МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное бюджетное образовательное учреждение высшего образования

«ТОМСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ СИСТЕМ УПРАВЛЕНИЯ И РАДИОЭЛЕКТРОНИКИ» (ТУСУР)

Методические указания к лабораторным работам по дисциплине “Математическое моделирование”

Уровень основной образовательной программы: магистратура

Направление подготовки магистра: 09.04.04 «Программная инженерия»

Магистерская программа: «Методы и технологии индустриального проектирования программного обеспечения»

Форма обучения: очная

Факультет систем управления (ФСУ)

Кафедра автоматизации обработки информации (АОИ)

Курс 1 Семестр 1, 2

Разработчики:

профессор каф. АОИ

_____Н.В. Замятин

Томск 2017

СОДЕРЖАНИЕ

Введение

Цель изучения дисциплины

Задачей математического моделирования как научного направления является представление и исследование моделей различных предметных областей. Из всего многообразия научных и технических исследований, определяемых математическим моделированием, в учебной дисциплине «Математическое моделирование» выбраны направления, связанные с проблемами аналитического и имитационного моделирования, а также с использованием нейросетевых моделей, искусственного интеллекта и нечеткого подхода.

1. Лабораторная работа №1. Визуальное моделирование в пакете simulink

Цель работы: изучение пакета Simulink и методик визуального моделирования.

Общие сведения

MATLAB – это высокоэффективный язык инженерных и научных вычислений, поддерживающий математические вычисления, визуализацию научной графики и программирование с использованием легко осваиваемого операционного окружения, когда задачи и их решения могут быть представлены в нотации, близкой к математической.

Система MATLAB разработана Молером (C. B. Moler) и с конца 70-х годов широко использовалась на больших ЭВМ. В начале 80-х годов Джон Литл (John Little) из фирмы MathWorks, Inc. разработал версии системы PC MATLAB для компьютеров класса IBM PC, VAX и Macintosh. В дальнейшем были созданы версии для рабочих станций Sun, компьютеров с операционной системой UNIX и многих других типов больших и малых ЭВМ. Сейчас свыше десятка популярных компьютерных платформ могут работать с

системой MATLAB. К расширению системы были привлечены крупнейшие научные школы мира в области математики, программирования и естествознания. Матлаб непрерывно развивается и теперь появилась новейшая версия этой системы — MATLAB 7.

Возможности MATLAB весьма обширны, и применимы для расчетов практически в любой области науки и техники. Например, MATLAB широко используется при математическом моделировании механических устройств и систем, в частности в динамике, гидродинамике, аэродинамике, акустике, энергетике и т. д. Этому способствует не только расширенный набор матричных и иных операций и функций, но и наличие пакета расширения (toolbox) Simulink, специально предназначенного для решения задач блочного моделирования динамических систем и устройств, а также десятков других пакетов расширений.

В системе MATLAB содержатся специальные средства для электро-технических и радиотехнических расчетов (операции с комплексными числами, матрицами, векторами и полиномами, обработка данных, анализ сигналов и цифровая фильтрация), обработки изображений, реализации нейронных сетей, а также средства, относящиеся к другим новым направлениям науки и техники. Они иллюстрируются множеством практически полезных примеров.

Важными достоинствами системы являются ее открытость и расширяемость. Большинство команд и функций системы реализованы в виде текстовых m-файлов (с расширением m) и файлов на языке Си, причем все файлы доступны для модификации. Пользователю дана возможность создавать не только отдельные файлы, но и библиотеки файлов для реализации специфических задач.

Легкость модификации системы и возможность ее адаптации к решению задач науки и техники привели к созданию десятков пакетов прикладных программ (toolbox), расширяюших сферы применения системы. Некоторые из них, например Notebook (интеграция с текстовым процессором Word и подготовка «живых» электронных книг), Symbolic Math и Extended Symbolic Math (символьные вычисления с применением ядра системы Maple V R5) и Simulink (моделирование динамических систем и устройств, заданных в виде системы блоков), настолько органично интегрировались с системой MATLAB, что стали ее составными частями.

С системой MATLAB могут интегрироваться такие популярные математические системы, как Mathcad, Maple V и Mathematica.

Новые свойства системе MATLAB придала ее интеграция с программной системой Simulink, созданной для моделирования динамических систем и устройств, заданных в виде системы блоков. Базируясь на принципах визуально-ориентированного программирования, Simulink позволяет выполнять моделирование сложных устройств с высокой степенью достоверности и с средствами представления результатов. В свою очередь, многие другие математические системы, например Mathcad и Maple, допускают установление объектных и динамических связей с системой MATLAB, что позволяет использовать в них эффективные средства MATLAB для работы с матрицами.

Программа Simulink является приложением к пакету MATLAB. При моделировании с использованием Simulink реализуется принцип визуального программирования, в соответствии с которым, пользователь на экране из библиотеки стандартных блоков создает модель устройства и осуществляет расчеты. При этом, в отличие от классических способов моделирования, пользователю не нужно досконально изучать язык программирования и численные методы математики, а достаточно общих знаний требующихся

при работе на компьютере и знаний той предметной области, в которой он работает.

Simulink является самостоятельным инструментом MATLAB и при работе с ним не требуется знать MATLAB и остальные его приложения. С другой стороны доступ к функциям MATLAB и другим его инструментам остается открытым и их можно использовать в Simulink. Часть входящих в состав пакетов имеет инструменты, встраиваемые в Simulink (например, LTI-Viewer приложения Control System Toolbox – пакета для разработки систем управления). Имеются также дополнительные библиотеки блоков для разных областей применения (например, Power System Blockset – моделирование электротехнических устройств, Digital Signal Processing Blockset – набор блоков для разработки цифровых устройств и т.д).

При работе с Simulink пользователь имеет возможность модернизировать библиотечные блоки, создавать свои собственные, а также составлять новые библиотеки блоков.

При математическом моделировании пользователь может выбирать метод решения дифференциальных уравнений, а также способ изменения модельного времени (с фиксированным или переменным шагом). В ходе моделирования имеется возможность следить за процессами, происходящими в системе. Для этого используются специальные устройства наблюдения, входящие в состав библиотеки Simulink. Результаты моделирования могут быть представлены в виде графиков или таблиц.

Преимущество Simulink заключается также в том, что он позволяет пополнять библиотеки блоков с помощью подпрограмм написанных как на языке MATLAB, так и на языках C + +, Fortran и Ada.

Запуск Simulink

Для запуска программы необходимо предварительно запустить пакет MATLAB. Основное окно пакета MATLAB показано на рисунке 1. Там же показана подсказка появляющаяся в окне при наведении указателя мыши на ярлык Simulink в панели инструментов.

Чтобы запустить программу дважды щелкните на иконку 

. Перед Вами откроется рабочая среда, изображенная на рисунке 1. Рабочая среда MatLab 6.x немного отличается от рабочей среды предыдущих версий, она имеет более удобный интерфейс для доступа ко многим вспомогательным элементам. Рабочая среда MatLab содержит следующие элементы:

- панель инструментов с кнопками и раскрывающимся списком;
- окно с вкладками **Launch Pad** и **Workspace**, из которого можно получить доступ к различным модулям ToolBox и к содержимому рабочей среды;
- окно с вкладками **Command History** и **Current Directory**, предназначенное для просмотра и повторного вызова ранее введенных команд, а также для установки текущего каталога;
- командное окно, в котором находится приглашение к вводу » и мигающий вертикальный курсор;
- строку состояния.

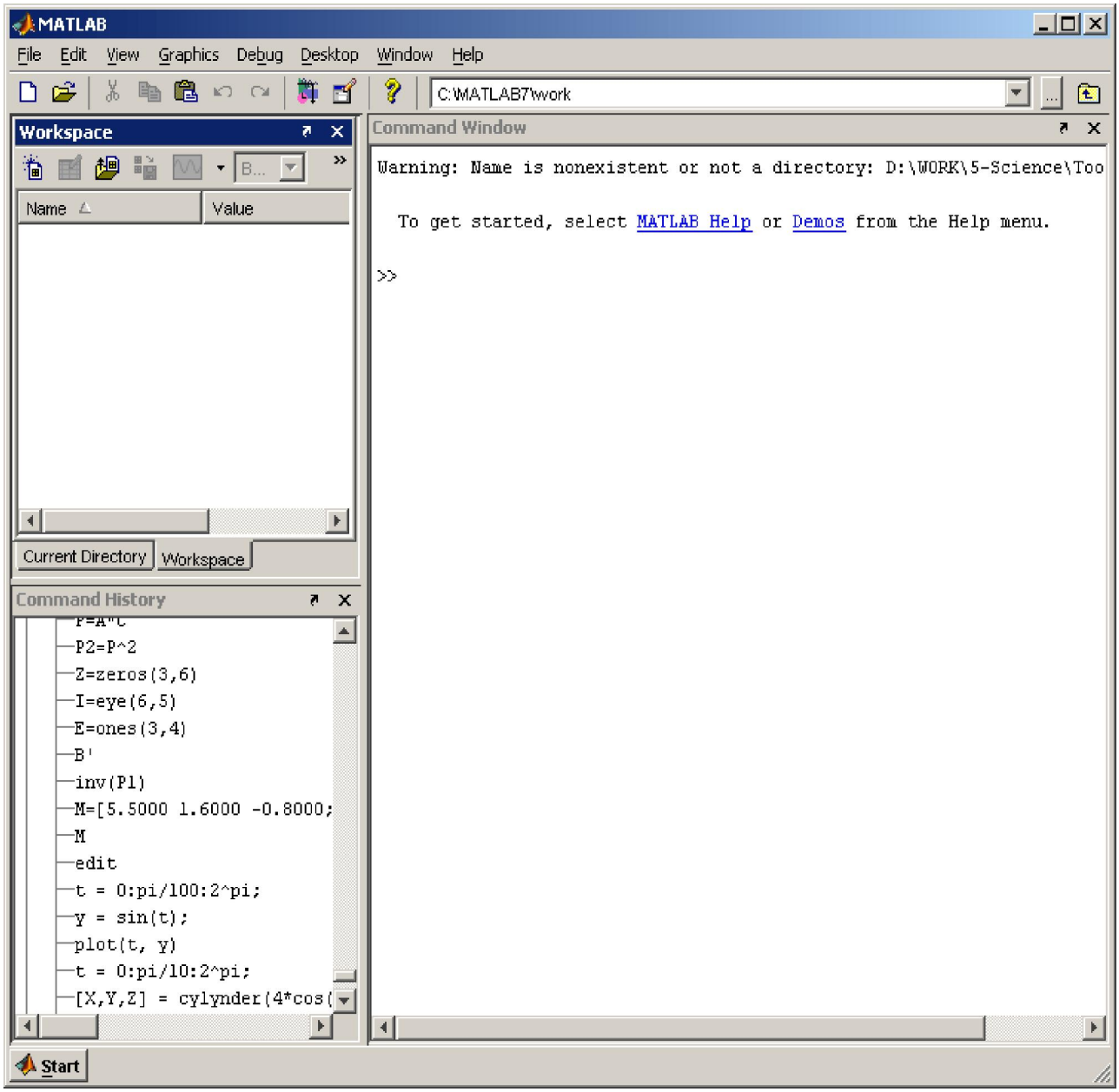


Рисунок 1 – Основное окно программы MATLAB

Если в рабочей среде MatLab отсутствуют некоторые окна, приведенные на рисунке, то следует в еню **View** выбратьсоответствующиепункты: **Command Window**,**Command History**, **Current Directory**, **Workspase**, **Launch Pad**.

Команды следует набирать в командном окне. Символ », обозначающий приглашение к вводу командной строки, набирать не нужно. Для просмотра рабочей области удобно использовать полосы скроллинга или клавиши **Home**, **End**, для перемещения влево или вправо,

и **PageUp**, **PageDown** для перемещения вверх или вниз. Если вдруг после перемещения по рабочей области командного окна пропала командная строка с мигающим курсором, просто нажмите **Enter**.

Важно помнить, что набор любой команды или выражения должен заканчиваться нажатием на **Enter**, для того, чтобы программа MatLab выполнила эту команду или вычислила выражение.

После открытия основного окна программы MATLAB нужно запустить программу Simulink. Это можно сделать одним из трех способов:

Нажать кнопку (Simulink) на панели инструментов командного окна MATLAB.

В командной строке главного окна MATLAB напечатать Simulink и нажать клавишу Enter на клавиатуре.

Выполнить команду Open в меню File и открыть файл модели (mdl - файл).

Последний вариант удобно использовать для запуска уже готовой и отлаженной модели, когда требуется лишь провести расчеты и не нужно добавлять новые блоки в модель. Использование первого и второго способов приводит к открытию окна обозревателя разделов библиотеки Simulink (рисунок 2).

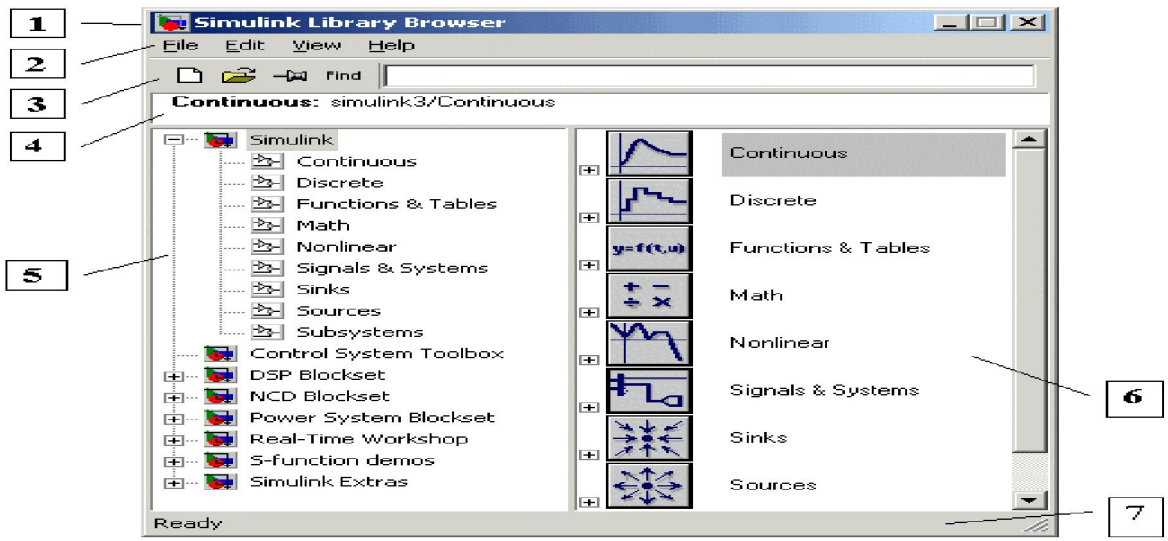


Рисунок 2 – Окно обозревателя разделов библиотеки Simulink

Обозреватель разделов библиотеки Simulink

Окно обозревателя библиотеки блоков содержит следующие элементы:

1. Заголовок, с названием окна – Simulink Library Browser.

2. Меню, скомандами File, Edit, View, Help.

3. Панель инструментов, с ярлыками наиболее часто используемых команд.

