или барабанным эквивалентом, однако правильней называть этот термин «Условная эквивалентная величина механической нагрузки» (УЭВМН).

УЭВМН – называют идентификацией диспергирования. Идентификацией диспергирования можно не только находить по данным испытания в одном аппарате результаты испытания в любом другом (если известны соотношения констант процесса), но и находить прошлое состояние насыпной массы и прогнозировать будущий состав кокса после нахождения величин $n_э$.

Известно, что качество кокса во многом определяет эффективность (в том числе и экономическую) металлургических процессов. Известно также, что по мере разрушения кокса его прочность возрастает, казалось бы – не проблема: надо разрушить кокс до получения максимальной прочности. Но, при разрушении часть кокса переходит из состояния металлургического в коксовую мелочь. Ценность её значительно ниже ценности металлургического кокса. Таким образом, ни металлурги, ни коксохимики не заинтересованы в разрушении кокса, несмотря на то, что более прочный кокс имеет более высокую цену. Типичная картина взаимодействия противоречивых процессов. В таких случаях вопрос решается нахождением оптимума. Иначе говоря, определением такой меры разрушения, при которой переход части металлургического кокса в мелкие фракции будет компенсирован повышением стоимости оставшейся части крупного кокса и повышением эффективности его использования.

4.2 Установление констант разрушения кокса разных классов крупности

Здесь и далее, для простоты и удобства в расчетах будем использовать табличный редактор MS Excel (аналогичные операции можно выполнить и в любом другом табличном редакторе, например в таблицах Google, Open Office и т.д.).

В расчетах исходим из данных, получаемых ОТК предприятия – поставщика ежесменных рассевов из стандартного испытания проб товарного кокса. Исходные данные для расчета представлены в таблице 7. $M_{25} = 84{,}55$, $M_{10} = 8{,}178$.

В этом случае в барабан для испытания (кокс класса >25 мм) будет загружаться соответственно по классам, %:

+80: $15/0{,}973 = 15{,}416$,

60-80: $25{,}9/0{,}973 = 26{,}618$,

40-60: $42{,}5/0{,}973 = 43{,}679$,

25-40: $13{,}9/0{,}973 = 14{,}285$,

где 0,973=(100-2,7)/100 – коэффициент пересчета состава;

2,7 – количество класса <25мм;

Полученные данные представлены в таблице 7 в виде суммарных остатков товарного кокса. Это исходные данные для расчетов. Здесь также представлены и некоторые другие, последовательно, аналитически получаемые данные. Суммарные остатки находят сложением количества классов – больших, чем определенный размер.

Так, остаток на сите 60 мм, % =>80 мм,%+(60-80) мм, %;

Остаток на сите 40=>60+(40-60), или, по – другому, >40=>80+(60-80)+(40-60) и так далее (указываем только одну сторону квадрата).

Ранее показано, что изменение суммарного остатка (%) на сите i при разрушении в барабане любой конструкции до 2000 оборотов описывается уравнением (134):

$$C_{ni} = C_{0i}\exp(-m_i n_i^{ti}),$$

где C_{0i} и C_{ni} – суммарные остатки на сите i (иначе говоря – куски кокса размерами больше, чем i, мм) соответственно в исходном коксе и после n оборотов барабана;

m_i, t_i – константы процесса разрушения для суммарных остатков кокса на ситах i.

Константа t_i связана с однородностью кусков и интенсивностью ее проявления. Известно, что для малого барабана при глубине разрушения порядка 300-400 оборотов значения t_i могут быть приняты постоянными. Их величина приведена в таблице 7.

Имея экспериментально установленные значения C_{0i} и C_{ni}, приняв постоянными значения t_i, для условий стандартного испытания (n=100) можно вычислить константы прочности m для суммарных остатков на ситах i по формуле (136):

$$m_i = \left(100^{ti}\right) \ln\left(\frac{C_{0i}}{C_{ni}}\right)$$

Таблица 7 – Результаты анализа и расчета параметров гранулометрического состава кокса на разных стадиях его транспортировки (Ms Excel)

	A	B	C	D	E	F	G	H
	i	Состав товарного кокса (C_{ti})	>25 мм в товарном коксе (C_0)	Для 100об (C_{100i})	Постоянная деградации (ti)	m_i	Состав предскипового кокса ($C_{пс}$)	Предскиповый после 100 об. ($C_{пс+100об}$)
13	>80мм	15	15,41	0,1	0,4	0,798	3,55	0,091
14	>60мм	40,9	42,03	8,25	0,5	0,162	23,95	7,506
15	>40мм	83,4	85,71	50,65	0,55	0,1041	62,6	45,604
16	>25мм	97,3	100	84,55	0,65	0,008	89,75	80,312
17	>10мм	98,8	100	91,82	0,7	0,003	94,87	89,277

4.3 Определение эквивалентного числа оборотов барабана ($n_{эi}$) для прогнозирования состава предскипового кокса

После логарифмирования уравнения (134) и незначительных преобразований можно увидеть, что:

$$n_{эi} = \sqrt[ti]{\frac{ln(C_{ti}/C_{nci})}{m_i}}, \qquad (148)$$

где C_{ti} и C_{nci} - суммарные остатки на сите i соответственно товарного и предскипового кокса.

Фактически разрушению подвергается кокс всего спектра крупности, включая и класс <25 мм, поэтому расчет ведем исходя из состава товарного кокса ,а не из состава >25 мм, как в случае вычисления констант m_i.

Предположим, что за период отбора проб товарного кокса, состав которого указан ранее (представлен в таблице 7, колонка C_{ti}), отбирали пробу предскипового кокса. Рассев этой пробы до и после 100 оборотов барабана соответствует по фракциям i данным, представленным в таблице 7.

Если занести результаты исследования состава предскипового кокса по фракциям i в колонку G таблицы 7, то для вычисления $n_{э80}$ в произвольной ячейке следует набрать:

$$= ((LN(B13/G13))/F13)\text{^}(1/E13)$$

В ячейке появится значение $n_{э80}$, равное 4,58. Аналогично, копированием формулы рассчитываем $n_э$ для остатков на ситах 60; 40; 25 и 10 мм.

4.4 Расчет гранулометрического состава скипового кокса

По данным предварительных экспериментальных исследований был установлен состав предскипового кокса и состав подрешетного продукта грохота. Допустим, что данные соответствуют представленным в таблице 8 (строки 15,16).

Из процентного количества мелочи <25 мм (замусоренности) предскипового кокса (10%) вычитаем количество этого же класса, ушедшего в отсев (8,28%). Допускаем, что класс >25 мм в отсеве (2,04%) представлен только кусками размером 25-40 мм. Вычитаем из количества класса 25-40 мм в предскиповом коксе (27,15%) количества кокса, перешедшего в «провал» (2,04%). Получаем остаток предскипового кокса, попадающий в скип, таблица 8 строка 17).

Пересчитываем результаты для получения общей суммы 100% (делим % каждого класса на (100-% отсева)/100, т.е на 0,916). Получаем состав скипового кокса (строка 18 таблица 8).

Следует заметить, что количество и состав подрешетного продукта грохочения могут изменятся по двум причинам: разработка (износ) классифицирующих отверстий и изменение количества мелочи в предскиповом коксе. Влияние износа грохотов устанавливают экспериментально, для чего определяют состав подрешетного продукта, как минимум в начале, так и в конце компании работы грохота. Эти данные служат основанием для нахождения в дальнейшем количества кокса >25 мм в отсеваемой мелочи расчетным путем.

Определение потерь от измельчения

Используя вычисленные данные о составе предскипового кокса (таблица 8, строка 20) можно прогнозировать с определенной степенью точности количество потерь от измельчения. Практически это инверсная задача поиска состава скипового кокса. Иными словами, допускаем, что замусоренность скипового кокса и количество кокса >25 мм, переходящего в подрешетный продукт с мелочью, сохраняются постоянными.

Таблица 8 – Изменение качества кокса по стадиям его обработки (Ms Excel)

	B	C	D	E	F	G	H	I	J
	Кокс		Гранулометрический состав (%) по классам, мм						Сум., %
			>80	60-80	40-60	25-40	<25	-10	
15	Предскиповый		3,55	20,4	38,65	27,15	5,12	5,12	100
16	Отсев на грохоте,%		0	0	0	2,04	0	0	10,32
17	Надрешетны кокс		3,55	20,04	38,65	25,11	5,12	5,12	89,67
18	Скиповый,%		3,96	22,80	43,19	28,06	0	0	100
19	Предскиповый +100об		0,091	7,41	38,09	34,70	8,96	10,7	100
20	Отсев до скипа,%		0	0	0	2,04			89
21	Остаток,%		0,091	7,41	38,09	32,66			67
22	Скиповый +100 об.		0,103	8,39	43,12	36,97			100

Вычислив состав предскипового кокса, находим разность между количествами класса <25 мм в предскиповом и скиповом коксе. Получаем количество мелочи в потерях от измельчения (в % от использованного кокса).

4.5 Определение газопроницаемости и плотности насыпной массы кокса

При транспортировке кокса одним из важнейших показателей является насыпная масса кокса. Установлено, что для товарного кокса, в пределах разрушения до 100 оборотов Микум-барабана включительно, зависимость насыпной массы от фракционного состава описывается уравнениями:

$$\rho = 420{,}6 + 171\gamma_{40-60} - 391\gamma_{>60}\gamma_{40-60} - 232\gamma_{40-60}\gamma_{25-40} + \\ +288\gamma_{40-60}\gamma_{<25} + 1047\gamma_{25-40}\gamma_{<25}, \text{ кг/м}^3, \quad (149)$$

где γ – количество кокса класса крупности, указанного в подстрочном индексе, в долях единицы.

Можно вести расчет по формуле:

$$\rho = 420{,}6 + 1{,}71a_{40-60} - 0{,}0391(a_{>80} + a_{80-60})a_{40-60} - 0{,}0232a_{40-60}a_{25-40} + \\ +0{,}0288a_{40-60}a_{<25} + 0{,}105a_{25-40}a_{<25}, \text{ кг/м}^3, \quad (150)$$

где а – количество кокса класса крупности, указанного в подстрочном индексе, в процентах.

Для доменного процесса играет роль газопроницаемость насыпной массы кокса, и это свойство многократно устанавливали экспериментально. Обработка опубликованных данных привела к следующему виду зависимости удельной газопроницаемости от гранулометрического состава:

а) для не разрушавшегося кокса, мм.вод.ст/м:

$$\Delta P_0 = 7{,}6 - 5{,}2\gamma_{>60} - 3{,}0\gamma_{40-60} + 19{,}0\gamma_{<25} + 5{,}1\gamma_{>60}\gamma_{25-40}, \quad (151)$$

где γ – количество кокса класса крупности, указанного в подстрочном индексе, в долях единицы.

Для получения результата в Па/м расчет ведут по формуле:

$$\Delta P_0 = 74{,}5 - 51\gamma_{>60} - 29{,}4\gamma_{40-60} + 186{,}2\gamma_{<25} + 50\gamma_{>60}\gamma_{25-40} \quad (152)$$

б) для кокса после 100 оборотов Микум – барабана применяют:

$$\Delta P_n = 4{,}4 + 145\gamma_{<25} - 274\gamma_{>60}\gamma_{<25} - 82{,}7\gamma_{<25}\,\gamma_{25-40}, \text{ мм.вод.ст/м} \quad (153)$$

Контрольные вопросы и задания к разделу 4

1) Что такое эквивалентная величина механической нагрузки?

2) Какие куски кокса более подвержены разрушению – мелкие или крупные, почему? Как это показать, используя метод УЭВМН?

3) На какие показатели качества кокса влияет его гранулометрический состав?

4) Для чего и как определяют константы деградации?

5) Для чего используют модели разрушения кускового материала, построенные по методу УЭВМН?

6) Что такое эквивалентное число оборотов Микум-барабана и для чего используется эта величина?

7) Что представляет собой Микум-барабан и какие процессы в нем моделируют?

8) Можно ли по известному гранулометрическому составу кокса рассчитать его насыпную массу? Какова область применения такого расчета?

9) Что показывает параметр газопроницаемости, как его рассчитать?

10) Каким образом производится прогноз (расчет) гранулометрического состава скипового кокса и какие данные необходимы для осуществления такого прогноза?

11) Как пересчитать состав валового кокса на товарный кокс?

12) Как выполнить прогноз показателей M_{25} и M_{10} для товарного кокса конечного потребителя по данным рассева и аналогичным показателям рампового кокса?

13) Что такое стабилизация качества кокса и как её осуществить?

14) Как оценить потери от переизмельчения в процессе транспортировки кокса конечному потребителю?

15) Как меняется качество товарного кокса при увеличении длины пути до конечного потребителя и количества транспортных операций загрузки/выгрузки кокса.

5 Методы решения задач оптимизации параметров процессов

При решении конкретной задачи оптимизации исследователь прежде всего должен выбрать математический метод, который приводил бы к конечным результатам с наименьшими затратами на вычисления или же давал возможность получить наибольший объем информации об искомом решении. Выбор того или иного метода в значительной степени определяется постановкой оптимальной задачи, а также используемой математической моделью объекта оптимизации.

В настоящее время для решения оптимальных задач применяют в основном следующие методы:

- методы исследования функций классического анализа;
- методы, основанные на использовании неопределенных множителей Лагранжа;
- вариационное исчисление;
- динамическое программирование;
- принцип максимума;
- линейное программирование;
- нелинейное программирование.

В последнее время разработан и успешно применяется для решения определенного класса задач метод геометрического программирования.

Как правило, нельзя рекомендовать какой-либо один метод, который можно использовать для решения всех без исключения задач, возникающих на практике. Одни методы в этом отношении являются более общими, другие - менее общими. Наконец, целую группу методов (методы исследования функций классического анализа, метод множителей Лагранжа, методы нелинейного программирования) на определенных этапах решения оптимальной задачи можно применять в сочетании с другими методами, например, динамическим программированием или принципом максимума.

Отметим также, что некоторые методы специально разработаны или наилучшим образом подходят для решения оптимальных задач с математическими моделями определенного вида. Так, математический аппарат линейного программирования, специально создан для решения задач с линейными критериями оптимальности и линейными ограничениями на переменные и позволяет решать большинство задач, сформулированных в такой постановке. Так же и геометрическое программирование предназначено для решения оптимальных задач, в которых критерий оптимальности и ограничения представляются специального вида функциями позиномами.

Динамическое программирование хорошо приспособлено для решения задач оптимизации многостадийных процессов, особенно тех, в которых состояние каждой стадии характеризуется относительно небольшим числом переменных состояния. Однако при наличии значительного числа этих переменных, т. е. при высокой размерности каждой стадии, применение метода динамического программирования затруднительно вследствие ограниченных быстродействия и объема памяти вычислительных машин.

Пожалуй, наилучшим путем при выборе метода оптимизации, наиболее пригодного для решения соответствующей задачи, следует признать исследование возможностей и опыта применения различных методов оптимизации. Ниже приводится краткий обзор математических методов решения оптимальных задач и примеры их использования. Здесь же дана лишь краткая характеристика указанных методов и областей их применения, что до некоторой степени может облегчить выбор того или иного метода для решения конкретной оптимальной задачи.

Методы исследования функций классического анализа представляют собой наиболее известные методы решения несложных оптимальных задач, с которыми известны из курса математического анализа. Обычной областью использования данных методов являются задачи с известным аналитическим выражением критерия оптимальности, что позволяет найти не очень сложное, также аналитическое выражение для производных. Полученные приравниванием нулю производных уравнения, опре-

деляющие экстремальные решения оптимальной задачи, крайне редко удается решить аналитическим путем, поэтому, как, правило, применяют вычислительные машины. При этом надо решить систему конечных уравнений, чаще всего нелинейных, для чего приходится использовать численные методы, аналогичные методам нелинейного программирования.

Дополнительные трудности при решении оптимальной задачи методами исследования функций классического анализа возникают вследствие того, что система уравнений, получаемая в результате их применения, обеспечивает лишь необходимые условия оптимальности. Поэтому все решения данной системы (а их может быть и несколько) должны быть проверены на достаточность. В результате такой проверки сначала отбрасывают решения, которые не определяют экстремальные значения критерия оптимальности, а затем среди остающихся экстремальных решений выбирают решение, удовлетворяющее условиям оптимальной задачи, т. е. наибольшему или наименьшему значению критерия оптимальности в зависимости от постановки задачи.

Методы исследования при наличии ограничений на область изменения независимых переменных можно использовать только для отыскания экстремальных значений внутри указанной области. В особенности это относится к задачам с большим числом независимых переменных (практически больше двух), в которых анализ значений критерия оптимальности на границе допустимой области изменения переменных становится весьма сложным.

Метод множителей Лагранжа применяют для решения задач такого же класса сложности, как и при использовании обычных методов исследования функций, но при наличии ограничений типа равенств на независимые переменные. К требованию возможности получения аналитических выражений для производных от критерия оптимальности при этом добавляется аналогичное требование относительно аналитического вида уравнений ограничений.

В основном при использовании метода множителей Лагранжа приходится решать те же задачи, что и без ограничений. Некоторое усложнение в данном случае

возникает лишь от введения дополнительных неопределенных множителей, вследствие чего порядок системы уравнений, решаемой для нахождения экстремумов критерия оптимальности, соответственно повышается на число ограничений. В остальном, процедура поиска решений и проверки их на оптимальность отвечает процедуре решения задач без ограничений.

Множители Лагранжа можно применять для решения задач оптимизации объектов на основе уравнений с частными производными и задач динамической оптимизации. При этом вместо решения системы конечных уравнений для отыскания оптимума необходимо интегрировать систему дифференциальных уравнений.

Следует отметить, что множители Лагранжа используют также в качестве вспомогательного средства и при решении специальными методами задач других классов с ограничениями типа равенств, например, в вариационном исчислении и динамическом программировании. Особенно эффективно применение множителей Лагранжа в методе динамического программирования, где с их помощью иногда удается снизить размерность решаемой задачи.

Методы вариационного исчисления обычно используют для решения задач, в которых критерии оптимальности представляются в виде функционалов и решениями которых служат неизвестные функции. Такие задачи возникают обычно при статической оптимизации процессов с распределенными параметрами или в задачах динамической оптимизации.

Вариационные методы позволяют в этом случае свести решение оптимальной задачи к интегрированию системы дифференциальных ' уравнений Эйлера, каждое из которых является нелинейным дифференциальным уравнением второго порядка с граничными условиями, заданными на обоих концах интервала интегрирования. Число уравнений указанной системы при этом равно числу неизвестных функций, определяемых при решении оптимальной задачи. Каждую функцию находят в результате интегрирования получаемой системы.

Уравнения Эйлера выводятся как необходимые условия экстремума функционала. Поэтому полученные интегрированием системы дифференциальных уравнений функции должны быть проверены на экстремум функционала.

При наличии ограничений типа равенств, имеющих вид функционалов, применяют множители Лагранжа, что дает возможность перейти от условной задачи к безусловной. Наиболее значительные трудности при использовании вариационных методов возникают в случае решения задач с ограничениями типа неравенств.

Заслуживают внимания прямые методы решения задач оптимизации функционалов, обычно позволяющие свести исходную вариационную задачу к задаче нелинейного программирования, решить которую иногда проще, чем краевую задачу для уравнений Эйлера.

Динамическое программирование служит эффективным методом решения задач оптимизации дискретных многостадийных процессов, для которых критерий оптимальности задается как аддитивная функция критериев оптимальности отдельных стадий. Без особых затруднений указанный метод можно распространить и на случай, когда критерий оптимальности задан в другой форме, однако при этом обычно увеличивается размерность отдельных стадий.

По существу, метод динамического программирования представляет собой алгоритм определения оптимальной стратегии управления на всех стадиях процесса. При этом закон управления на каждой стадии находят путем решения частных задач оптимизации последовательно для всех стадий процесса с помощью методов исследования функций классического анализа или методов нелинейного программирования. Результаты решения обычно не могут быть выражены в аналитической форме, а получаются в виде таблиц.