4. Окно комментария для вывода поясняющего сообщения о выбранном блоке.

5. Список разделов библиотеки, реализованный в виде дерева.

6. Окно содержимого раздела библиотеки (список вложенных разделов библиотеки или блоков).

7. Строка состояния, содержащая подсказку по выполняемому действию.

На рисунке 2 выделена основная библиотека Simulink (в левой части окна) и показаны ее разделы (в правой части окна).

Библиотека Simulink содержит следующие основные разделы:

Continuous – линейные блоки.

Discrete – дискретные блоки.

Functions & Tables – функцииитаблицы.

Math – блоки математических операций.

Nonlinear – нелинейные блоки.

Signals & Systems – сигналы и системы.

Sinks – регистрирующие устройства.

Sources — источники сигналов и воздействий.

Subsystems – блоки подсистем.

Список разделов библиотеки Simulink представлен в виде дерева, и правила работы с ним являются общими для списков такого вида:

Пиктограмма свернутого узла дерева содержит символ "+", а пиктограмма развернутого содержит символ "-".

Для того чтобы развернуть или свернуть узел дерева, достаточно щелкнуть на его пиктограмме левой клавишей мыши (ЛКМ).

При выборе соответствующего раздела библиотеки в правой части окна отображается его содержимое (рисунок 3).

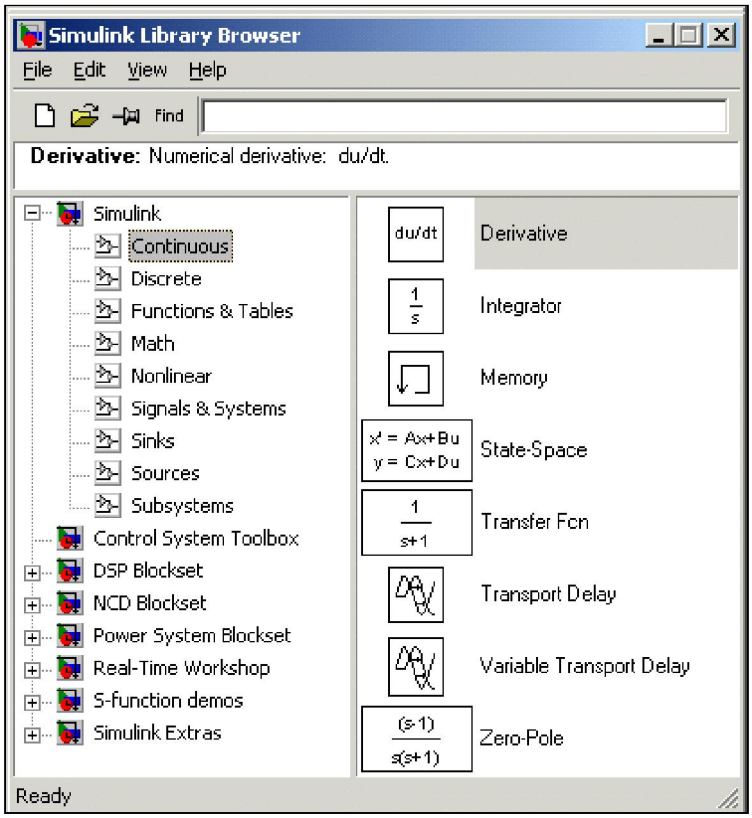


Рисунок 3 – Окно обозревателя с набором блоков раздела библиотеки

Для работы с окном используются команды собранные в меню. Меню обозревателя библиотек содержит следующие пункты:

File (Файл) — Работа с файлами библиотек.

Edit (Редактирование) — Добавление блоков и их поиск (по названию).

View (Вид) — Управление показом элементов интерфейса.

Help (Справка) — Вывод окна справки по обозревателю библиотек.

Для работы с обозревателем можно также использовать кнопки на панели инструментов (рисунок 4).

Рисунок 4 – Панель инструментов обозревателя разделов библиотек

Кнопки панели инструментов имеют следующее назначение:

- создать новую S-модель (открыть новое окно модели).
- открыть одну из существующих S-моделей.
- изменить свойства окна обозревателя. Данная кнопка позволяет установить режим отображения окна обозревателя "поверх всех окон". Повторное нажатие отменяет такой режим.

4 Поиск блока по названию (по первым символам названия). После того как блок будет найден, в окне обозревателя откроется соответствующий раздел библиотеки, а блок будет выделен. Если же блок с таким названием отсутствует, то в окне комментария будет выведено сообщение Not found <имя блока> (Блок не найден).

Перед выполнением имитационного моделирования необходимо предварительно задать параметры расчета. Задание параметров расчета выполняется в панели управления меню Configuraation Parameters. Вид панели управления приведен на рисунке 5.

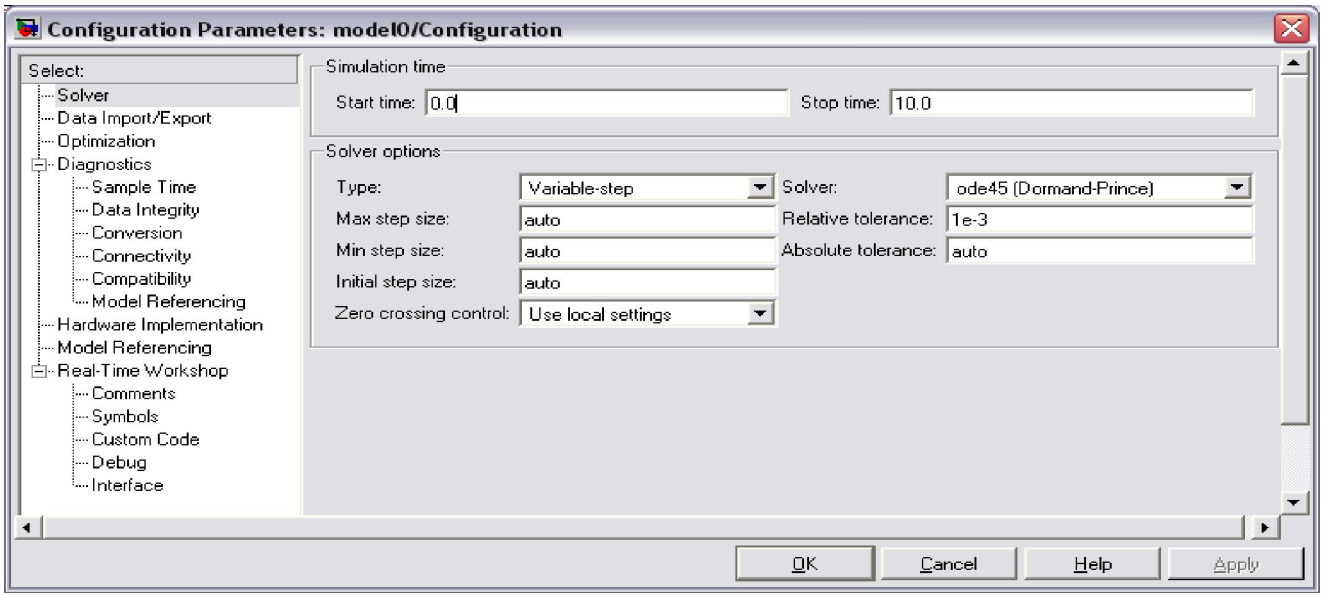


Рисунок 5 – Панель управления меню Configuraation Parameters

Окно настройки параметров расчета имеет 4 вкладки:

- Solver (Расчет) — Установка параметров расчета модели.
- Workspace I/O (Ввод/вывод данных в рабочую область) — Установка параметров обмена данными с рабочей областью MATLAB.
- Diagnostics (Диагностика) — Выбор параметров диагностического режима.
- Advanced (Дополнительно) — Установка дополнительных параметров.

Установка параметров расчета модели выполняется с помощью элементов управления, размещенных на вкладке Solver. Эти элементы разделены на три группы (Рисунок 5): Simulation time (Интервал моделирования или, иными словами, время расчета), Solver options (Параметры расчета), Output options (Параметры вывода).

Время расчета (Simulation time) задается указанием начального (Start time) и конечного (Stop time) значений времени расчета. Начальное время, как правило, задается равным нулю. Величина конечного времени задается пользователем исходя из условий решаемой задачи.

При выборе параметров расчета (Solver options) необходимо указать способ моделирования (Type) и метод расчета нового состояния системы. Для параметра Type доступны два варианта - с фиксированным (Fixed-step) или с переменным (Variable-step) шагом. Как правило, Variable-step используется для моделирования непрерывных систем, а Fixed-step - для дискретных.

Список методов расчета нового состояния системы содержит несколько вариантов. Первый вариант (discrete) используется для расчета дискретных систем. Остальные методы используются для расчета непрерывных систем. Эти методы различны для переменного (Variable-step) и для фиксированного (Fixed-step) шага времени, но, по сути, представляют собой процедуры решения систем дифференциальных уравнений.

Ниже двух раскрывающихся списков Type находится область, содержимое которой меняется зависимости от выбранного способа изменения модельного времени. При выборе Fixed-step в данной области появляется текстовое поле Fixed-step size (величина фиксированного шага) позволяющее указывать величину шага моделирования. Величина шага моделирования по умолчанию устанавливается системой автоматически (auto). Требуемая величина шага может быть введена вместо значения auto либо в форме числа, либо в виде вычисляемого выражения (то же самое относится и ко всем параметрам, устанавливаемым системой автоматически).

При выборе Fixed-step необходимо также задать режим расчета (Mode). Для параметра Mode доступны три варианта:

- MultiTasking (Многозадачный) – необходимо использовать, если в модели присутствуют параллельно работающие подсистемы, и результат работы модели зависит от временных параметров этих подсистем. Режим позволяет выявить несоответствие скорости и дискретности сигналов, пересылаемых блоками друг другу.
- SingleTasking (Однозадачный) - используется для тех моделей, в которых недостаточно строгая синхронизация работы отдельных составляющих не влияет на конечный результат моделирования.
- Auto (Автоматический выбор режима) - позволяет Simulink автоматически устанавливать режим MultiTasking для тех моделей, в которых используются блоки с различными скоростями передачи сигналов и режим SingleTasking для моделей, в которых содержатся блоки, оперирующие одинаковыми скоростями.

При выборе Variable-step в области появляются поля для установки трех параметров:

- Max step size - максимальный шаг расчета. По умолчанию он устанавливается автоматически (auto) и его значение в этом случае

равно (StopTime — StartTime)/50. Довольно часто это значение оказывается слишком большим, и наблюдаемые графики представляют собой ломаные (а не плавные) линии. В этом случае величину максимального шага расчета необходимо задавать явным образом.

- Min step size - минимальный шаг расчета.
- Initial step size - начальное значение шага моделирования.

В нижней части вкладки Solver задаются настройки параметров вывода выходных сигналов моделируемой системы (Output options). Для данного параметра возможен выбор одного из трех вариантов:

- Refine output (Скорректированный вывод) – позволяет изменять дискретность регистрации модельного времени и тех сигналов, которые сохраняются в рабочей области MATLAB с помощью блока To Workspace. Установка величины дискретности выполняется в строке редактирования Refine factor, расположенной справа. По умолчанию значение Refine factor равно 1, это означает, что регистрация производится с шагом Dt = 1.
- Produce additional output (Дополнительный вывод) — обеспечивает дополнительную регистрацию параметров модели в заданные моменты времени; их значения вводятся в строке редактирования (в этом случае она называется Output times) в виде списка, заключенного в квадратные скобки. При использовании этого варианта базовый шаг регистрации (Dt) равен 1.
- Produce specified output only (Формировать только заданный вывод) — устанавливает вывод параметров модели только в заданные моменты времени, которые указываются в поле Output times (Моменты времени вывода).

Запуск расчета (моделирование) выполняется с помощью выбора пункта меню Simulation/Start. или инструмента ▶ на панели инструментов. Процесс

расчета можно завершить досрочно, выбрав пункт меню Simulation/Stop или инструмент ■. Расчет также можно остановить (Simulation/Pause) и затем продолжить (Simulation/Continue).

К возможности изменения свойств модели прибегают обычно только опытные пользователи обычно бывает достаточно установок свойств по умолчанию.

Подготовка и запуск модели

- Создание модели
- Моделирование ограничителя
- Основные приемы подготовки и редактирования модели
- Операции форматирования модели.

Создание модели

Постановка задачи и начало создания модели

Решение любой проблемы в системе Simulink должно начинаться с постановки задачи. Чем глубже продумана постановка задачи, тем больше вероятность успешного ее решения. В ходе постановки задачи нужно оценить, насколько суть задачи отвечает возможностям пакета Simulink какие компоненты последнего могут использоваться для построения модели.

Основные команды редактирования модели сосредоточены в меню Edit. В качестве примера применения этих команд рассмотрим построение простой модели, а точнее, сразу трех простых моделей в пределах одного окна, можно одновременно моделировать несколько систем.

Сначала откроем пустое окно для новой модели (кнопка Createanewmodelв панели инструментов браузера библиотек Simulink).

Ввод текстовой надписи

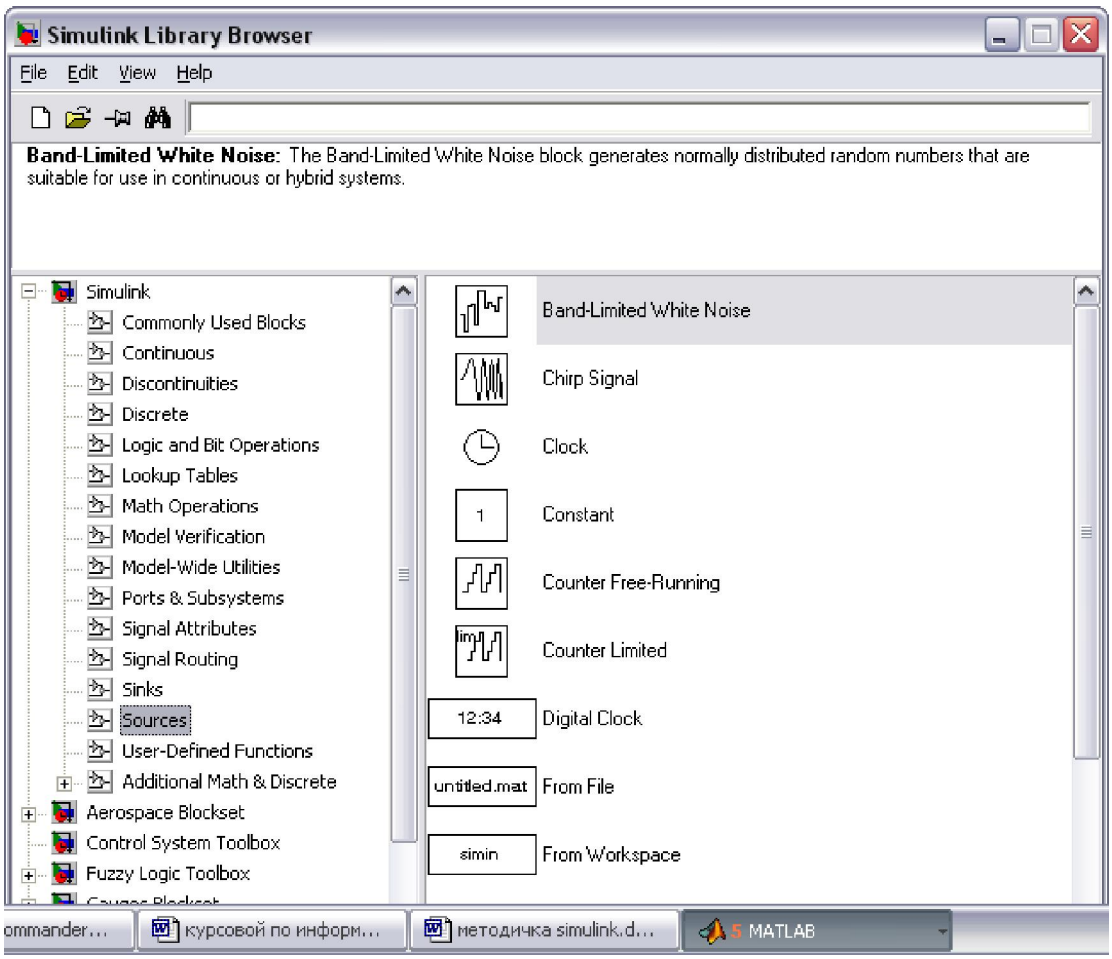
Введем заголовок нашей будущей модели - «Simplemodel» (Простая модель). Для этого достаточно установить курсор мыши в нужное место окна и

дважды щелкнуть левой кнопкой мыши. Появится прямоугольная рамка, внутри которой находится мигающий маркер ввода в виде вертикальной палочки.

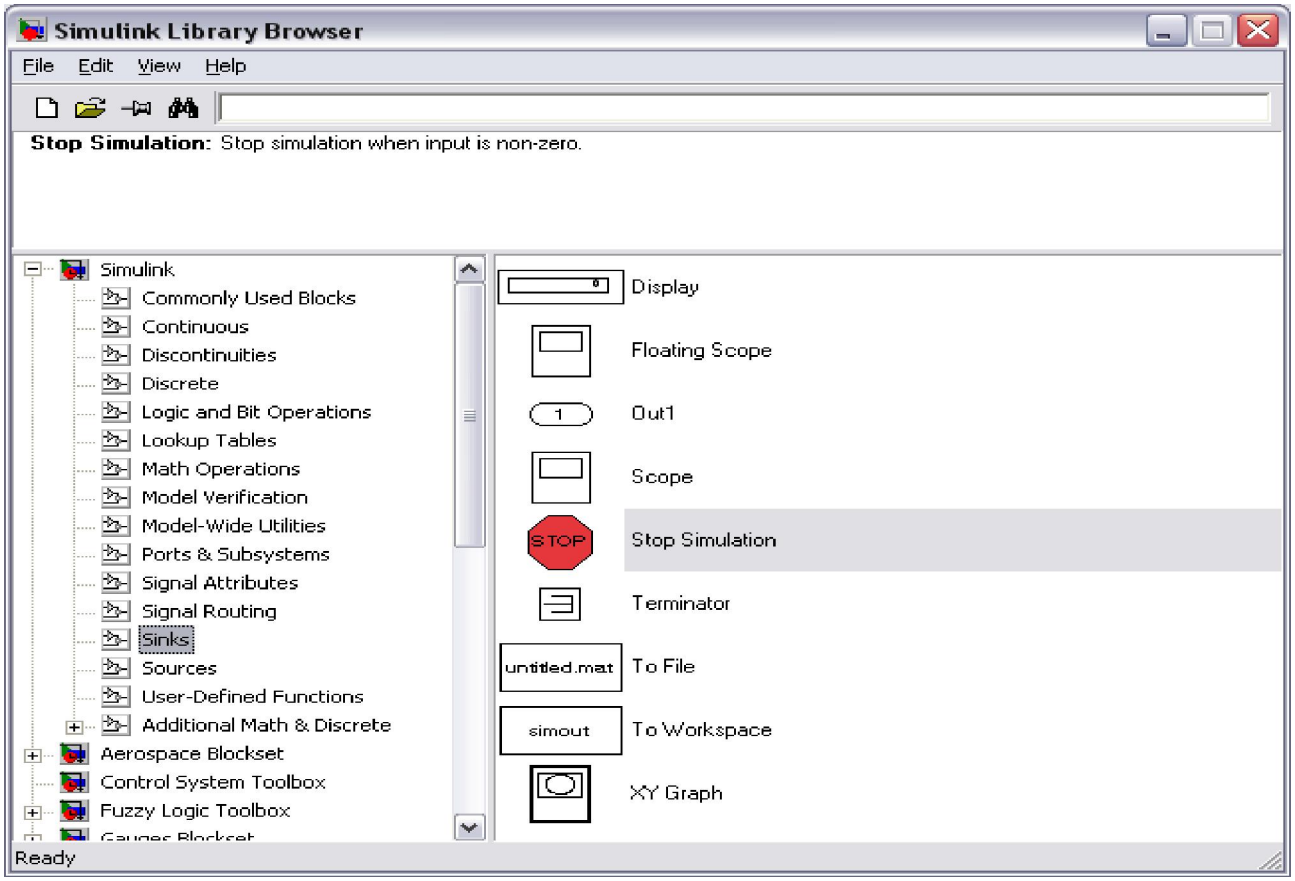
Теперь можно ввести нужную надпись по правилам, действующим для строчного редактора. Пока будем считать, что параметры надписи по умолчанию нас вполне устраивают.

Размещение блоков в окне модели

Из раздела библиотеки Sources перетащим мышью три источника сигнала: синусоидального, прямоугольного (дискретного) и пилообразного.



Затем из раздела Sinks перетащим в окно модели блок осциллографа.



В результате получим модель в виде, представленном на рисунке 6.

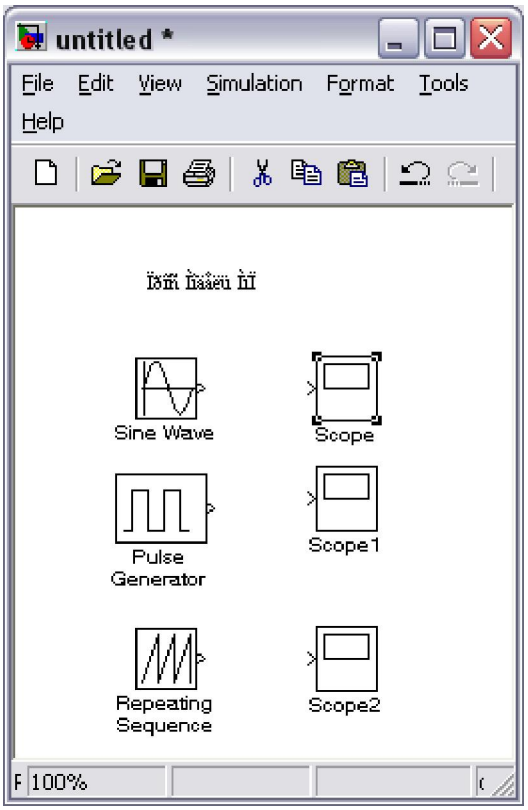


Рисунок 6 – Начало подготовки модели

Выделение блока модели

На рисунке 6 показано также меню редактирования Edit в открытом виде – при выделении блока в этом меню становятся доступны команды редактирования свойств блока. Для выделения блока достаточно навести на него маркер мыши и нажать левую кнопку. В рамке блока по углам появятся маленькие темные прямоугольники, которые и являются признаком того, что блок выделен. На рисунке 6 выделен блок осциллографа Scope.

Если захватить курсором мыши уголок выделенного блока, то можно заметить, что курсор мыши превратится в перекрестие тонких диагональных двухсторонних стрелок. Это означает, что можно пропорционально увеличивать или уменьшать блок в диагональных направлениях.

Меню редактирования Edit

Кратко рассмотрим основные команды меню Edit(рисунок 6). Это меню содержит ряд типовых команд, которые разбиты на 6 групп. В первой группе есть две команды:Undo(отмена последней операции) и Redo(восстановление последней отмененной операции). Эти команды являются контекстно-зависимыми.

Следующая группа команд связана с операциями с буфером обмена Windows:

- Cut - перенос выделенных объектов в буфер;
- Copy - копирование выделенных объектов в буфер;
- Paste - вставка объектов из буфера в заданное курсором мыши место;
- Clear - уничтожение выделенных объектов;
- SelectAll - выделение всех объектов модели;
- Copymodeltoclipboard - копирование всей модели в буфер;
- Find - поиск в модели заданного объекта.

Остальные команды подменю Edit носят специальный характер.

Теперь можно приступить к соединению выходов источников со входами осциллографов. Для этого достаточно указать курсором мыши на начало соединения (выход источника) и затем при нажатой левой кнопке мыши протянуть соединение в его конец (вход осциллографа). В итоге получим модель, показанную на рисунке. 7.

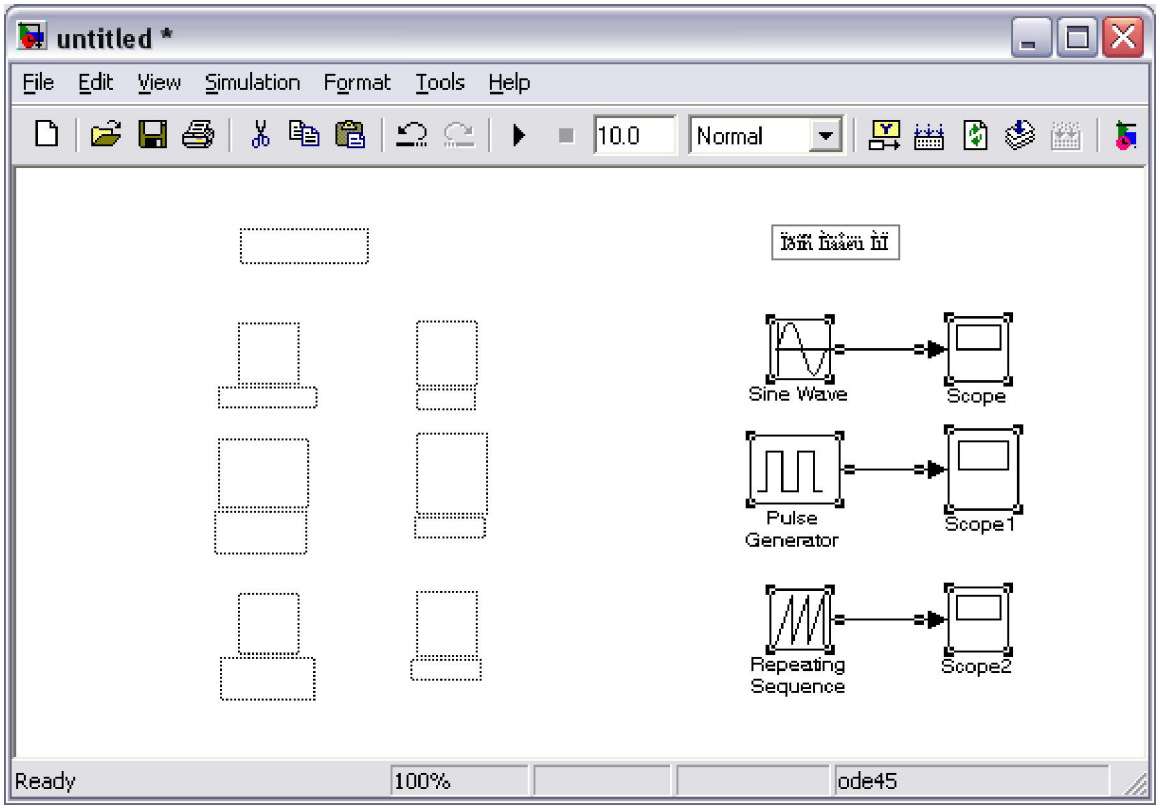


Рисунок 7 – Готовая модель

Порядок выполнения работы

1. Запустить Simulink.

2. Изучить основную библиотеку Simulink. Открыть и изучить все разделы библиотеки.

3. В библиотеке Simulink вызвать DEMOS.

4. Изучить простые модели:

- трассировка прыгающего шара;
- моделирование маятника.

Требования к отчету: Отчет должен содержать краткие общие сведения, схему одной Simulink модели, график или рисунок результатов моделирования, выводы.

Контрольные вопросы

1. Расскажите об основных разделах библиотеки.
2. Что входит в раздел Continuous?
3. Что входит в раздел Discrete?
4. Что входит в раздел Functions&Tables?
5. Что входит в раздел Math?
6. Что входит в раздел Signals & Systems?
7. Что входит в раздел Sinks?
8. Что входит в раздел Sources?
9. Какие результаты получены при моделировании прыгающего шара?
10. Какие результаты получены при моделировании.

2 Лабораторная работа №2. Математические модели на основе обыкновенных дифференциальных уравнений (ОДУ)

Цель работы: научиться составлять схемы решения систем обыкновенных дифференциальных уравнений (ОДУ) в среде Simulink пакета MatLab.

1. Решение ОДУ первого порядка.

Основой для решения обыкновенных дифференциальных уравнений первого порядка является задача Коши:

$$\frac{dy}{dx} = f(x, y)$$

с одной зависимой переменной y(x).

Пример.