Ограничения на переменные задачи не оказывают влияния на общий алгоритм решения, а учитываются при решении частных задач оптимизации на каждой стадии процесса. При наличии ограничений типа равенств иногда даже удается снизить размерность этих частных задач за счет использования множителей Лагранжа. Примене-

ние метода динамического программирования для оптимизации процессов с распределенными параметрами или в задачах динамической оптимизации приводит к решению дифференциальных уравнений в частных производных. Вместо решения таких уравнений зачастую значительно проще представить непрерывный процесс как дискретный с достаточно большим числом стадий. Подобный прием оправдан особенно в тех случаях, когда имеются ограничения на переменные задачи и прямое решение дифференциальных уравнений осложняется необходимостью учета указанных ограничений.

При решении задач методом динамического программирования, как правило, используют вычислительные машины, обладающие достаточным объемом памяти для хранения промежуточных результатов решения, которые обычно получаются в табличной форме.

Принцип максимума применяют для решения задач оптимизации процессов, описываемых системами дифференциальных уравнений. Достоинством математического аппарата принципа максимума является то, что решение может определяться в виде разрывных функций; это свойственно многим задачам оптимизации, например, задачам оптимального управления объектами, описываемыми линейными дифференциальными уравнениями.

Нахождение оптимального решения при использовании принципа максимума сводится к задаче интегрирования системы дифференциальных уравнений процесса и сопряженной системы для вспомогательных функций при граничных условиях, заданных на обоих концах интервала интегрирования, т. е. к решению краевой задачи. На область изменения переменных могут быть наложены ограничения. Систему дифференциальных уравнений интегрируют, применяя обычные программы на цифровых вычислительных машинах.

Принцип максимума для процессов, описываемых дифференциальными уравнениями, при некоторых предположениях является достаточным условием оптимальности. Поэтому дополнительной проверки на оптимум получаемых решений обычно не требуется.

Для дискретных процессов принцип максимума в той же формулировке, что и для непрерывных, вообще говоря, несправедлив. Однако условия оптимальности, получаемые при его применении для многостадийных процессов, позволяют найти достаточно удобные алгоритмы оптимизации.

Линейное программирование представляет собой математический аппарат, разработанный для решения оптимальных задач с линейными выражениями для критерия оптимальности и линейными ограничениями на область изменения переменных. Такие задачи обычно встречаются при решении вопросов оптимального планирования производства с ограниченным количеством ресурсов, при определении оптимального плана перевозок (транспортные задачи) и т. д.

Для решения большого круга задач линейного программирования имеется практически универсальный алгоритм - симплексный метод, позволяющий за конечное число итераций находить оптимальное решение подавляющего большинства задач. Тип используемых ограничений (равенства или неравенства) не сказывается на возможности применения указанного алгоритма. Дополнительной проверки на оптимальность для получаемых решений не требуется. Как правило, практические задачи линейного программирования отличаются весьма значительным числом независимых переменных. Поэтому для их решения обычно используют вычислительные машины, необходимая мощность которых определяется размерностью решаемой задачи.

Методы нелинейного программирования применяют для решения оптимальных задач с нелинейными функциями цели. На независимые переменные могут быть наложены ограничения также в виде нелинейных соотношений, имеющих вид равенств или неравенств. По существу, методы нелинейного программирования используют, если ни один из перечисленных выше методов не позволяет сколько-нибудь продвинуться в решении оптимальной задачи. Поэтому указанные методы иногда называют также прямыми методами решения оптимальных задач.

Для получения численных результатов важное место отводится нелинейному программированию и в решении оптимальных задач такими методами, как динамическое программирование, принцип максимума и т. п. на определенных этапах их применения.

Названием «методы нелинейного программирования» объединяется большая группа численных методов, многие из которых приспособлены для решения оптимальных задач соответствующего класса. Выбор того или иного метода обусловлен сложностью вычисления критерия оптимальности и сложностью ограничивающих условий, необходимой точностью решения, мощностью имеющейся вычислительной машины и т.д. Ряд методов нелинейного программирования практически постоянно используется в сочетании с другими методами оптимизации, как, например, метод сканирования в динамическом программировании. Кроме того, эти методы служат основой построения систем автоматической оптимизации - оптимизаторов, непосредственно применяющихся для управления производственными процессами.

Геометрическое программирование есть метод решения одного специального класса задач нелинейного программирования, в которых критерий оптимальности и ограничения задаются в виде позиномов - выражений, представляющих собой сумму произведений степенных функций от независимых переменных. С подобными задачами иногда приходится сталкиваться в проектировании. Кроме того, некоторые задачи нелинейного программирования иногда можно свести к указанному представлению, используя аппроксимационное представление для целевых функций и ограничений.

Специфической особенностью методов решения оптимальных задач (за исключением методов нелинейного программирования) является то, что до некоторого этапа оптимальную задачу решают аналитически, т. е. находят определенные аналитические выражения, например, системы конечных или дифференциальных уравнений, откуда уже отыскивают оптимальное решение. В отличие от указанных методов при использовании методов нелинейного программирования, которые, как уже отмечалось выше, могут быть названы прямыми, применяют информацию, получаемую

при вычислении критерия оптимальности, изменение которого служит оценкой эффективности того или иного действия.

Важной характеристикой любой оптимальной задачи является ее размерность «n», равная числу переменных, задание значений которых необходимо для однозначного определения состояния оптимизируемого объекта. Как правило, решение задач высокой размерности связано с необходимостью выполнения большого объема вычислений. Ряд методов (например, динамическое программирование и дискретный принцип максимума) специально предназначен для решения задач оптимизации процессов высокой размерности, которые могут быть представлены как многостадийные процессы с относительно невысокой размерностью каждой стадии.

За основу характеристики областей применения различных методов оптимизации положена сравнительная оценка эффективности использования каждого метода для решения различных типов оптимальных задач. Классификация задач проведена по следующим признакам:

- вид математического описания процесса;
- тип ограничений на переменные процесса;
- число переменных.

Предполагается, что решение оптимальной задачи для процессов, описываемых системами конечных уравнений, определяется как конечный набор значений управляющих воздействий (статическая оптимизация процессов с сосредоточенными параметрами), а для процессов, описываемых системами обыкновенных дифференциальных уравнений, управляющие воздействия характеризуются функциями времени (динамическая оптимизация процессов с сосредоточенными параметрами) или пространственных переменных (статическая оптимизация процессов с распределенными параметрами).

Классификация задач по группам с числом независимых переменных, большим и меньшим трех или равным трем как характеристика размерности задач с большим и малым числом переменных, разумеется, весьма условна и в данном случае выбрана

скорее из соображений наглядности графического изображения пространства изменения переменных задачи - фазового пространства (при числе переменных большем трех графическое изображение фазового пространства обычными приемами отсутствует). Тем не менее, такая классификация до некоторой степени все же отражает действительные трудности, возникающие при решении задач с размерностью выше трех.

5.1 Метод оптимизации Лагранжа

В общем виде задача нелинейного программирования состоит в определении максимального (минимального) значения функции

$$f(x_1, x_2, \dots, x_n) \tag{154}$$

при условии

$$\begin{cases} g_i(x_1, x_2, \dots, x_n) \le b_i, i = \overline{1,k}, \\ g_i(x_1, x_2, \dots, x_n) \le b_i, i = \overline{k+1,m}, \end{cases} \tag{155}$$

где f и g_i – некоторые известные функции *n* переменных,

b_i – заданные числа.

Класс задач нелинейного программирования шире класса задач линейного программирования. Подробное изучение практических задач, которые условились считать линейными, показывает, что они в действительности являются нелинейными. Существующие методы позволяют решать узкий класс задач, ограничения которых имеют вид $g_i(x_1, x_2, \dots, x_n) = \sum_{j=1}^{n} a_{ij}x_j$ $(i = \overline{1,m})$, а целевая функция является сепарабельной (суммой *n* функций $\varphi_j(x_j)$), или квадратической.

Даже если область допустимых решений – выпуклая, то в ряде задач целевая функция может иметь несколько локальных экстремумов. С помощью большинства

же вычислительных методов можно найти точку локального оптимума, но нельзя установить, является ли она точкой глобального (абсолютного) оптимума или нет. Если задача содержит нелинейные ограничения, то область допустимых решений не является выпуклой и кроме глобального оптимума могут существовать точки локального оптимума.

Рассмотрим частный случай общей задачи нелинейного программирования (154) – (155), предполагая, что система ограничений (155) содержит только уравнения, отсутствуют условия неотрицательности переменных, $f(x_1, x_2, \dots, x_n)$ и $g_i(x_1, x_2, \dots, x_n)$ – функции, непрерывные вместе со своими частными производными. Ограничения в задаче заданы уравнениями, поэтому для ее решения можно воспользоваться классическим методом отыскания условного экстремума функций нескольких переменных. Вводят набор переменных $\lambda_1, \lambda_2, \dots, \lambda_m$, называемых множителями Лагранжа, и составляют функцию Лагранжа

$$F(x_1, \dots, x_n, \lambda_1, \dots, \lambda_m) = f(x_1, \dots, x_n) + \sum_{i=1}^{m} \lambda_i [b_i - g_i(x_1, \dots, x_n)], \qquad (156)$$

находят частные производные

$$\frac{\partial F}{\partial x_j}\ (j = \overline{1,n}), \frac{\partial F}{\partial \lambda_i}\ (i = \overline{1,m}) \qquad (157)$$

и рассматривают систему n + m уравнений с n + m неизвестными $x_1, x_2, \dots, x_n, \lambda_1, \lambda_2, \dots, \lambda_m$

$$\begin{cases} \frac{\partial F}{\partial x_j} = \frac{\partial f}{\partial x_i} - \sum_{i=1}^{m} \lambda_i \frac{\partial g_i}{\partial x_j} = 0,\ j = \overline{1,n} \\ \frac{\partial F}{\partial \lambda_i} = b_i - g_i(x_1, x_2, \dots, x_n) = 0,\ i = \overline{1,m} \end{cases} \qquad (158)$$

Решив систему уравнений (158), получают все точки, в которых функция (154) может иметь экстремальные значения. Дальнейшее исследование найденных точек

проводят так же, как и в случае безусловного экстремума. Метод множителей Лагранжа имеет ограниченное применение, так как система (158), как правило, имеет несколько решений.

Задание. Найти точку условного экстремума функции $f(x) = x_1 \cdot x_2 + x_2 \cdot x_3$ при ограничениях

$$\begin{cases} x_1 + x_2 = 2 \\ x_2 + x_3 = 2 \end{cases}$$

Решение. Составим функцию Лагранжа:

$$F(x_1, x_2, x_3, \lambda_1, \lambda_2) = x_1 \cdot x_2 + x_2 \cdot x_3 + \lambda_1(x_1 + x_2 - 2) + \lambda_2(x_2 + x_3 - 2)$$

Продифференцируем ее по переменным $x_1, x_2, x_3, \lambda_1, \lambda_2$. Приравнивая полученные выражения к нулю, получим следующую систему уравнений:

$$\begin{cases} x_1+\lambda_1=0 \\ x_1+x_3+\lambda_1+\lambda_2=0 \\ x_2+\lambda_2=0 \\ x_1+x_2-2=0 \\ x_2+x_3-2=0 \end{cases}$$

Из второго и третьего уравнений следует, что $\lambda_1 = \lambda_2 = -x_2$; тогда

$$\begin{cases} x_1 - 2x_2 + x_3 = 0 \\ x_1 + x_2 = 2 \\ x_2 + x_3 = 2 \end{cases}$$

Решив данную систему, получим:

$$x_1^* = \mathrm{x}_2^* = x_3^* = 1, f(x^*) = 2.$$

Задание. Имеется два способа производства некоторого продукта. Издержки производства при каждом способе зависят от произведенных y_1 и y_2 следующим образом: $g(y_1) = 9y_1 + y_1^2$, $g(y_2) = 6y_2 + y_2^2$. За месяц необходимо произвести *3×50* единиц продукции, распределив ее между двумя способами так, чтобы минимизировать общие издержки (при решении используйте метод множителей Лагранжа).

Решение. Найдем экстремум функции суммарных издержек $F(x) = 9 \cdot x_1 + +x_1^2 + 6 \cdot x_2 + x_2^2$, используя функцию Лагранжа:

$$L(x_1, \dots, x_n, \lambda_1, \dots, \lambda_m) = F(\bar{X}) + \sum_{i=1}^{m} \lambda_i \cdot \varphi_i, (\bar{X})$$

где $F(\bar{X})$- целевая функция вектора $\bar{X}$;

$\varphi_i, (\bar{X})$- ограничения в неявном виде.

В качестве целевой функции, подлежащей оптимизации, в этой задаче выступает функция:

$$F(x) = 9 \cdot x_1 + +x_1^2 + 6 \cdot x_2 + x_2^2$$

Перепишем ограничение задачи в неявном виде:

$$\varphi_1(\bar{X}) = x_1 + x_2 - 150 = 0$$

Составим вспомогательную функцию Лагранжа:

$$L(X, \lambda) = 9x_1 + x_1^2 + 6x_2 + x_2^2 + \lambda(x_1 + x_2 - 150)$$

Необходимым условием экстремума функции Лагранжа является равенство нулю ее частных производных по переменным x_i и неопределенному множителю λ. Составим систему:

$$\begin{cases} \frac{\partial L}{\partial x_1} = 2x_1 + \lambda + 9 = 0 \\ \frac{\partial L}{\partial x_2} = \lambda + 2x_2 + 6 = 0 \\ \frac{\partial F}{\partial \lambda} = x_1 + x_2 - 150 = 0 \end{cases} \quad (159)$$

Систему (159) решаем с помощью метода Гаусса или используя формулы Крамера.

Запишем систему в виде:

$$\begin{vmatrix} 2 & 0 & 1 \\ 0 & 2 & 1 \\ 1 & 1 & 0 \end{vmatrix} = \begin{vmatrix} -9 \\ -6 \\ 150 \end{vmatrix}$$

Для удобства вычислений поменяем строки местами:

$$\begin{vmatrix} 0 & 2 & 1 \\ 1 & 1 & 0 \\ 2 & 0 & 1 \end{vmatrix} = \begin{vmatrix} -6 \\ 150 \\ -9 \end{vmatrix}$$

Умножим 2-ую строку на (2). Умножим 3-ую строку на (-1). Добавим 3-ую строку к 2-ой:

$$\begin{vmatrix} 0 & 2 & 1 \\ 0 & 2 & -1 \\ 2 & 0 & 1 \end{vmatrix} = \begin{vmatrix} -6 \\ 309 \\ -9 \end{vmatrix}$$

Умножим 2-ую строку на (-1). Добавим 2-ую строку к 1-ой:

$$\begin{vmatrix} 0 & 0 & 2 \\ 0 & 2 & -1 \\ 2 & 0 & 1 \end{vmatrix} = \begin{vmatrix} -315 \\ 309 \\ -9 \end{vmatrix}$$

Из 1-ой строки выражаем λ

$$\lambda = -\frac{315}{2} = -157{,}5$$

Из 2-ой строки выражаем x_2

$$x_2 = \frac{309-(-1)(-157,5)}{2} = \frac{151,5}{2} = 75,75$$

Из 3-ой строки выражаем x_1

$$x_1 = \frac{-9-1(-157,5)}{2} = \frac{148,5}{2} = 74,25$$

Таким образом, чтобы общие издержки производства были минимальны, необходимо производить $y_1 = 74,25$ первым способом производства и вторым способом: $y_2 = 75,75$.

Отметим, что пакет Matlab позволяет решать уравнения и системы уравнений. Для этого следует использовать команду *solve ({функция=0 или набор функций равных нолю})*.

5.2 Метод градиентов

Пусть имеется система нелинейных уравнений вида:

$$\begin{cases} f_1(x_1, x_2, \dots, x_n) = 0 \\ \dots \\ f_n(x_1, x_2, \dots, x_n) = 0 \end{cases}, \quad (160)$$

где $x_1, x_2, \dots, x_n$- неизвестные аргументы;

$f_1, f_2, \dots, f_n$- нелинейные функции n переменных.

Тогда записав (160) в матричной форме получим:

$$f(x) = 0, \quad (161)$$

где

$$f = \begin{vmatrix} f_1 \\ f_2 \\ \cdots \\ f_n \end{vmatrix} \quad (162)$$

Составим знакоположительную функцию $\varphi(x_1, x_2, \dots, x_n)$ и поставим задачу минимизировать её значение:

$$\varphi(x_1, x_2, \dots, x_n) = \sum_{i=1}^{n} f_i^2(x) = f_1^2(x_1, x_2, \dots, x_n) + \cdots + f_n^2(x_1, x_2, \dots, x_n), \quad (163)$$

$$\min\big(\varphi(x_1, x_2, \dots, x_n)\big) = 0 \quad (164)$$

Градиентом функции $\varphi(x)$ в данной точке M_0 называется вектор, расположенный в плоскости аргументов (x_1, x_2), имеющий своим началом эту точку $M_0\ (x_{\{10\}}, x_{\{20\}})$ и имеющий проекции на координатные оси, равные значениям частных производных функции $\varphi(x)$ в этой точке M_0:

$$grad\varphi_{\{M_0\}} = \varphi'_{\{x_1\}}(M_0) \cdot\ i + \varphi'_{\{x_2\}}(M_0) \cdot\ j \quad (165)$$

Градиент функции указывает направление её возрастания, а, следовательно, двигаясь в направлении антиградиента будет выполняться решение задачи минимизации исходной функции. таким образом, выполняя ряд последовательных шагов – приближений, мы достигаем окрестности минимума исследуемой функции множества аргументов. Задав точность решения и сравнивая значения функции нескольких последовательных приближений, мы получаем результат минимизации (или максимизации, при движении в направлении градиента) с заданной точностью (рисунок 62).

Для ускорения поиска решения, движение осуществляют с некоторым шагом λ, чаще – переменным по мере продвижения к решению. Таким образом, каждая последующая итерация будет представлена в виде:

$$x^{p+1} = x^p -\ \lambda_p \cdot \nabla(x^p), p = 0,1,2, \dots, n, \quad (166)$$

где x^p, x^{p+1} – координаты точки текущего и последующего приближений;

$\nabla(x^p)$ – градиент в точке текущего приближения;

p – номер приближения (шага, итерации).

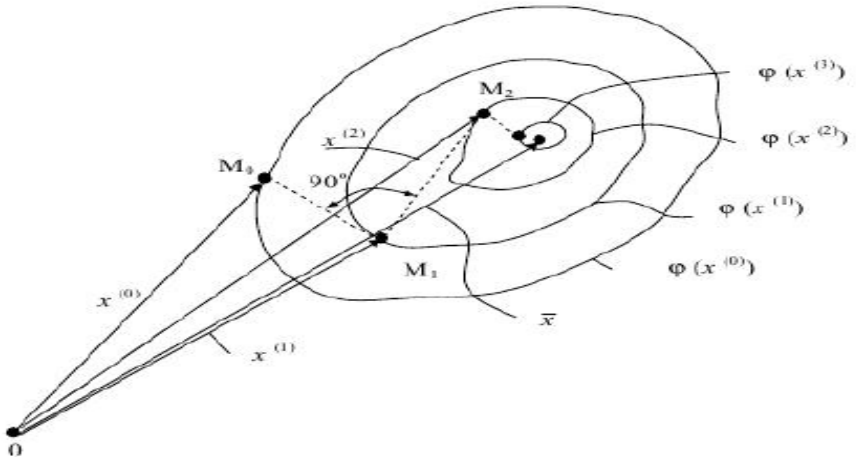


Рисунок 62 – Последовательное приближение в задаче минимизации функции методом градиентного спуска

Длина пути при движении в направлении антиградиента однозначно определяется величиной λ_p, причем каждая координата изменяется по линейному закону:

$$\varphi(x^p) - \lambda_p \cdot \nabla\varphi(x^p) \tag{167}$$

Минимизируя функцию (167) по переменной λ_p можно найти оптимальную длину шага, то есть для поиска оптимального шага на каждом шагу необходимо решать уравнение вида:

$$\frac{d\left(\varphi(x^p) - \lambda_p \cdot \nabla\varphi(x^p)\right)}{d\lambda_p} = 0 \tag{168}$$