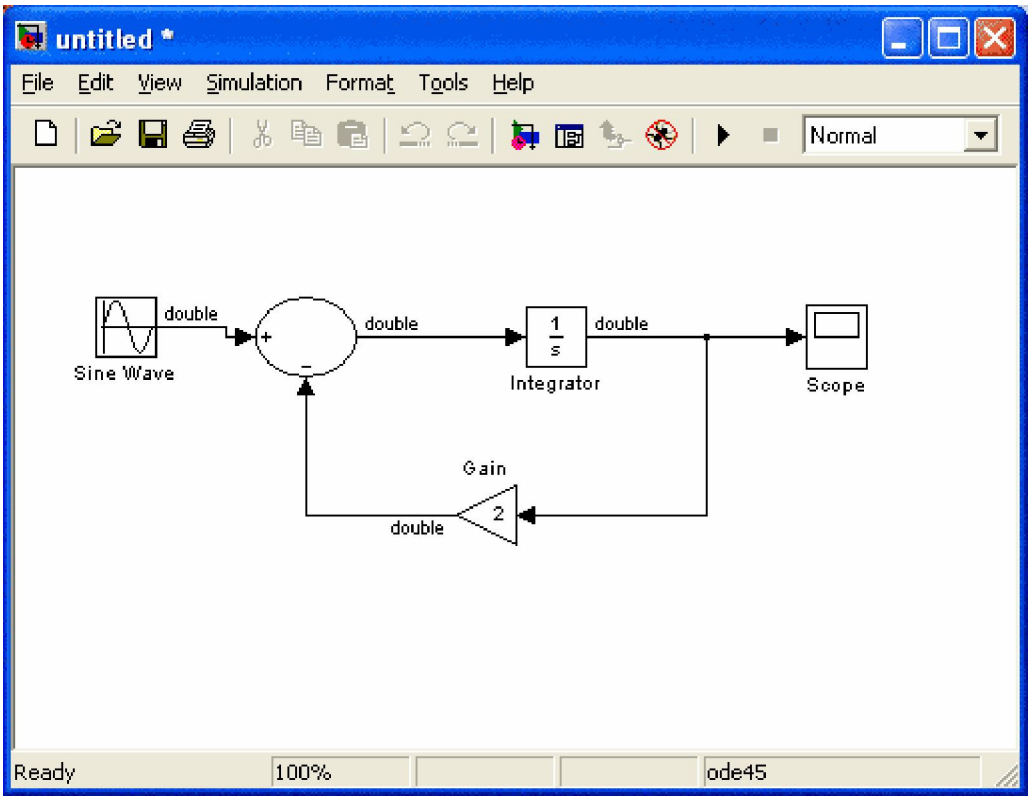
Дано дифференциальное уравнение

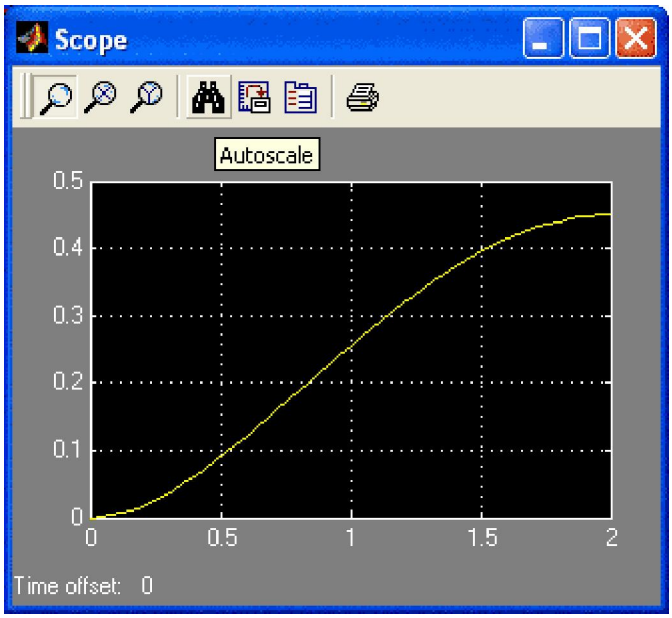
x'(t) + 2x(t) = sin(t),

x(0) = 0.

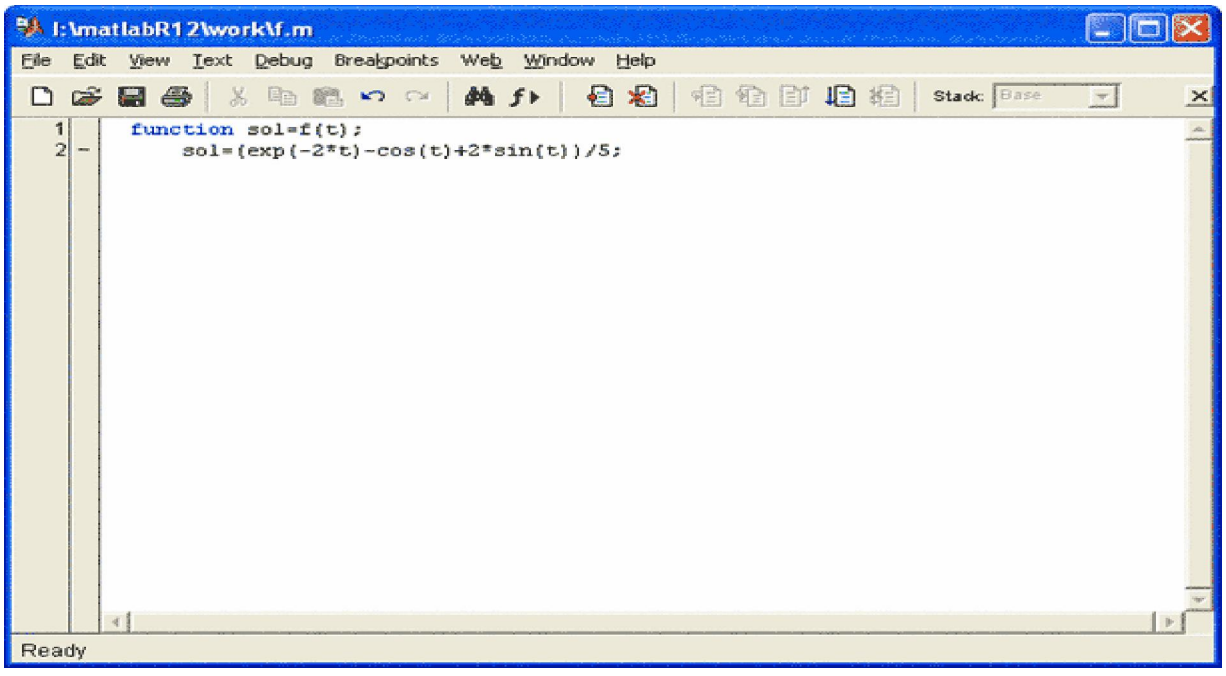
После запуска системы MatLab нажмем кнопку Simulink, а затем в открывшемся окне кнопку Create a new Model. В открывшемся файле создадим схему решения уравнения, перетаскивая при нажатой левой кнопки мыши необходимые блоки из окна Simulink Library Brouser.

Для построения схемы решения уравнения в Simulink используется блок Integrator (класс Continuos). На его вход подается производная, а на выходе получают величину x. Блоки Sum (Сумматор) и Gain (Усилитель) (класс Math) необходимы для формирования значения x' в соотствии с ОДУ. Для получения сигнала sin(t) используется блок Sine Wave (класс Sources), в котором необходимо провести установки, соответствующие задаче, открыв блок двойным щелчком мыши или выбрав опцию Block Parameters при нажатой правой кнопке мыши. Полученное значение x(t) подается на вход блока Scope. При открытии данного блока появляется график решения. Установить масштабы осей, соответствующие полученному решению можно, нажав кнопку Autoscale.





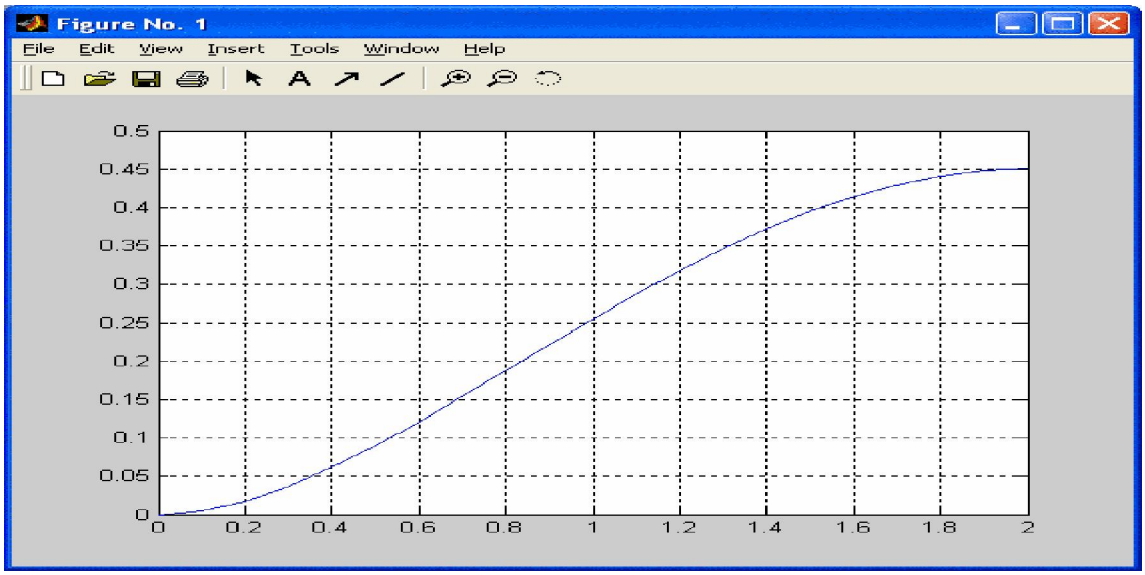
Для проверки найденного решения в окне Command Window создадим M - файл для решения задачи (File ==> New ==> M-file). В открывшемся окне создадим функцию решения задачи, которую сохраним в текущей директории под именем f.m (указанное имя система предлагает по умолчанию).



После этого в командном окне наберем текст:

```
>> t=(0:0.1:2);
>> y=f(t);
>> plot(t,y)
>> grid on
```

После выполнения команд открывается окно с графиком функции. Два полученных графика идентичны.



2. Решение систем ОДУ первого порядка.

Рассмотрим решение системы ОДУ первого порядка.

Пример.

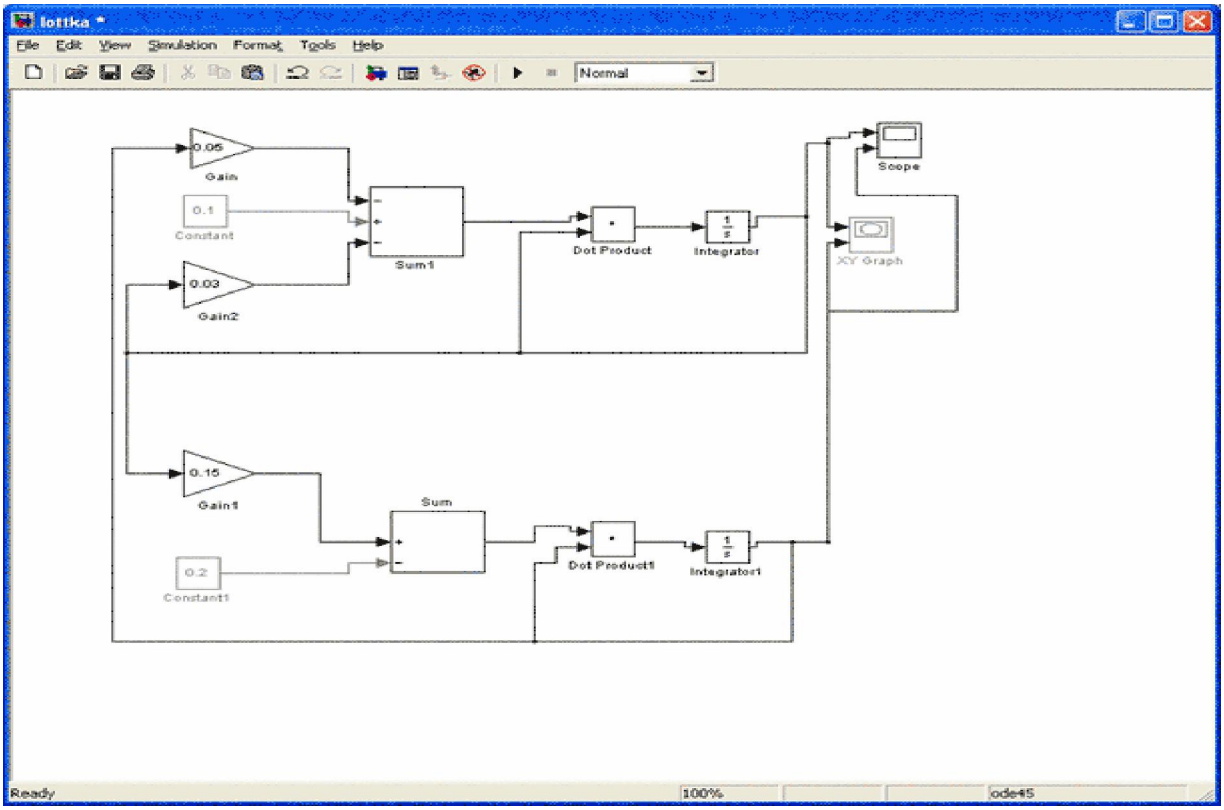
Модифицированная задача Лотки – Вольтера (модель хищник-жертва).

С учетом самоограничения на рост популяции жертв

$$\begin{cases} \frac{dx}{dt} = x(\alpha - \beta y - \gamma x), \\ \frac{dy}{dt} = -y(\delta - \varepsilon x). \end{cases}$$

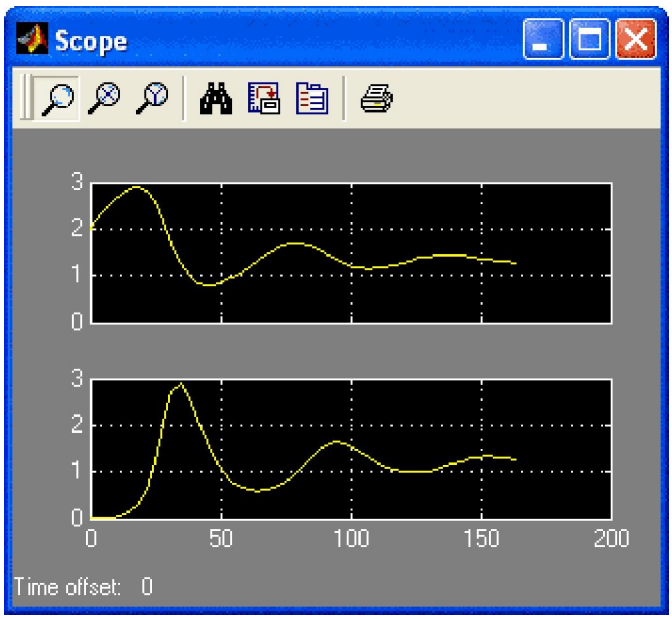
Зададим параметры задачи: $\alpha = 0.1$; $\beta = 0.05$; $\gamma = 0.03$; $\delta = 0.2$; $\varepsilon = 0.15$.

Блок-схема решения задачи в системе Simulink:



Раскрыв блок интегратора, зададим начальные значения: $y_1 = 2$, $y_2 = 0.01$.

После окончания моделирования, раскрывая блоки Scope и XY-Graph, можно увидеть графики изменения численности и фазовый портрет решения системы:



Создадим также в MatLab M - файл для задания правой части системы ОДУ:

```
function dy=vlm(t,y)
dy=zeros(2,1);
dy(1)=y(1)*(0.1-0.05*y(2)-0.03*y(1));
dy(2)=-y(2)*(0.2-0.15*y(1));
```

Для конкретного набора начальных значений систему можно решить, используя метод Рунге - Кутты 4-го порядка (встроенная функция ode45):

```
> [T, Y] = ode45('vlm',[0 164],[2 0.01]);
```

Параметры функции ode45: имя M - файла, диапазон изменения независимой переменной, начальные значения.

Для построения фазового портрета можно использовать функцию plot:

```
> plot(Y(:,1),Y(:,2));
>axis[0 4 0 4]
```

Задание:

1. Построить схемы решения рассмотренных задач в системе Simulink, получить графики решения. Сравнить с решением задач в MatLab с помощью функции ode45.
2. Решить эти же задачи в MatLab, построить графики решения.
3. Построить схему решения в Simulink и получить графики решения следующих задач:

1) $\begin{cases} y' = \dfrac{z}{x}, \\ z' = \dfrac{2z^2}{x(y-1)} + \dfrac{z}{x}, \\ y(1) = 0, \quad z(1) = \dfrac{1}{3} \end{cases}$ на $[1,2]$.