Рассмотрим данный метод на примере решения простейшей задачи вида:

$$\begin{cases} x_1 \cdot x_2 - x_2^2 - 1 = 0 \\ x_1 \cdot x_2 - x_2 - 3 = 0 \end{cases} \quad (169)$$

Составим вспомогательную функцию:

$$\varphi(x) = (x_1 \cdot x_2 - x_2^2 - 1)^2 + (x_1 \cdot x_2 - x_2 - 3)^2 \quad (170)$$

Составляющие градиента этой функции:

$$\frac{d\varphi}{dx_1} = 2 \cdot (x_1 \cdot x_2 - x_2^2 - 1) \cdot x_2 + 2 \cdot (x_1 \cdot x_2 - x_2 - 3) \cdot x_2, \quad (171)$$

$$\frac{d\varphi}{dx_2} = 2 \cdot (x_1 \cdot x_2 - x_2^2 - 1) \cdot (x_1 - 2 \cdot x_2) + 2 \cdot (x_1 \cdot x_2 - x_2 - 3)X$$

$$X(x_1 - 1) \quad (172)$$

Выберем точку начального приближения M_0 (1;1) и найдем для неё составляющие градиента:

$$\frac{d\varphi}{dx_1} = 2 \cdot (1 \cdot 1 - 1^2 - 1) \cdot 1 + 2 \cdot (1 \cdot 1 - 1 - 3) \cdot 1 = -8,$$

$$\frac{d\varphi}{dx_2} = 2 \cdot (1 \cdot 1 - 1^2 - 1) \cdot (1 - 2 \cdot 1) + 2 \cdot (1 \cdot 1 - 1 - 3) \cdot (1 - 1) = 2$$

Координаты точки нового приближения получим, используя (166):

$$x_1^1 = 1 + \lambda_1 \cdot 8,$$

$$x_2^1 = 1 - \lambda_1 \cdot 2$$

Отсюда

$$\varphi(x^1) = \varphi\big(x_0 - \lambda_0 \cdot \nabla\varphi(\mathrm{x}_0)\big) = [(1 + 8\lambda_0) \cdot (1 - 2\lambda_0) - (1 - 2\,\lambda_0)^2 — 1]^2 +$$
$$+[(1 + 8\lambda_0) \cdot (1 - 2\lambda_0) - 1 + 2\lambda_0 - 3]^2 = (20\lambda_0^2 - 10\lambda_0 + 1)^2 + (16\lambda_0^2 - 8\lambda_0 + 3)^2$$

Оптимальный шаг определим минимизируя эту функцию по λ_0:

$$\frac{d\varphi(x_0-\lambda_0\cdot\nabla\varphi(x_0))}{d\lambda_0} = 2\cdot(20\lambda_0^2-10\lambda_0+1)\cdot(40\lambda_0-10)+2\cdot(16\lambda_0^2-8\lambda_0++3)\cdot(32\lambda_0-8)=0$$

После преобразования получим:

$$656\lambda_0^3-492\lambda_0^2+150\lambda_0-17=0$$

Наименьший действительный корень составит величину:

$$\lambda_0=\frac{1}{4}$$

Определим первое приближение корней:

$$x_1^1=1+8\lambda_0=1+8\cdot\tfrac{1}{4}=3,$$

$$x_2^1=1-2\lambda_0=1-2\cdot\tfrac{1}{4}=0{,}5$$

Таким образом получили очередную точку M_1 (3;0,5), координаты которой следует подставить в функцию $\varphi(x)$ и выполнить минимизацию по λ_1 и т.д.

В качестве критериев достигнутой точности полагают:

$$x_j^p-x_j^{p+1}\le\epsilon_{\text{доп}}, \qquad (173)$$

и / или

$$f_i(x^{p+1})\le\psi_{\text{доп}}, \qquad (174)$$

где $\epsilon_{\text{доп}}, \psi_{\text{доп}}$ – допустимые точность нахождения параметров уравнения и точность решения.

5.3 Динамическое программирование в задачах оптимизации

Динамическое программирование обычно относят к методам нелинейного математического программирования. Оно позволяет оптимизировать многошаговые операции.

Операция – это любое мероприятие (или система действий), объединенное единым замыслом и направленное к достижению поставленной цели. Операция всегда управляема. От нас зависит выбор тех или иных параметров, характеризующих способ ее организации. Для сравнения различных вариантов выполнения операции вводят целевую функцию. Конкретный вид ее зависит от задачи исследования. При применении метода динамического программирования операция разбивается на ряд последовательных шагов. Некоторые задачи позволяют естественным образом производить такое разбиение. В других случаях его приходится делать искусственно.

В общем случае задача динамического программирования ставится так. Пусть имеется некоторая операция с суммирующейся (или аддитивной) целевой функцией Z. Эта операция естественно или искусственно распадается на «k» шагов. На каждом шаге i определяются переменные $x_1^i, \dots, x_n^i$, а также значения целевой функции $Z_i(x_1^i, \dots, x_n^i)$. Требуется найти такие значения переменных (т. е. найти оптимальное управление всей операцией), чтобы целевая функция $Z\left(x_1^i, \dots, x_n^i\right) = \sum Z_i\left(x_1^i, \dots, x_n^i\right)$ обратилась бы в максимум или минимум, в зависимости от постановки задачи.

Решать поставленную многошаговую задачу в целом сложно. Проще искать оптимальное управление шаг за шагом, на каждом этапе оптимизируя только один шаг. На первый взгляд все кажется очень простым. Вместо поиска оптимального управления всей операцией следует оптимизировать управление на каждом отдельном шаге, что проще решения целиком всей задачи.

Оптимальное управление всей операцией получится как результат оптимального управления на отдельных шагах. Такой подход был бы правилен в том случае, если оптимальное управление на каждом шаге можно было бы выбирать независимо от управления на других, в частности, последующих шагах. Иными словами, при выборе шагового управления необходимо учитывать его последействия на будущих шагах.

Очевидно, что это общее правило имеет исключение. Последний шаг можно планировать без оглядки на будущее. Управление на последнем шаге можно выбирать таким, чтобы оно приносило бы максимальный выигрыш. Поэтому решение задачи методом динамического программирования производится, начиная с последнего, k-го шага, т. е. двигаясь от конца к началу. Выбирать управление на k-м шаге мы можем только в том случае, если знаем, чем закончился выбор на предыдущем, (k-1)-м шаге. Поскольку мы этого не знаем, то задаемся различными предположениями на этот счет.

Для каждого такого предположения находим оптимальное управление на k -м шаге. Это управление следует считать не оптимальным, а условно-оптимальным, поскольку мы не знаем, какое управление на (k-1)-м шаге действительно выбрано. После этого мы переходим к выбору управления на предпоследнем, (k-1)-м шаге. Здесь мы также должны сделать предположение о том, чем закончился (k-2)-й шаг, и для каждого из этих предположений определить управление на (k -1)-м шаге так, чтобы выигрыш на последних двух шагах был бы максимален, т. е.

$$Z_k + Z_{k-1} \rightarrow \max \text{ или } min \qquad (175)$$

Далее переходим к (k-2)-му шагу и т. д. Таким образом, на каждом шаге ищется такое управление, которое обеспечивает оптимальное продолжение процесса относительно достигнутого в данный момент состояния. В результате мы получаем условно-оптимальное управление на всех шагах.

Теперь, если мы, закончив определение условно-оптимального управления на первом шаге, подошли к началу операции и знаем ее начальное состояние S_o, мы можем начать движение от начала к концу.

Зная S_o, мы найдем оптимальное управление на первом шаге и определим состояние S_2 в начале второго шага. Далее находим оптимальное управление на втором шаге и состояние S_2 в начале третьего шага и т. д.

В конце концов, мы найдем оптимальное управление всей операцией, приводящее к Z_{max} (или Z_{min}).

Таким образом, многошаговый процесс проходится дважды:

- первый раз – от конца к началу (находятся условные оптимальные управления на каждом шаге и условный выигрыш на всех шагах);

- второй раз – от начала к концу (находятся оптимальные шаговые управления и действительный выигрыш на всех шагах).

5.3.1 Пример применения алгоритма динамического программирования в химической технологии

Рассмотрим применение динамического программирования для решения конкретной задачи. Необходимо проложить трубную линию (трубопровод) в условиях крупного цеха из точки А в точку A_1. Рассматриваемая часть цеха разбита на производственные участки (рисунок 63). Прокладку можно производить только вдоль границ участков и проездов. Затраты на прокладку линии по участкам в условных единицах указаны на рисунке 63.

В нашем случае вся операция легко разбивается на отдельные шаги. За шаг можно принять прокладку линии на отдельном участке цеха или проезда. Следует выбрать трассу прокладки таким образом, чтобы суммарные затраты по всей линии были бы наименьшими. Таким образом, как следует из рисунка 63, вся задача разбивается на восемь шагов.

Начнем решение с последнего восьмого шага. Сначала посмотрим, каким образом мы можем попасть в точку A_1 за один шаг. Очевидно, что это можно сделать только из точек B_1 и B_2 (рисунок 63), при этом из каждой из них только единственным образом: из точки B_1 – по горизонтали, из B_2 – по вертикали. Иными словами, на последнем шаге нет выбора условного направления, оно единственное.

Изобразим эти условные направления из точек B_1 и B_2 на последнем шаге стрелками и в соответствующих кружках покажем затраты, связанные с этим (12 и 13 единиц). Таким образом, мы нашли условно-оптимальное управление на последнем, восьмом, шаге для любого (B_1 или B_2) исхода предпоследнего, седьмого, шага. Для каждого из этих исходов найдены условные минимальные затраты на последнем, восьмом, шаге.

Перейдем к планированию предпоследнего, седьмого, шага. Для этого рассмотрим все возможные результаты шестого шага. После шестого шага мы можем оказаться в одной из точек: C_1, C_2 или C_3. Только из них мы можем попасть за один шаг в точки B_1 и B_2. Если после шестого шага мы оказались в точке C_1, то здесь у нас нет выбора. Для того чтобы попасть в точку B_1, можно двигаться только по горизонтали. При этом суммарные затраты на двух последних шагах составят 23 единицы. Так же обстоит дело с точкой C_3. Из нее движение возможно только вверх, суммарные затраты равны 25 единицам. Из точки же C_2 можно двигаться по двум направлениям. Если двигаться по горизонтали, то суммарные затраты будут равны 27, если двигаться по вертикали, то они будут меньше – 22 единицы. Мы выбираем движение по вертикали и в кружке у точки C_2 ставим стрелку вверх.

Продолжая наши рассуждения, мы дойдем до исходной точки А. Теперь оптимальная трасса прокладки трубопровода ясна. Она показана штриховой линией на рисунке 63. Минимальные затраты при этом составят 75 единиц.

При решении подобной задачи может оказаться, что при условной оптимизации оба управления на каком-то шаге являются одинаковыми, т.е. приводят к одинаковым затратам при движении от соответствующей точки до конца. В этом случае можно

выбирать любое движение в произвольном направлении. В нашем случае такой точкой является точка E на рисунке 63. Движение в любом из направлений из этой точки дает расход затрат в 45 единиц. Мы выбираем движение из точки E по горизонтали.

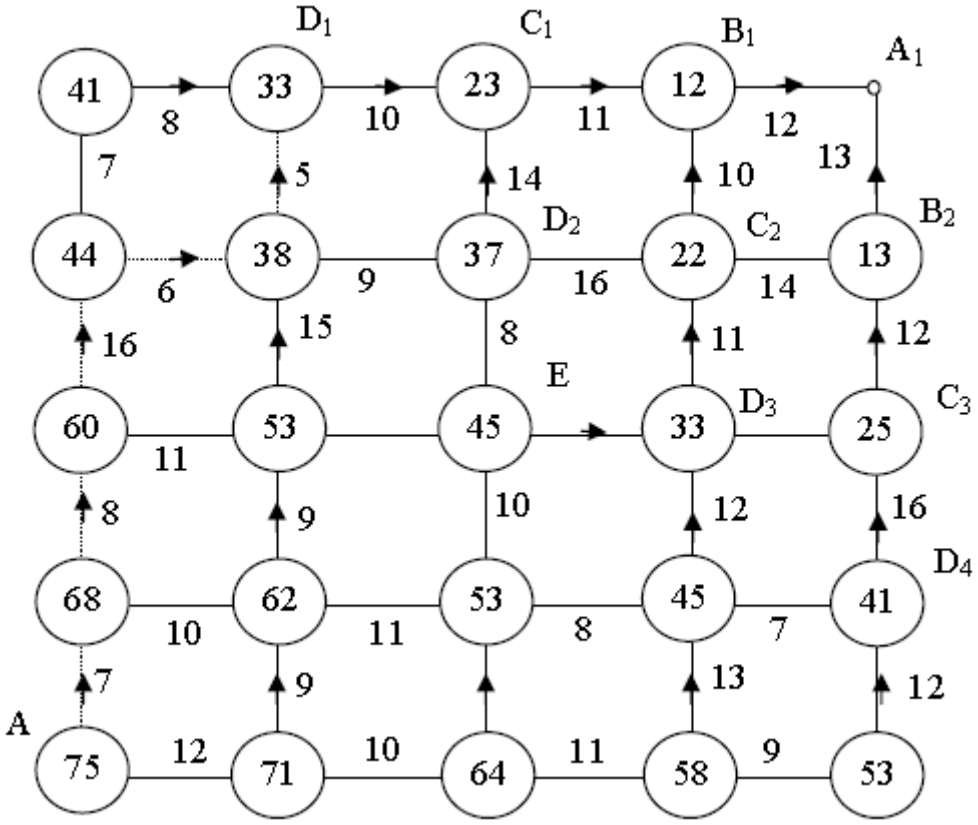


Рисунок 63 – Оптимальный путь по сетке

5.3.2 Реализация алгоритма в MS Excel

Рассмотренный нами алгоритм программируется довольно легко. Рассмотрим реализацию данного алгоритма с применением офисного пакета компании Microsoft. Для начала создадим на листе MS Excel цветовую сетку 9X9 с ячейками желтого и серого цветов, имитирующих проезды и расположение участков для удобного ввода исходных данных (рисунок 64). В полях серого цвета введем относительную стоимость прокладки трубопровода через этот участок. Ячейки желтого цвета будем заполнять при помощи рассмотренного ранее алгоритма.

Для этого присвоим значения стоимости соответствующей прокладки трубопровода по участкам с номерами точек B2 и C2 записав в ячейке (4,8): «=RC[1]», а в ячейке (6,10) – «=R[-1]C». В остальных желтых ячейках верхнего ряда укажем «=RC[1]+RC[2]», нижнего ряда – «=R[-2]C+R[-1]C».

Ячейки отличные от крайних заполняются по принципу минимизации управления на каждом шаге, для этого в каждой такой ячейке используют оператор вариантов следующим образом: «=ЕСЛИ((RC[1]+RC[2])>(R[-1]C+R[-2]C);(R[-1]C+R[-2]C);RC[1]+RC[2])». Причем во всех ячейках, кроме отмеченных ранее, вводимая формула будет одинаковой.

В результате получаем простейший расчет оптимального решения (рисунок 65).

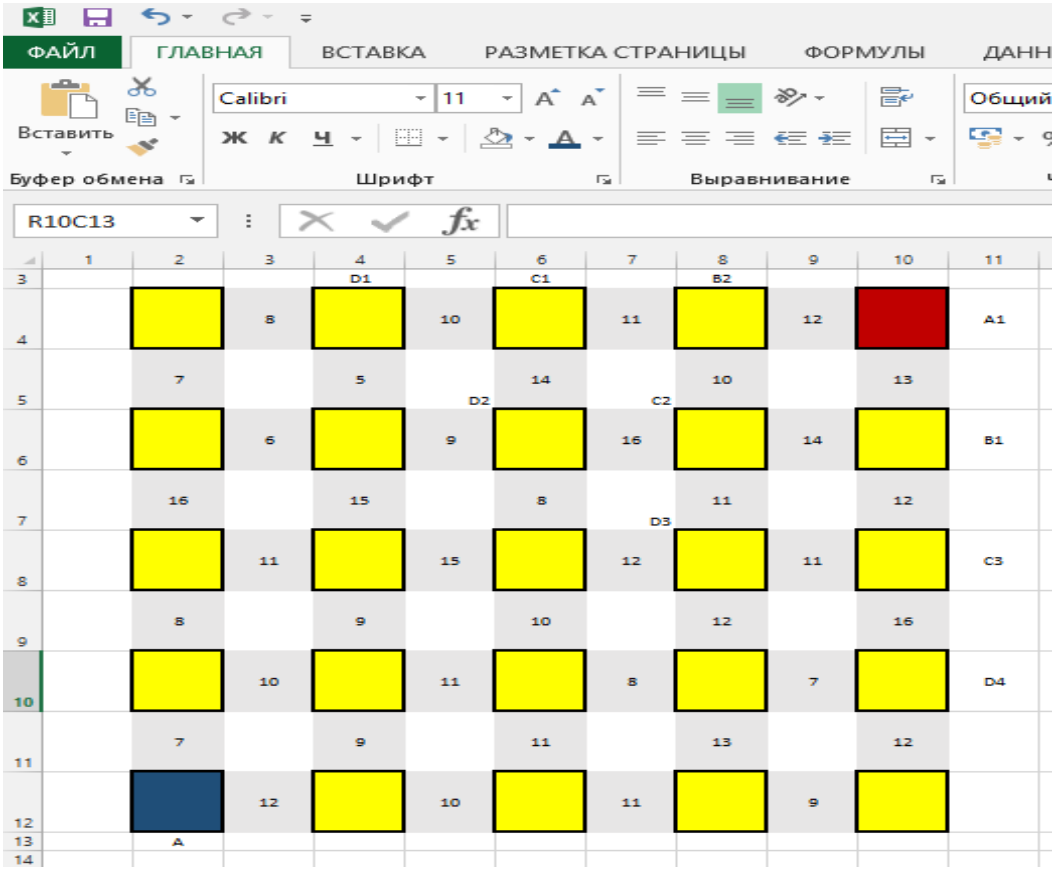


Рисунок 64 – Макет с исходными данными

Заметим, что основной расчет выполнен, однако нашей модели не хватает наглядности. Будем отмечать оптимальную траекторию перекрашивая ячейки участков, через которые будет прокладываться трубопровод в красный цвет. Для автоматической реализации данной задачи будем использовать макросы, написанные на встроенном в MS Office языке программирования Visual Basic for Application (VBA). Перед этим необходимо позаботиться о том, чтобы в настройках безопасности офисного пакета был разрешен запуск макросов.