2) $\begin{cases} y' = (z - y)x, \\ z' = (z + y)x, \\ y(0) = 1, \quad z(0) = 1 \end{cases}$ на $[0,1]$.

3) $\begin{cases} y' = \cos(y + 2z) + 2, \\ z' = \dfrac{2}{x + 2y^2} + x + 1, \\ y(0) = 1, \quad z(0) = 0.05 \end{cases}$ на $[0, 0.3]$.

4) $\begin{cases} y' = e^{-(x^2+z^2)} + 2x, \\ z' = 2y^2 + z, \\ y(0) = 0.5, \quad z(0) = 1 \end{cases}$ на $[0, 0.3]$.

5) $y'' = -\dfrac{y'}{x} + \dfrac{y}{x^2} + 1$, $y(3) = 6$, $y'(3) = 3$.

6) $y'' - y = e^x$, $y(0) = 0$, $y'(0) = 0.5$ на $[0,1]$.

7) $y'' - 2y' = x^2 - 1$, $y(1) = -1/6$, $y'(1) = -3/4$ на $[1,2]$.

8) $y'' - 2y' = 3e^x$, $y(0.3) = 1.415$, $y'(0) = 5.83$ на $[0.3, 0.6]$.

9) $y'' + y' = 3x^2$, $y(1) = -1$, $y'(1) = 2$ на $[1,2]$.

Сравнить с решением задач в MatLab с помощью функции ode45.

Контрольные вопросы

1. Почему при решении уравнения необходимо определять константу

2. Почему уравнение Коши называют обыкновенным дифференциальным уравннием

3. Как повысить точность решения уравнения Эйлера

4. Для какой цели используются функции ode45 в MATLAB simulink

5. Какие условия модели “хищник-жертва”

Содержание отчета

1. Цель работы.
2. Ход работы.
3. Визуальная модель решения варианта задачи.
4. Графики полученных зависимостей.
5. Выводы.
6. Ответы на вопросы.

Лабораторная работа № 3. Модель канала связи

Цель работы состоит в изучении моделирования канала связи на основе потерь энергии при передаче сигналов определенной формы по этому каналу.

Теоретические сведения

Использование информации потребителей связано с ее транспортировкой от источника. Транспортировка всегда осуществляется в пространстве и во времени. Основные требования к каналам связи заключаются в следующем:

1. Высокая эффективность передачи, обуславливаемая пропускной способностью канала.
2. Надежная передача данных путем использования помехоустойчивого кодирования.
3. Использование перспективных (в основном оптических) каналов связи.

Для повышения эффективности использования каналов связи широко применяется частотное разделение каналов. При этом для различных каналов отводятся непересекающиеся полосы частот Δf_1, Δf_2,…, Δf_n, на

частотной шкале. Спектры сигналов соответствующих каналов должны укладываться в пределы Δf_1. Поэтому необходимо изучать спектры сигналов и оценивать какое количество энергии сигналов теряется при несоблюдении ширины спектра сигнала и канала связи.

Информационная модель канала связи состоит из источника информации, передающей оконечной аппаратуры для формирования сигнала с модуляцией, преобразователя кода, канала связи, приемника, декодера, приемной оконечной аппаратуры, потребителя информации.

Наиболее перспективно в качестве линии передачи использовать оптический кабель, по которому может передаваться сигнал. Состоит из сердечника, оболочки, и внешнего покрытия. Сердечник изготовляют из материала с наименьшими потерями, в оболочке допускаются большие потери.

Процент передачи излучения по кабелю осуществляется в результате отражения света от оболочки, имеющий больший коэффициент отражения, чем сердечник.

Рассмотренная выше структура канала является двухточечной (т.е. канала соединяет только два узла). Существует два больших класса двухточечных каналов: цифровые и аналоговые. При модульном подходе цифровой канал является просто битовым трактом с битовыми потоками на входе и выходе. Что касается аналогового сигнала, то на его вход поступает непрерывный сигнал, т.е. произвольная функция либо времени, либо пространства.

При изучении сетей передачи данных основное внимание уделяется аналоговым каналам, хотя и цифровые каналы важны для практики и более перспективны, но их лучше всего можно понять на основе изучения аналоговых каналов.

Для изучения каналов связи лучше всего применять спектральные методы, которые используют частотное представление функции в виде преобразования Фурье во временной или пространственной форме.

При рассмотрении спектров основных видов сигналов главное внимание уделяется определению их ширины, поскольку в основном этот фактор используется для согласования сигнала с аппаратурой обработки информации, причем для исключения потерь информации ширина спектра сигнала не должна превышать полосы пропускания канала.

Для периодического сигнала $U_t(t)$ спектр определяется соотношением

$$A_K = \frac{2}{T} \int_{\frac{-T}{2}}^{\frac{T}{2}} U_t(t) * e^{-jk\omega_t t} dt ,$$

где к = ±1, ±2, ±3, …;

$$\omega_t = \frac{2\pi}{T}$$ – круговая частота.

Спектр периодического сигнала имеет дискретный характер.

Для непериодического сигнала, определяемого на бесконечном интервале времени, преобразование Фурье имеет вид:

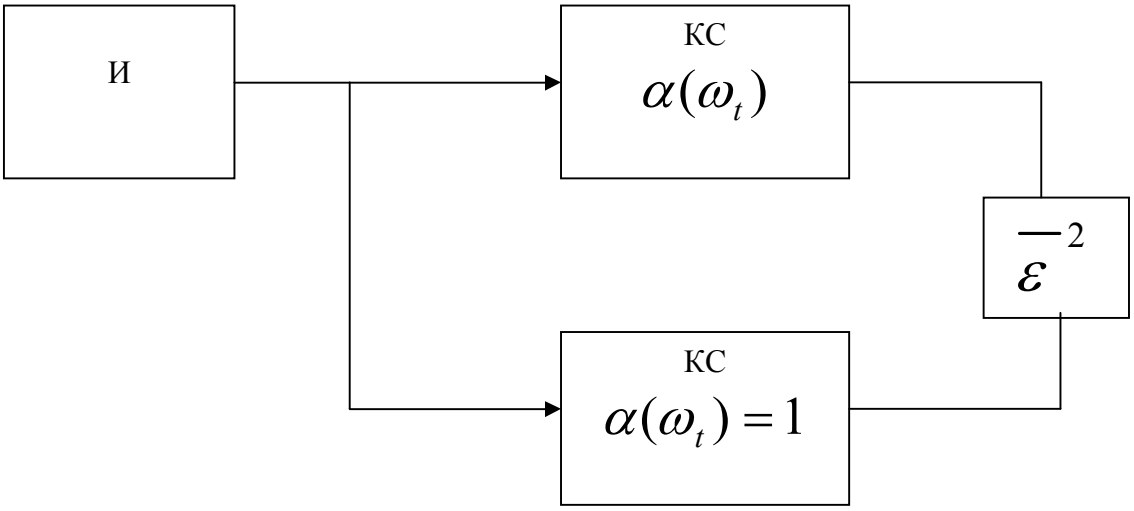
$$W(\omega_t) = \int_{-\infty}^{\infty} U(t) e^{-j\omega t} dt ,$$

$$U(t) = \frac{1}{2\pi} \int_{-\infty}^{\infty} W(j\omega) e^{j\omega t} d\omega .$$

В случае не периодических сигналов рассматривается не спектр сигнала, а его производная функция по ω_t = W(jω), носящего название спектральной плотности. Поэтому спектр непериодической функции имеет непрерывный характер.

Для исследования качества передачи канала связи наиболее целесообразно использовать критерий в виде потерь энергии сигнала.

Для анализа удобно пользоваться следующей моделью:



И – источник, формирующий сигнал U(t),

$\alpha(\omega_t)$ – передаточная характеристика канала,

ε^2– потери энергии сигнала связи.

Для оценки потерь энергии сигнала необходимо найти разность энергий сигналов, прошедших реальный канал с передаточной характеристикой $\alpha(\omega_t)$ и идеальный канал с $\alpha(\omega_t) = 1$. Тогда энергия потерь ε^2 будет иметь вид:

$$\overline{\varepsilon}^2 = \frac{1}{2\pi}\int_{-\infty}^{\infty} |W(\omega_t)|^2 * d\omega_t - \frac{1}{2\pi}\int_{-\infty}^{\infty} |W(\omega_t)|^2 * |\alpha(\omega_t)|^2 \, d\omega_t =$$

$$= \frac{1}{2\pi}\int_{-\infty}^{\infty} |W(\omega_t)|^2 * (1 - |\alpha(\omega_t)|^2) d\omega_t,$$

где $W(\omega_t)$ – спектральная функция сигнала,

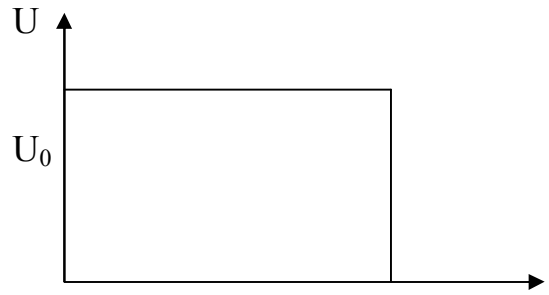
$\alpha(\omega_t)$ – передаточная характеристика канала,

$\overline{\varepsilon}^2$ _ потери энергии, усредненные по всем частотам.

Сигналы и их спектральные функции

1. Прямоугольный сигнал

Вид сигнала



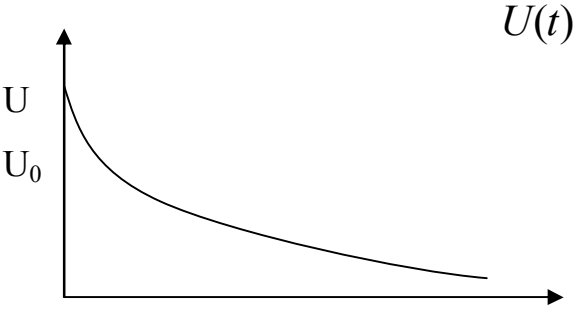
$$U(t)=\begin{cases} U_0, \text{ при } 0 < t < T_i \\ 0, \text{ для остальных } T_i \end{cases}$$

$T_i t$

$$W(\omega_t) = U_0 T_i * \frac{\sin\frac{\omega_t T}{2}}{\frac{\omega_w T}{2}}$$

2. Экспоненциальный сигнал

$$U(t) = U_0 e^{-pt},$$

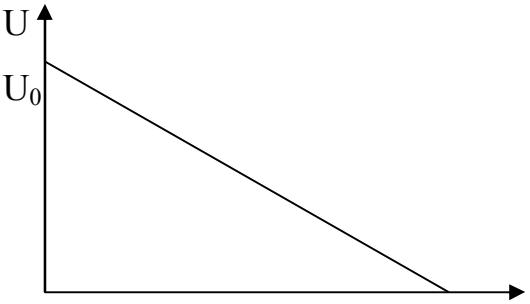


где p - параметр, характеризующий вид экспоненты.

T t

$$|W(\omega_t T)| = U_0 T \sqrt{\frac{(1-e^{-pt}\cos(\omega_t T))^2 + (e^{-pt}\sin(\omega_t T))^2}{(pT)^2 + (\omega_t T)^2}}$$

3. Линейный сигнал

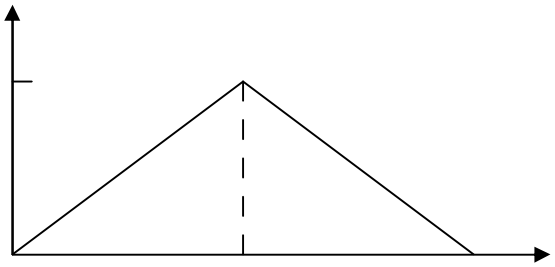


$$U(t) = U_0(1-\frac{t}{T_H})$$

$T_H t$

$$|W(\omega_t T)| = \frac{U_0}{\omega_t T}\sqrt{(\cos(\frac{\omega_t T}{2}) - \frac{\sin(\frac{\omega_t T}{2})}{\frac{\omega_t T}{2}})^2 + (\sin(\frac{\omega_t T}{2}))^2}$$

4. Треугольный сигнал



$$U(t) = \begin{cases} U_0 \frac{t}{T}, & \text{при } t<M \\ U_0 & \\ U_0(1-\frac{t-M}{T-M}), & t>M \end{cases}$$

для M =T/2

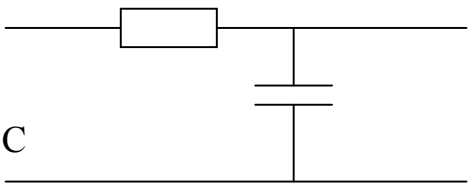
M T t

$$W(\omega_t T) = \frac{U_0}{\omega_t}\frac{\sin(\frac{\omega_t T}{4})}{\frac{\omega_t T}{4}}\sqrt{2-2\cos(\frac{\omega_t T}{2})}.$$

Передаточные характеристики канала

Функцию пропускания канала наиболее просто описывать интегральной RC-цепью.

R



C

В этом случае передаточная характеристика имеет вид

$$\alpha(\omega_t) = \frac{1}{1+\omega_t^2 T^2}, \text{ где } \quad T = RC.$$

Энергия сигнала на выходе канала

Энергия сигнала определяется соотношением

$$E^2 = \frac{1}{2\pi}\int_{-\infty}^{\infty} |\, W(\omega_t)\,|^2 \; d\omega_t .$$

Задания для лабораторной работы

1. Получить номер задания (вид сигнала) у преподавателя.
2. Найти спектральную функцию для этого сигнала $W(\omega_t)$.
3. Выбрать передаточную функцию канала $\alpha(\omega_t)$.
4. Найти энергию сигнала на выходе реального и идеального каналов.
5. Найти количество энергии, которое теряется при несовпадении полосы частот сигнала и канала.
6. Построить зависимости $W(\omega_t)$, $\alpha(\omega_t)$, $\varepsilon^2(\omega_t)$.
7. Ответить на вопросы.
8. Подготовить отчет по всем пунктам задания.

Вопросы

1. Что такое спектральная функция сигнала?
2. Почему спектральный метод наиболее удобно использовать при анализе линейных систем?
3. Что означает передаточная функция канала связи?
4. Каким образом осуществляется частотное уплотнение каналов?
5. Каким образом осуществляется временное уплотнение каналов?
6. Чем отличается структура аналогового и цифрового каналов связи?
7. В чем достоинство критерия потерь количества информации от критерия отношения сигнал/шум?

Литература

1. Цымбал В.М. Теория информации и кодирования. К., Наукова думка, 1982.

2. Д. Бертсекас, Р.Галлагер. Сети передачи данных. М.:Мир, 1989.

Лабораторная работа 4. Мультиагентная модель диффузии инноваций на рынках ИКТ

Цель работы: изучить методологию моделирования диффузии (распространения) инноваций. Приобрести практические навыки работы с системой AnyLogic при построении агентных моделей распространения диффузий инноваций в виде программного продукта.

Порядок выполнения работы. В данной лабораторной работе предлагается изучить агентный подход моделирования сложных систем в виде рынка информационно-коммуникационных технологий. В ходе работы нужно создать классическую модель распространения (диффузии) инноваций и те ее расширения, которые демонстрируют возможности AnyLogic для создания агентных моделей.