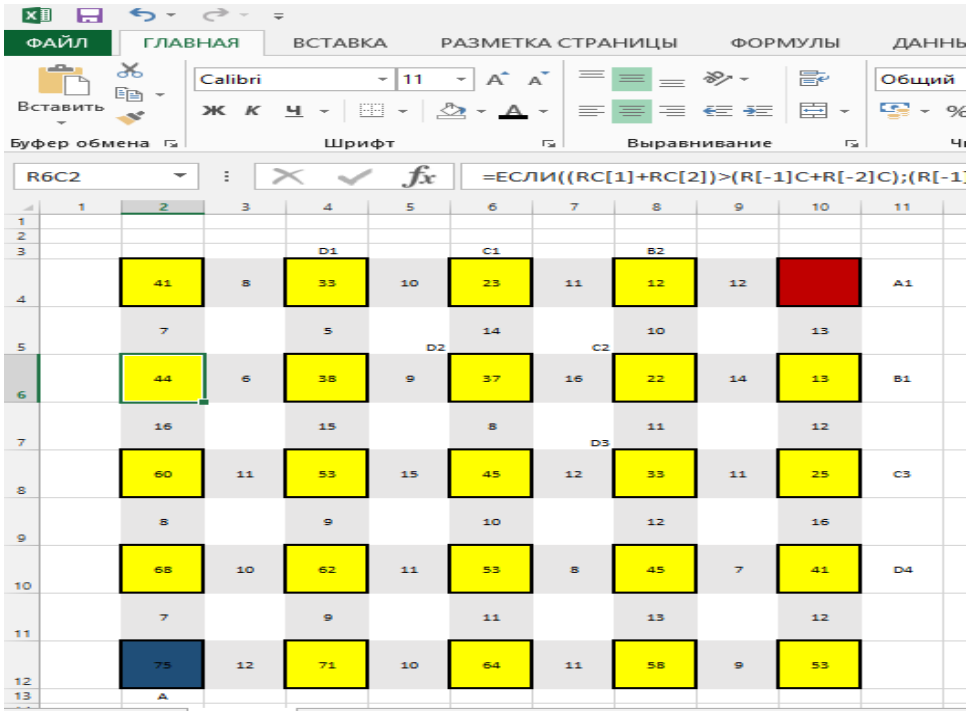


Рисунок 65 – Макет с исходными данными и расчетом

Поместим справа от основного макета два элемента управления – кнопки. Переименуем их в «Очистить» и «Показать оптимальную траекторию» (рисунок 66).

При создании кнопки (вкладка «Разработчик» – «Вставить» – «Элементы управления формы» – «Кнопка») появится окно, предлагающее назначить макрос объекту. Оставим имя макроса по умолчанию и нажмем «Создать». Дальнейший код – обработчик событий будем писать в открывшемся редакторе VBA.

Макрос, назначенный кнопке «Очистить» будет возвращать исходную цветовую схему, а макрос кнопки «Показать оптимальную траекторию» будет менять цвета ячеек, лежащих на пути оптимального маршрута на красный.

И так, исходный код кнопки «Очистить» выглядит так:

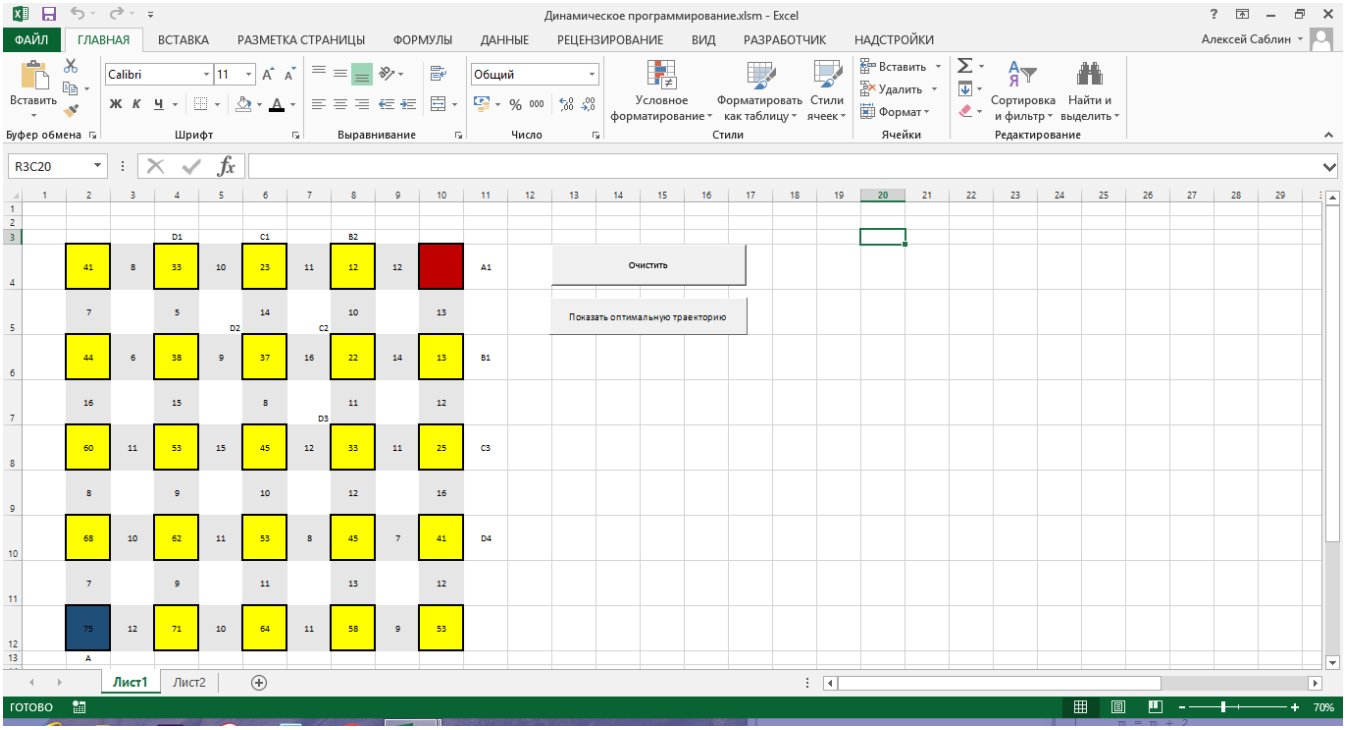


Рисунок 66 – Макет с исходными данными, расчетом и элементами управления

```
Sub Кнопка1_Щелчок()
Sheets(«Лист1»).Select
k = 4
m = 2
For i = 1 To 5
For j = 1 To 5
ActiveSheet.Cells(k, m).Select
  With Selection.Interior
      .Pattern = xlSolid
      .PatternColorIndex = xlAutomatic
      .Color = 65535
```

```
        .TintAndShade = 0
        .PatternTintAndShade = 0
    End With
k = k + 2
Next j
m = m + 2
k = 4
Next i
ActiveSheet.Cells(12, 2).Select
  With Selection.Interior
        .Pattern = xlSolid
        .PatternColorIndex = xlAutomatic
        .ThemeColor = xlThemeColorAccent1
        .TintAndShade = -0,499984740745262
        .PatternTintAndShade = 0
    End With
ActiveSheet.Cells(4, 10).Select
With Selection.Interior
        .Pattern = xlSolid
        .PatternColorIndex = xlAutomatic
        .Color = 192
        .TintAndShade = 0
        .PatternTintAndShade = 0
    End With
End Sub
```

Для кнопки «Показать оптимальную траекторию» макрос имеет более сложный вид:

```
Sub Кнопка2_Щелчок()
k = 10
```

```
m = 2
For i = 1 To 23
If m > 8 Then
If (m = 10) And (k > 2) Then
ActiveSheet.Cells(k, m).Select
With Selection.Interior
        .Pattern = xlSolid
        .PatternColorIndex = xlAutomatic
        .Color = 192
        .TintAndShade = 0
        .PatternTintAndShade = 0
    End With
   If k > 4 Then
   k = k - 2
   End If
    GoTo m0
End If
GoTo m1
End If
m0:
If k >= 4 Then
If (ActiveSheet.Cells(k, m).Value + ActiveSheet.Cells(k + 1, m).Value) <= (Ac-
tiveSheet.Cells(k + 2, m + 2).Value + ActiveSheet.Cells(k + 2, m + 1).Value) Then
ActiveSheet.Cells(k, m).Select
With Selection.Interior
        .Pattern = xlSolid
        .PatternColorIndex = xlAutomatic
        .Color = 192
        .TintAndShade = 0
```

```
        .PatternTintAndShade = 0
    End With
    Else
    ActiveSheet.Cells(k + 2, m + 2).Select
With Selection.Interior
        .Pattern = xlSolid
        .PatternColorIndex = xlAutomatic
        .Color = 192
        .TintAndShade = 0
        .PatternTintAndShade = 0
    End With
    k = k + 2
    m = m + 2
   End If
 k = k - 2
 Else
 ActiveSheet.Cells(k + 2, m + 2).Select
 With Selection.Interior
        .Pattern = xlSolid
        .PatternColorIndex = xlAutomatic
        .Color = 192
        .TintAndShade = 0
        .PatternTintAndShade = 0
    End With
     m = m + 2
End If
 Next i
m1:
End Sub
```

Результат выполнения макроса, вызванного нажатием данной кнопки показан на рисунке 67. Красным цветом показана оптимальная траектория. При изменении исходных данных (стоимость прокладки трубопровода вдоль каждого участка цеха) следует нажать кнопку «Очистить» для возврата исходной цветовой схемы, а после нажать кнопку «Показать оптимальную траекторию» и получить отображения оптимального маршрута.

Напомним, что значения в желтых, красных и синей ячейках рассчитываются автоматически и меняются в зависимости от исходных данных.