Теоретические сведения

За последние полвека накоплен достаточно большой опыт экономико-математического моделирования процесса распространения инновационных продуктов на рынке информационно-коммуникационных технологий.

Схема движения потребителей инновационного продукта между сегментами рынка ИКТ, представлена на рис. 1, на котором

T(t) — суммарное число индивидов на рынке в момент времени t;

M(t) — суммарное число потенциальных потребителей инновационного продукта на рынке в момент времени t;

N(t) — суммарное число действующих потребителей инновационного продукта на рынке в момент времени t;

$m(t)=\frac{dM(t)}{dt}$ - число индивидов, переходящих за бесконечно малый промежуток времени dt с неохваченного рынка на потенциальный;

$n(t) = \frac{dN(t)}{dt}$ - скорость распространения инновации — число индивидов, переходящих за бесконечно малый промежуток времени dt с потенциального рынка на охваченный. В теории инноваций под диффузией инноваций понимается решение N = N(t) задачи Коши для дифференциального уравнения

$$\frac{dN}{dt} = f(t, N(t))$$

с начальным условием N(0) = N0,

где t — время;

N(t) — объем распространения инновации к моменту t, который определяется обычно количеством проданных экземпляров или количеством действующих потребителей инновационного продукта;

ft,N(t) — функция, определяющая формулой диффузионной кривой и отражающая определенные предположения о природе процесса распространения инновации.

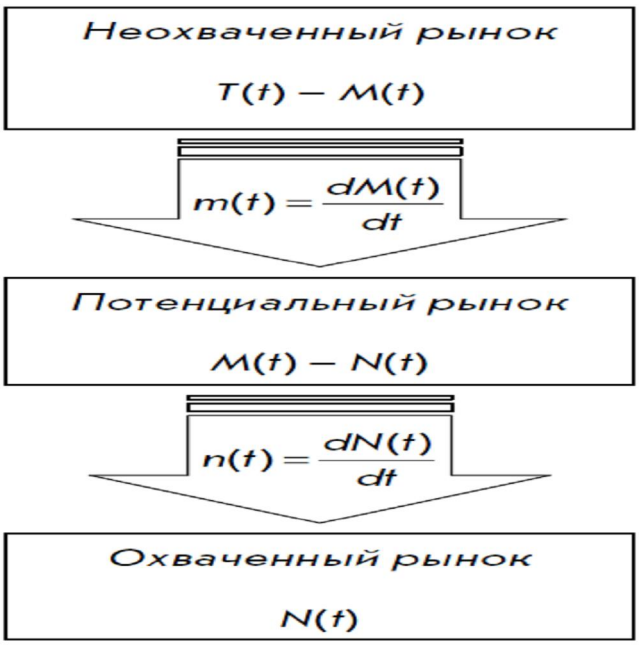


Рис.1 – Перемещение потребителей между сегментами инновационного рынка ИКТ

При этом предполагается обычно, что функция N(t) непрерывна и дифференцируема при всех неотрицательных t, а функция ft,N(t) унимодальна.

Базовая модель диффузии инноваций представляется следующим образом:

$$\frac{dN(t)}{dt} = g\big(t, N(t)\big)\big(M - N(t)\big)$$

В этой модели общее число потенциальных потребителей инновации M неизменно во времени, а скорость распространения инновации dN(t)/dt в каждый момент времени пропорциональна объему потенциального рынка M – N(t).

По мере увеличения общего числа действующих потребителей инновационного продукта N(t) и соответственно уменьшения числа потенциальных потребителей M – N(t) скорость распространения инновации снижается.

Функция g(t,N(t)) в модели (1.3.2) называется в теории инноваций скоростью адаптации, обычно интерпретируется как вероятность того, что потенциальный потребитель инновационного продукта приобретет его в момент t, и считается линейной функцией N(t):

$$g\big(t, N(t)\big) = a + bN(t)$$

Подстановка скорости адаптации (1.3.3) в базовую модель диффузии инноваций (1.3.2) дает следующее обыкновенное дифференциальное уравнение фундаментальной модели диффузии инноваций:

$$\frac{dN}{dt} = (a + bN)(M - N)$$

Параметры a и b в фундаментальной модели диффузии инноваций отражают соответственно степень внешних и внутренних воздействий на скорость адаптации и, следовательно, на скорость распространения инновации.

Внешние влияния на скорость адаптации определяются потребностью пользователей в инновациях и уровнем маркетинговых и

рекламных коммуникаций, чему соответствует слагаемое a(M-N) в правой части.

В н у т р е н н и е в л и я н и я на скорость адаптации обусловлены коммуникациями между действующими пользователями инновации и потенциальными потребителями (в результате которых потенциальным потребителям передается информация об инновационном продукте), и этому соответствует слагаемое bN(t)(M-N).

В этом лабораторной работе нужно исследовать влияние внешних воздействий в виде рекламы на продвижение инноваций, (программного продукта) при b = 0, тогда:

$$\frac{dN}{dt} = a(M - N)$$

Разделяя переменные в уравнении, имеем

$$\int \frac{dN}{M - N} = \int a\, dt$$

решение

$$\ln(M - N) = -at + \text{Const}$$

или $N(t) = M - \mathrm{e}^{-at+\text{Const}}$

Постоянная интегрирования определяется из начального условия, и окончательно рост охвата рынка во времени описывается функцией

$$N(t) = M - (M - N_0)\,\mathrm{e}^{-at}$$

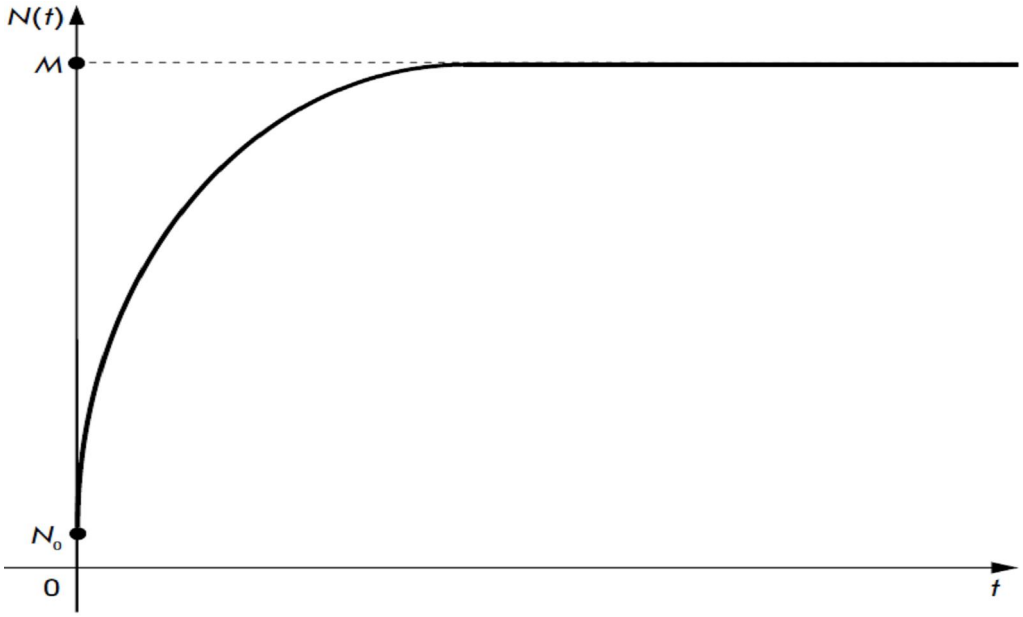


Рис. 2. Рост объема охваченного рынка в моделях внешнего влияния

Агенты в AnyLogic Агент – это некоторая сущность, обладающая активностью, автономным поведением, принимающая решения в соответствии с некоторым набором правил, взаимодействующая с окружением и другими агентами, а также сама при этом изменяющаяся. При помощи агентов целесообразно моделировать рынки ИКТ (агенты – производитель программного продукта и потенциальный пользователь). Активный объект имеет параметры, которые можно изменять извне, переменные, которые можно считать памятью агента, а также поведение (рис. 3).

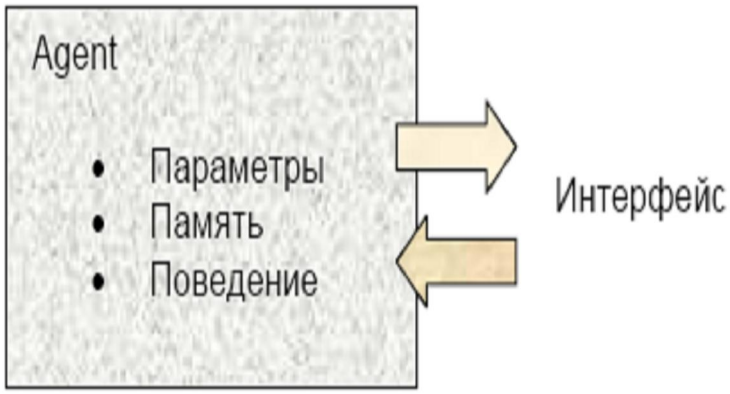


Рис. 3. – Активный объект

Ход работы. Создайте проект для своей модели и сохраните его в папке. Первым шагом при создании агентной модели является создание агентов. Для каждого агента задается набор правил, согласно которым он взаимодействует с другими агентами; это взаимодействие и определяет общее поведение системы. В данной работе агентами будут пользовали программного продукта. Создадим агентную модель с помощью Мастера создания модели.

Шаг 1. Щелкните мышью по кнопке панели инструментов Создать. Появится - Agent • Параметры • Память • Поведение Интерфейс. Появится диалоговое окно Новая модель. Задайте имя новой модели. Щелкните мышью по кнопке Далее.

Шаг 2. Выберите шаблон модели (рис. 2). В связи с тем, что создаем новую агентную модель, то нужно установить флажок Использовать шаблон модели и выбрать Агентная модель в расположенном ниже списке Выберите метод моделирования. Щелкните мышью по кнопке Далее.

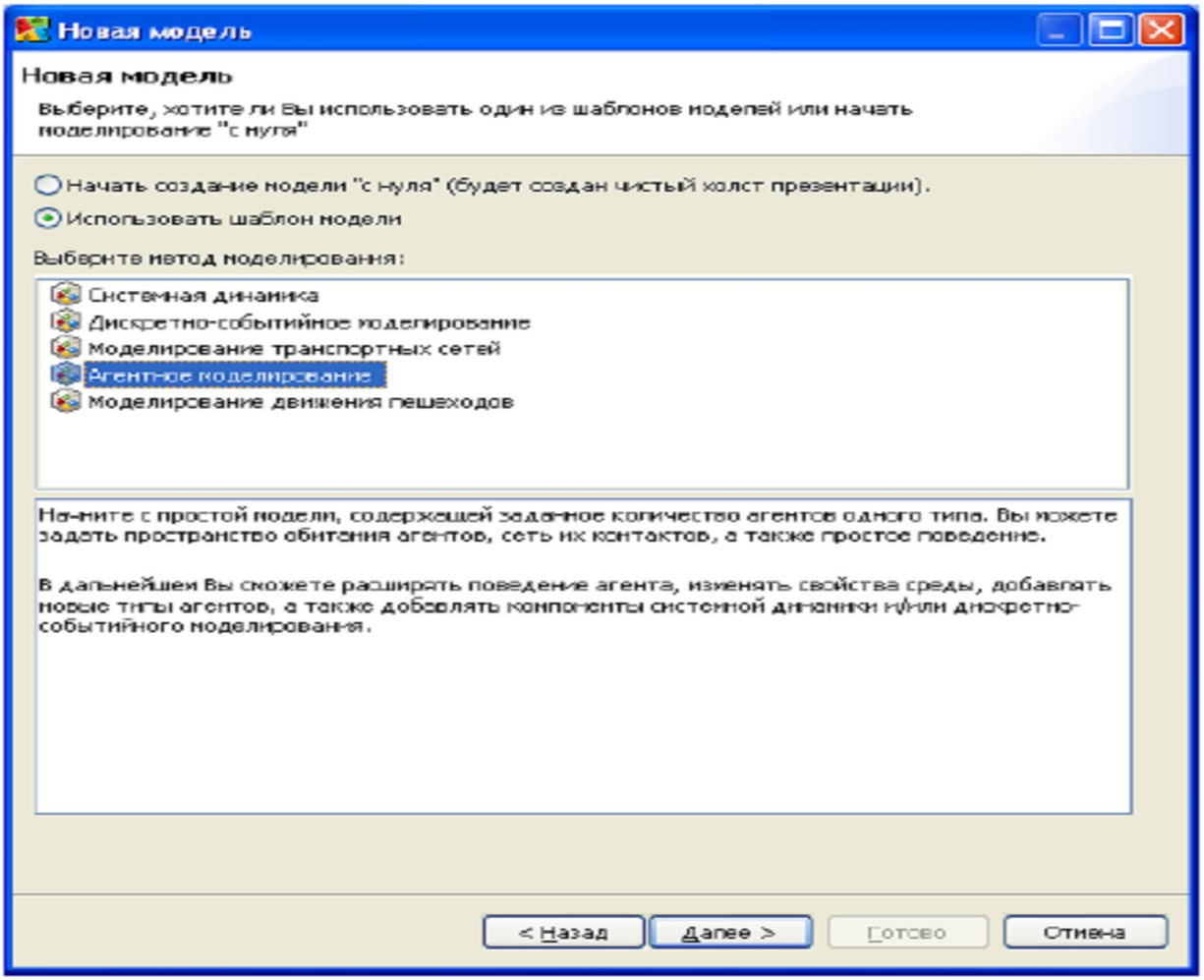


Рис. 4. Экран агентной модели

Шаг 3. Создайте агентов, т.е. задайте имя класса агента и количество агентов, которое будет изначально создано в нашей модели. Задайте в качестве имени класса Person и введите в поле Начальное количество пользователей программного продукта (агентов) 1000. Щелкните мышью по кнопке Далее.

Шаг 4. Задайте свойства пространства, в котором будут обитать агенты и выберите фигуру анимации агента. Установите флажок Добавить пространство и выберите ниже тип этого пространства – Непрерывное. Задайте размер данного пространства: введите в поле Ширина – 600, а в поле Высота – 350. В результате агенты будут располагаться каким-то образом в пределах непрерывного пространства, отображаемого на презентации модели 65 областью размером 600×350 пикселей. Не меняйте значения, выбранные в выпадающих списках Начальное расположение и Анимация;

Пусть агенты изначально расставляются по пространству случайным образом, а анимируются с помощью фигурки человечка. Щелкните мышью по кнопке Далее.

Шаг 5. Задайте сеть взаимосвязей агентов (рис. 3). Установите флажок Использовать сеть и оставьте выбранной опцию Случайное. Ниже можно установите флажок Показывать связи, чтобы отображать на презентации связи между знакомыми (или потенциально могущими встретиться и пообщаться) агентами с помощью линий. Щелкните мышью по кнопке Далее.

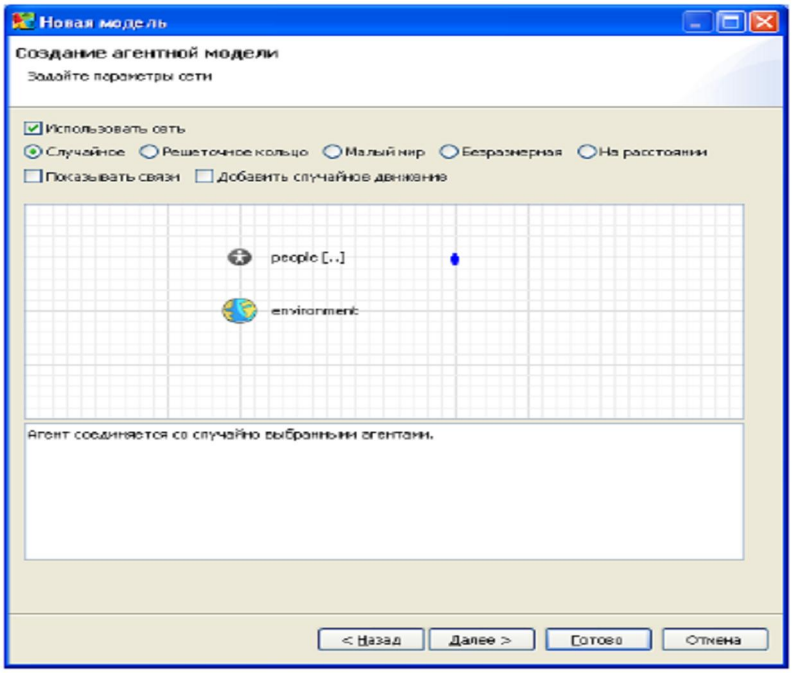


Рис. 5. Экран установки связей

Шаг 6. Установите флажок Добавить простое поведение. В результате у агента будет создана диаграмма состояний.

Задание характеристик агента. Характеристики агента задаются с помощью параметров класса. Все агенты обладают общей структурой, поскольку все они задаются объектами одного класса. Параметры же позволяют задавать характеристики индивидуально для каждого агента.

Создадим параметр, задающий подверженность пользователя влиянию рекламы. Откройте структурную диаграмму класса Person. Перетащите элемент Параметр из палитры Основная на диаграмму класса, в окне свойств параметра задайте имя AdEffectiveness, значение по умолчанию – 0.011.

Задание поведения агента Поведение агента обычно описывается в классе этого агента (в этой модели - класс Person) с помощью диаграммы состояний (стейтчарт). Мастер создания моделей уже создал простейшую диаграмму состояний из двух состояний, между которыми существует два разнонаправленных перехода. Изменим данный стейтчарт.

1. Откройте структурную диаграмму класса Person. На диаграмме класса видно следующую диаграмму состояний (рис. 5).

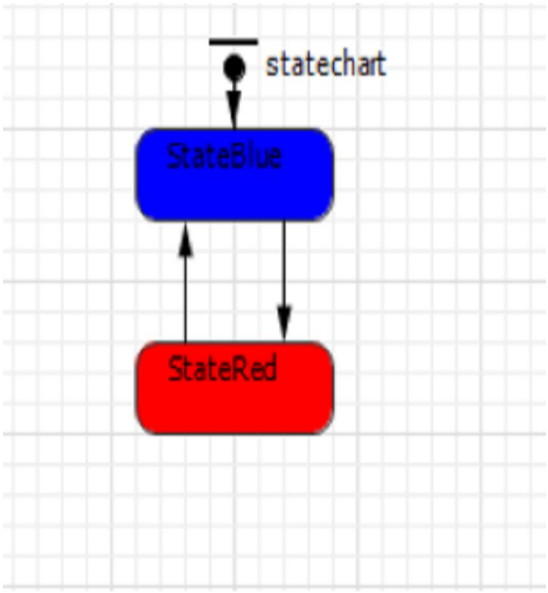


Рис. 6. Стейтчарт. Поведение агента

2. Откройте свойства верхнего состояния, переименуйте верхнее состояние в PotentialAdopter. Это начальное состояние. Нахождение стейтчарта в данном состоянии означает, что пользователь еще не купил программный продукт.

3. Нижнее состояние назовите Adopter (т.е. пользователь уже купил программный продукт).

4. Измените свойства перехода из состояния PotentialAdopter в состояние Adopter. Этот переход будет моделировать покупку программного продукта. В окне свойств перехода выберите С заданной интенсивностью из выпадающего списка Происходит и введите AdEffectiveness в расположенном ниже поле Интенсивность. Время, через которое

пользователь купит программный продукт, экспоненциально зависит от эффективности рекламы п программного продукта.

5. Удалите переход, ведущий из нижнего состояния в верхнее, поскольку пока создается простейшая модель, в которой пользователь, однажды приобредший продукт, навсегда остается его потребителем, и соответственно перехода из состояния Adopter в состояние PotentialAdopter пока что быть не должно (рис. 7). Чтобы удалить переход, выделите его на диаграмме и нажмите Del.

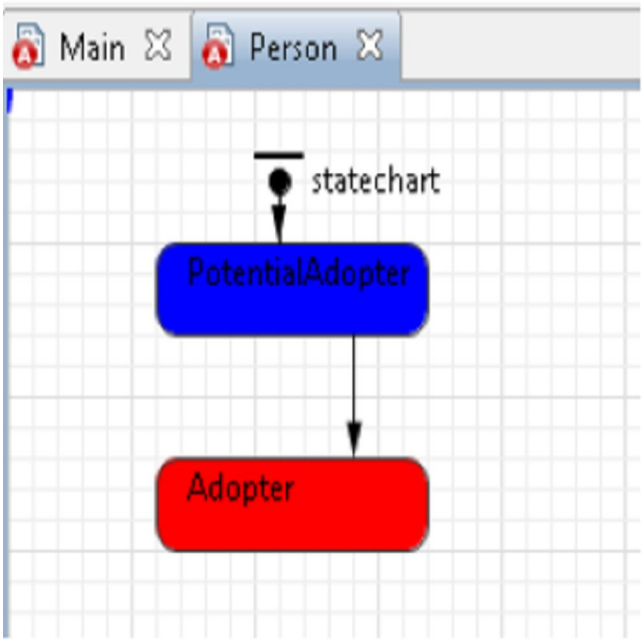


Рис. 7. Стейтчат. Удаление перехода

6. Настройте выполнение модели (рис. 8). В окне свойств эксперимента перейдите на вкладку Модельное время и задайте останов модели после 8 единиц модельного времени.

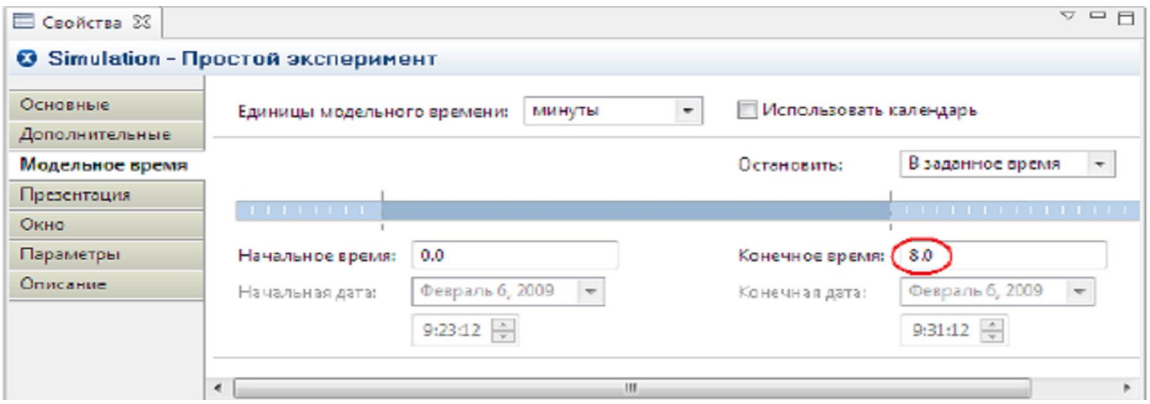


Рис. 8. Работа модели

7. Постройте проект с помощью кнопки панели инструментов Построить (клавиша F7). Если ошибок в проекте нет, то запустите модель. Вы увидите, как число потенциальных покупателей (синих) переходит в разряд покупателей (красных).

Подсчет потребителей программного продукта Главная задача модели распространения инноваций в виде программного продукта – изучение того, как быстро пользователи покупают новый программный продукт. Для этого будем подсчитывать число пользователей и потенциальных пользователей программного продукта, что можно сделать с помощью функций сбора статистики.

Создадим функции сбора статистики для подсчета потенциальных пользователей программного продукта.

1. Откройте диаграмму класса Main. Выделите на диаграмме вложенный объект people.

2. Перейдите на вкладку Статистика панели свойств объекта people. Щелкните мышью по кнопке Добавить функцию сбора статистики. Откроется секция свойств для задания свойств новой функции сбора статистики по элементам этого реплицированного объекта (people).

3. Задайте имя функции – potentialAdopters. Оставьте выбранный по умолчанию Тип функции – количество. Задайте Условие: item.statechart.isStateActive(item.PotentialAdopter) Эта функция будет вести подсчет количества агентов, для которых выполняется заданное условие, т.е. тех агентов, которые находятся в текущий момент времени в состоянии PotentialAdopter (являются потенциальными потребителями программного продукта). Здесь item – это агент (элемент реплицированного объекта people).

4. Создайте еще одну функцию сбора статистики (рис. 9). Назовите ее adopters. Тип функции – количество. Условие: item.statechart.isStateActive(item.Adopter) Данная функция будет вести подсчет количества агентов, которые находятся в состоянии Adopter (т.е. уже приобрели программный продукт).

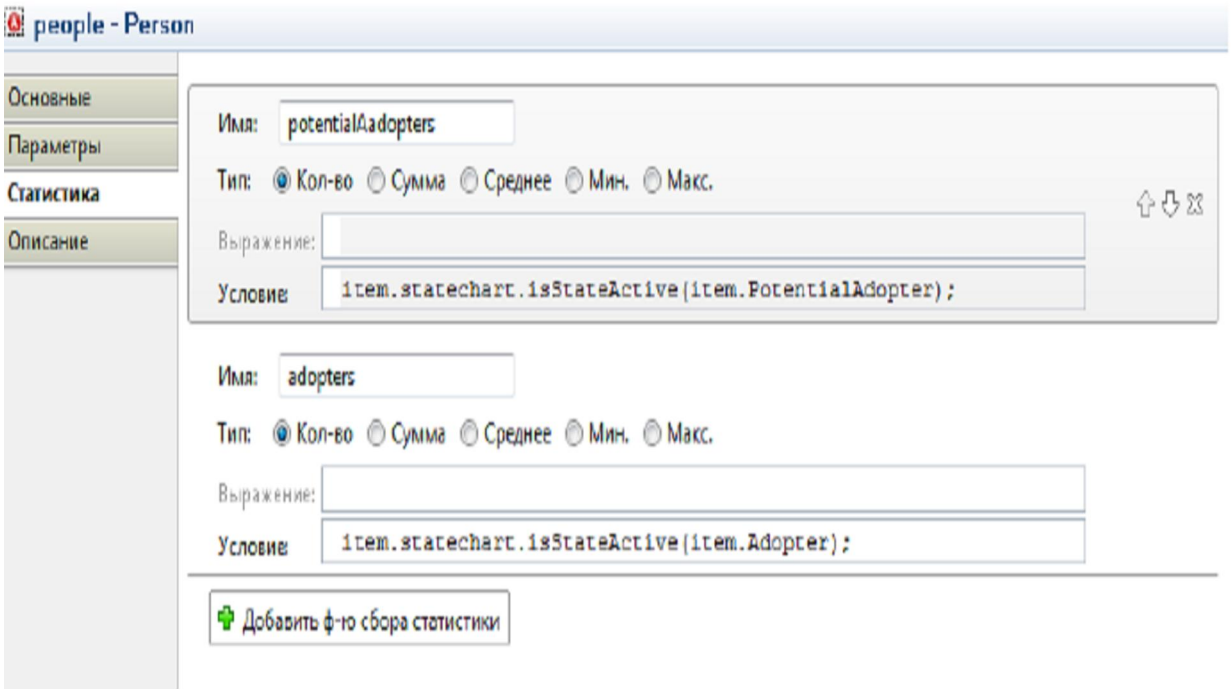


Рис. 9. Создание функции adopters

Добавьте временной график, отображающий динамику изменения численностей потребителей и потенциальных потребителей продукта. Расположите его, как показано на рис. 10.

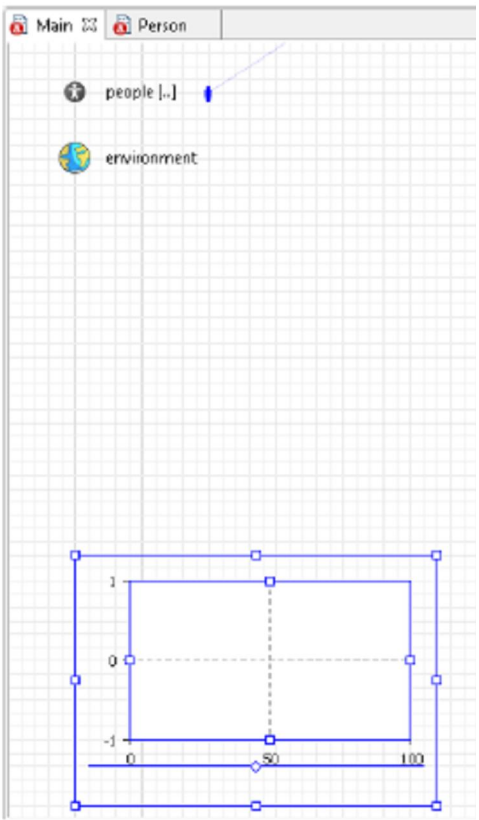


Рис. 10. Функции потребителей

Настройте свойства графика (рис. 11).