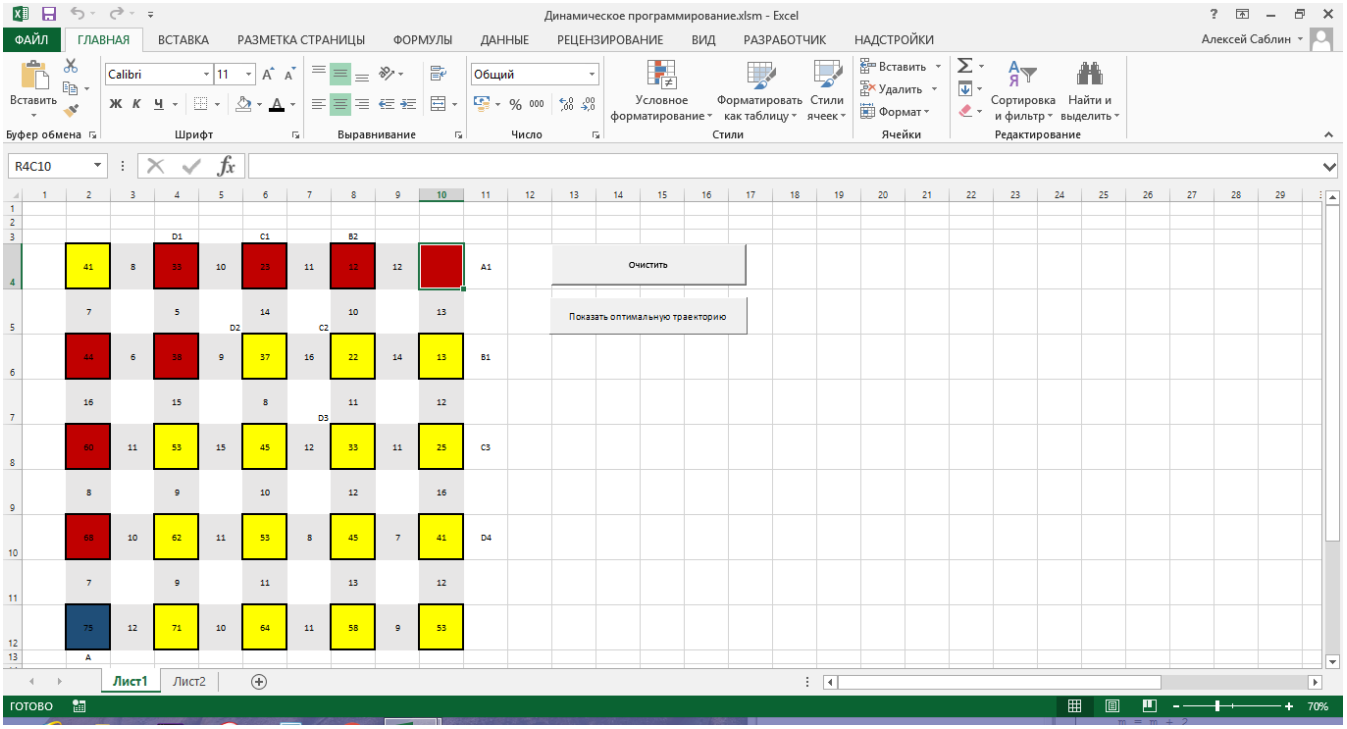


Рисунок 67 – Результат поиска оптимального решения

Контрольные вопросы и задания к разделу 5

1) Раскройте сущность метода Лагранжа в решении задач оптимизации функции с наличием ограничений.

2) Перечислите и дайте краткую характеристику методов линейного программирования в решении задач оптимизации.

3) Перечислите и дайте краткую характеристику методов нелинейного программирования в решении задач оптимизации.

4) Что такое и как применять методы геометрического программирования?

5) Найти точку условного экстремума функции

$$f(x) = x_1 + x_2 + x_2 \cdot x_3$$

при ограничениях

$$\begin{cases} x_1 + x_2 = 2 \\ x_2 + x_3 = 2 \end{cases}$$

6) Дайте определение понятию градиент.

7) Как использовать метод градиентов для решения систем уравнений и задач оптимизации, приведите пример. Решить задачу методом градиентов с максимальной погрешностью нахождения корня 0,01:

$$\begin{cases} x_1 \cdot x_2 - x_2^2 - 1 = 0 \\ x_1 \cdot x_2 - x_2 - 3 = 0 \end{cases}$$

8) Сущность методов динамического программирования. Недостатки и достоинства метода. Реализуйте в Ms Excel решение задачи, разобранной в п.5.3 и найдите оптимальную траекторию прокладки трубопровода. если стоимость работ на участках A_1-B_2 изменится до 5 единиц, а на участке B_2-C_2 до 3 единиц. Остальные данные останутся неизменными.

Список литературы

1 Model Reduction and Coarse-Graining Approaches for Multiscale Phenomena (англ.). Springer, Complexity series, Berlin-Heidelberg-New York, 2006. XII+562 pp. ISBN 3-540-35885-4. Проверено 18 июня 2013. Архивировано из первоисточника 19 июня 2013.

2 Анищенко В. С., Динамические системы, Соросовский образовательный журнал, 1997, № 11, с. 77-84.

3 Арнольд В. И. Жёсткие и мягкие математические модели. - М.: МЦНМО, 2004. - ISBN 5-94057-134-4.

4 Безручко Б. П., Смирнов Д. А. Математическое моделирование и хаотические временные ряды. - Саратов: ГосУНЦ «Колледж», 2005.- ISBN 5-94409-045-6.

5 Блехман И. И., Мышкис А. Д., Пановко Н. Г. Прикладная математика: Предмет, логика, особенности подходов. С примерами из механики: Учебное пособие. - 3-е изд., испр. и доп. - М.: УРСС, 2006. - 376 с. - ISBN 5-484-00163-3

6 Введение в математическое моделирование. Учебное пособие. Под ред. П. В. Трусова. - М.: Логос, 2004. - ISBN 5-94010-272-7.

7 Вероятностные разделы математики / Под ред. Ю. Д. Максимова. - Спб.: «Иван Фёдоров», 2001. - 592 с. - ISBN 5-81940-050-X.

8 Горбань А. Н., Хлебопрос Р. Г., Демон Дарвина: Идея оптимальности и естественный отбор. - М: Наука. Гл ред. физ.-мат. лит., 1988. - 208 с. - (Проблемы науки и технического прогресса) - ISBN 5-02-013901-7 (Глава «Изготовление моделей»)

8 Данилов Ю. А., Лекции по нелинейной динамике. Элементарное введение. Серия «Синергетика: от прошлого к будущему». Изд.2. - М.: URSS, 2006. - 208 с. ISBN 5-484-00183-8

9 Дьяконов В. П. Matlab R2006/2007/2008. Simulink 5/6/7. Основы применения. Серия: Библиотека профессионала. - М.: Солон-Пресс, 2008. - 800 с. - ISBN 978-5-91359-042-8

10 Дьяконов В.П., Круглов В.В. MATLAB 6,5 SP1/7/7 SP1/7 SP2 Simulink 5/6. Инструменты искусственного интеллекта и биоинформатики. - М.: СОЛОН-ПРЕСС, 2006. - 456с.

11 Заде Л. Понятие лингвистической переменной и ее применение к принятию приближенных решений. - М.: Мир, 1976. - 167 с.

12 Зайцевский И.В., Свиридов А.П., Слесарев Д.А. Нейронные сети и их приложения. - М.: МЭИ, 2002. - 95с.

13 Каллан Р. Основные концепции нейронных сетей. - М.: Издательский дом «Вильямс», 2001. - 287с.

14 Краснощёков П. С., Петров А. А. Принципы построения моделей. - издание второе, пересмотренное и дополненное. - М.: ФАЗИС; ВЦ РАН, 2000. - 412 с. - (Математическое моделирование; Вып.1). - ISBN 5-7036-0061-8.

15 Круглов В.В., Борисов В.В. Искусственные нейронные сети. - М.: Горячая линия - Телеком, 2001. - 382с.

16 Лазарев А.И., Харламов И.П., Яковлев П.Я., Яковлева Е.Ф. Справочник химика-аналитика .- М.: Металлургия, 1976.- с 184

17 Медведев В.С., Потемкин В.Г. Нейронные сети. Матлаб 6. - М.: Диалог МИФИ, 2002. - 496с.

18 Мучник Д.А, Гуляев В.М. Расчеты и прогнозирование показателей качества металлургического кокса с использованием ПК. Учебное пособие для студентов специальности 7,090401 «Металлургия чёрных металлов», 7,091604 «Химическая технология топлива и углеродных материалов». - Днепродзержинск.: издательство Днепродзержинского государственного технического университета, 2007. - 225 с.

19 Мышкис А. Д., Элементы теории математических моделей. - 3-е изд., испр. - М.: КомКнига, 2007. - 192 с. - ISBN 978-5-484-00953-4

20 Нейронные сети. STATISTICA Neural Networks. - М.: Горячая линия - Телеком, 2001. - 182с.

21 Осовский С. Нейронные сети для обработки информации. - М.: Финансы и статистика, 2002. - 344с.

22 Петров А. А., Поспелов И. Г., Шананин А. А. Опыт математического моделирования экономики. - М.: Энергоатомиздат, 1996. - 544 с. – ISBN 5 - 7036 - 0061-8.

23 Peierls R. Model-Making in Physics. — Contemp. Phys., January/February 1980, v. 21, pp. 3-17; Перевод: Пайерлс Р., Построение физических моделей, УФН, 1983, № 6.

24 Самарский А. А., Михайлов А. П. Математическое моделирование. Идеи. Методы. Примеры. - 2-е изд., испр. - М.: Физматлит, 2001. - ISBN 5-9221-0120-X.

25 Севостьянов, А. Г. Моделирование технологических процессов: учебник / А. Г. Севостьянов, П. А. Севостьянов. - М.: Легкая и пищевая промышленность, 1984. - 344 с.

26 Советов Б. Я., Яковлев С. А., Моделирование систем: Учеб. для вузов - 3-е изд., перераб. и доп. - М.: Высш. шк., 2001. - 343 с. - ISBN 5-06-003860-2

27 Тарков М.С. Нейрокомпьютерные системы. М.: БИНОМ, 2006. - 142с.

28 Уоссермен Ф. Нейрокомпьютерная техника: Теория и практика. - М.: Мир, 1992.

29 Цымбал Б. П. Математическое моделирование сложных систем в металлургии. - Кемерово-Москва: «Российские университеты» Кузбассвузиздат - АСТШ, 2006. - ISBN 5-202-00925-9.

САБЛИН АЛЕКСЕЙ ВАЛЕРЬЕВИЧ

МОДЕЛИРОВАНИЕ ХИМИКО - ТЕХНОЛОГИЧЕСКИХ ПРОЦЕССОВ
Учебное пособие по дисциплинам
«Моделирование химико-технологических процессов»,
«Химическая технология топлива и углеродных материалов»

Направление подготовки: «Химическая технология»
Профиль «Химическая технология природных энергоносителей и углеродных материалов»
Квалификация (степень) выпускника: бакалавр
Форма обучения: очная, заочная

Подписано в печать 16.11.2016 г.		
Формат 60x90 1/16 Рег.№ 89	Печать цифровая Тираж 100 экз.	Уч.-изд.л. 11,3

ФГАОУ ВО
Национальный исследовательский технологический университет «МИСиС»
Новотроицкий филиал
462359, Оренбургская обл., г. Новотроицк, ул. Фрунзе, 8.
E-mail: nfmisis@yandex.ru
Контактный тел. 8 (3537) 679729.