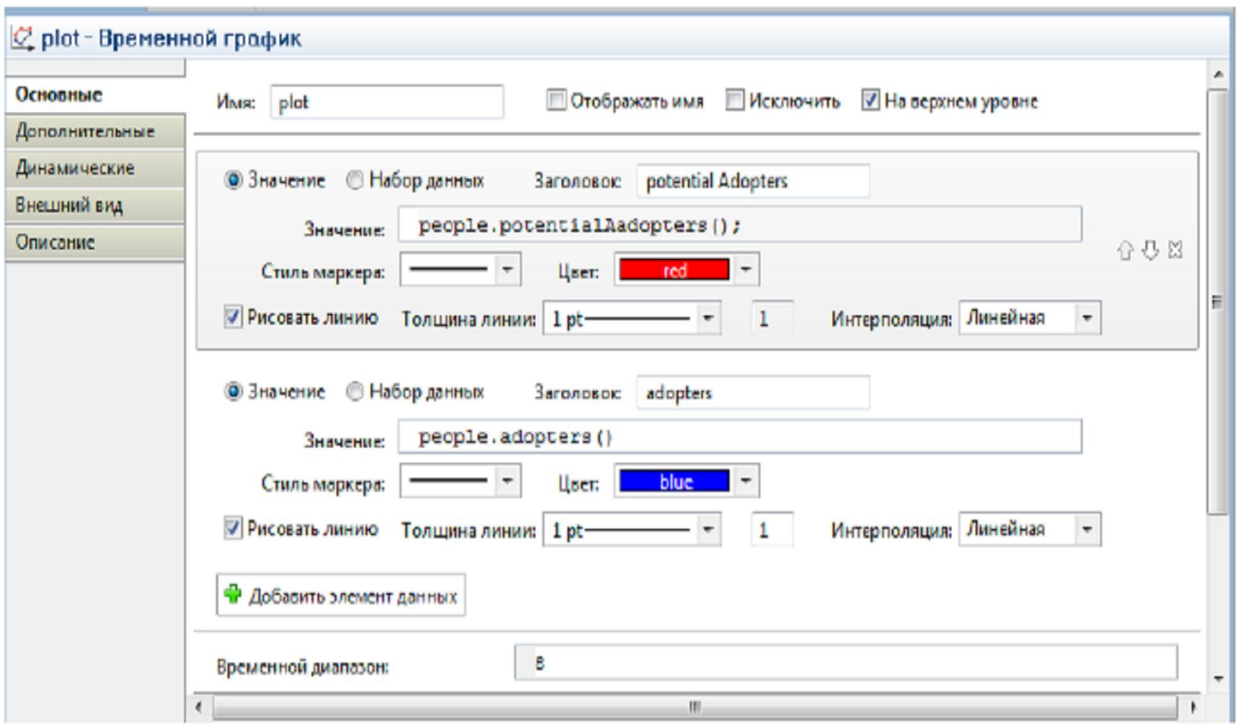


Рис. 11. Настройка свойства графиков

Запустите модель. На графике (рис. 12) проследите динамику моделируемого процесса. Под влиянием рекламы каждую единицу времени постоянная доля от общей численности потенциальных потребителей программного продукта приобретает распространяемый программный продукт.

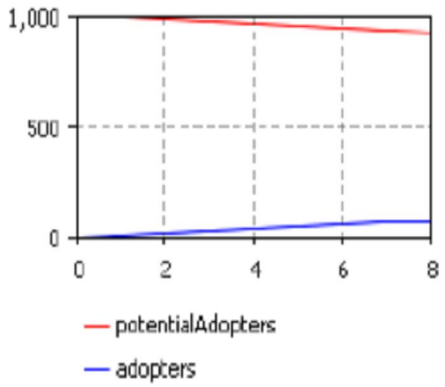


Рис. 12. Графики динамики моделирования процесса приобретения программного продукта

В результате работы студент должен предоставить отчет по лабораторной работе с выводами.

Контрольные вопросы

1. В чем суть диффузии инноваций?

2. Что такое внутренние и внешние воздействия на диффузию инноваций?

3. К какому дифференциальному уравнению сводится диффузия инноваций?

4. Что такое мультиагентные системы?

5. Что такое стейтчарт?

Содержание отчета

1. Цель работы.

2. Кратко основные теоретические сведения.

3. Диффузия инноваций в виде ОДУ Коши.

4. Скриншоты работы модели инновацийю

5. Графики диффузии инноваций в виде программного продукта.

6. Ответы на контрольные вопросы.

Рекомендуемые источники

1. Замятин Н.В.Рынки ИКТ и организация продаж : учеб. пособие. –Томск, 2015. – 202 с.

2. Соловьев В. И. Стратегия и тактика конкуренции на рынке программного обеспечения : Опыт экономико_математического моделирования: монография / В. И. Соловьев. — М.: Вега_Инфо, 2010. — 200 с.

3. Замятин Н.В. Методические указания к выполнению лабораторных и практических работ по дисциплине «Рынки ИКТ и организация продаж».- Томск: ТУСУР, 2015 - 100 с. [Электронный ресурс]: сайт кафедры АОИ.

4. AnyLogic User’s Manual. XJ Technologies : [электрон. ресурс]. Режим доступа : http://www.xjtek.com

5. AnyLogic Tutorial. XJ Technologies : [электрон. ресурс]. Режим доступа: http://www.xjtek.com.

Лабораторная работа 5. Нейросетевые модели аппроксимации функций

Цель работы: изучить процедуры идентификации моделей, уяснить необходимость аппроксимации функций входов и выходов при идентификации математических моделей (для этой цели целесообразно применить новый подход в виде нейронных сетей); научиться работать с нейронной сетью прямой передачи данных и с алгоритмом обучения методом обратного распространения ошибки для аппроксимации нелинейной функции.

Краткие теоретические сведения

При построении математических моделей большого числа объектов достаточно эффективным оказывается использование методов идентификации - процедур построения адекватной в регламентированном смысле модели какого-либо объекта по экспериментальным данным (реализациям входных и выходных процессов). Идентифицированные модели используются в дальнейшем для анализа и оптимизации процессов функционирования, диагностики, управления технических систем, создания математического обеспечения соответствующих САПР, АСНИ. В связи с этим современному инженеру необходимы знания и навыки построения достаточно простых и адекватных математических моделей технических систем, основываясь на экспериментальных данных. Под идентификацией объекта понимают построение его математической модели по данным о реакции объекта на известные внешние воздействия.

Пусть в результате каких-либо экспериментов над некоторым объектом измерены его входные переменные $X=(x_1,x_2,\ldots,x_n)$ и выходные переменные $Y=(y_1,y_2,\ldots,y_n)$. Требуется определить модель (структуру) ставящей в соответствие переменные x и y: $Y=\hat{A}(X)$. Различают задачи идентификации в широком (структурная идентификация) и узком (параметрическая идентификация) смысле. В первом случае считаются

неизвестными структура и параметры оператора $\hat{A}$, во втором – лишь параметры этого оператора.

В качестве «объекта идентификации» в общем случае могут рассматриваться механическая, электрическая или биологическая система, замкнутые или разомкнутые (с обратной связью) системы управления и т. д. Методы идентификации таких разнообразных по назначению, виду и структуре объектов естественно различаются, хотя существует и ряд общих закономерностей. В практике идентификации обычно используют совокупность различных методов, применяя их в порядке возрастания адекватности и точности получаемых моделей.

Требуется для каждой выходной переменной y_j построить аналитическую зависимость $y_j = \hat{A}_j(X)$, адекватно описывающую имеющиеся экспериментальные данные (в дальнейшем индекс при у будем опускать, предполагая, что выходные переменные друг от друга не зависят, и без ограничения общности достаточно рассмотреть случай одномерного вектора Y: $y = \hat{A}(X)$).

Если заданные величины *x* и *y* являются экспериментальными данными, то могут содержать в себе существенные ошибки, так как получены в результате измерений или наблюдений. Поэтому построение аппроксимирующего многочлена, воспроизводящего в точности заданное значение функции, означало бы тщательное копирование допущенных при измерениях ошибок.

2. Если имеются точные значения функции в некоторых точках, но число таких точек *n* весьма велико, то интерполяционный многочлен будет очень высокой степени (при условии, что разности не будут постоянными).

Поэтому возникает задача аппроксимации – построения многочлена некоторой вполне определенной степени, но меньшей чем $n-1$, который хотя и не дает точных значений функции в узлах интерполяции, но достаточно близко к ним подходит.

В лабораторной работе необходимо получить значения экспериментальных данных из таблицы и аппроксимировать эти зависимости нейронной сетью в программной среде МатЛаб.

Для реализации процедуры аппроксимации использовать нейронную сеть с прямой передачей сигнала (с прямой связью), то есть сеть, в которой сигналы передаются только в направлении от входного слоя к выходному, и элементы одного слоя связаны со всеми элементами следующего слоя. Важнейшим для реализации нейронных сетей является определение алгоритма обучения сети.

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*. Присвоим выходам нейронной сети номера с индексом $j = 1,2,\ldots,n$, а обучающие примеры индексом $M = 1,2,\ldots,M_0$. 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 $\mathbf{x} = \{x_i\}$ – столбец входных значений, где i=1,2,..,n. Тогда $\mathbf{y} = \{y_j\}$ – выходные значения, где j=1,2,…,m. В общем случае $n \neq m$. 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

$$\|\mathbf{y} - \mathbf{d}\| = \sqrt{(\mathbf{y} - \mathbf{d}, \mathbf{y} - \mathbf{d})^2} \quad . \qquad (1)$$

Пусть целевая функция имеет вид

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

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 $E(\mathbf{w})(E \geq 0).$

На рис. 1 показана архитектура нейронной сети с прямой передачей сигнала

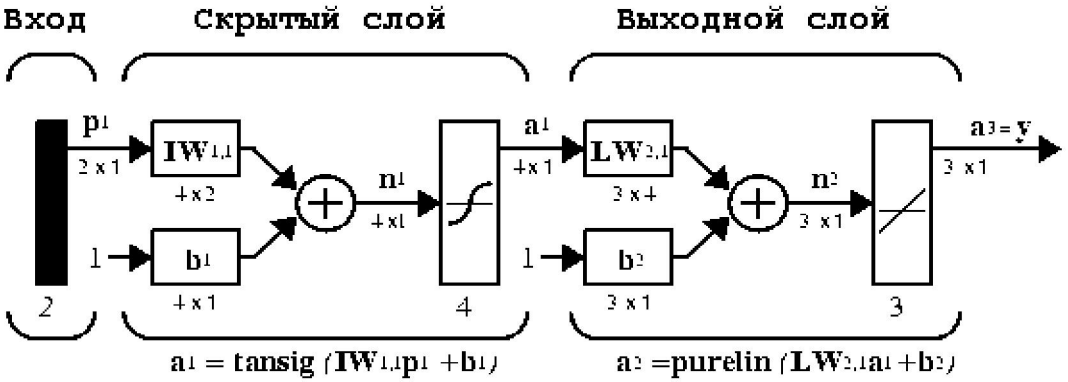


Рисунок 1 –Схема архитектуры нейронной сети с прямой передачей сигнала

Обозначения: p^1 - вектор входа, $IW^{i,j}$, $LW^{i,j}$ - матрицы весов входа и выхода, b^i - смещение, a^i - выход слоя, y - выход сети, *tansig* (гиперболическая тангенциальная), *purelin*(линейная) - соответствующие функции активации.

Веса и смещения определяются с помощью алгоритма обратного распространения ошибок.

Обучение сети обратного распространения требует выполнения следующих операций:

Выбрать очередную обучающую пару из обучающего множества; подать входной вектор на вход сети.

Вычислить выход сети.

Вычислить разность между выходом сети и требуемым выходом (целевым вектором обучающей пары).

Скорректировать веса сети так, чтобы минимизировать ошибку.

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

Пример решения задачи

Выполнение лабораторной работы состоит из следующих этапов: прежде всего, необходимо оцифровать график функции **y=f(x)**, то есть получить ряд соответствующих значений по горизонтальной и вертикальной осям.

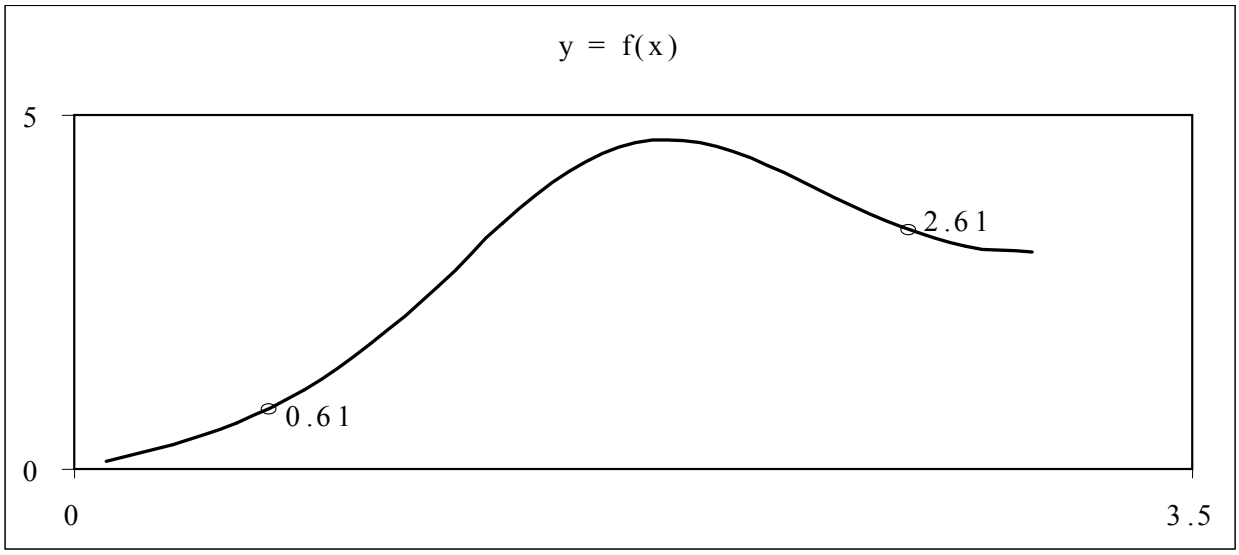


Рис. 2. – Пример зависимости для функции одной переменной

В примере, показанном на рис. 2 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]**.

Ниже приводится программа создания, обучения нейронной сети и вывода результатов.

```
x=[0.10  0.31  0.51  0.72  0.93  1.14 ...
1.55  1.76  1.96  2.17  2.38 ...
2.79  3.00];
y=[0.1010 0.3365 0.6551 1.1159 1.7632 2.5847 ...
4.2115 4.6152 4.6095 4.2887 3.8349 ...
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 рис. 3 и 4.

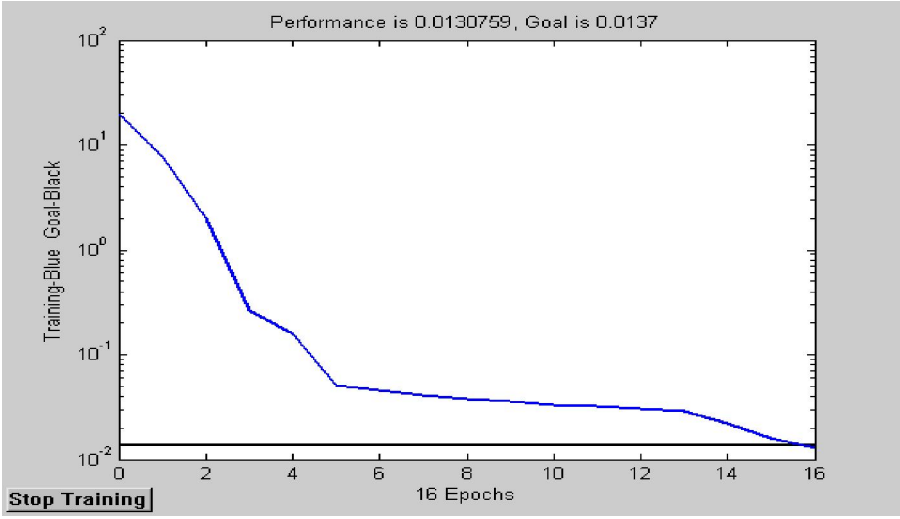


Рис. 3. Характеристика точности обучения в зависимости от числа эпох обучения

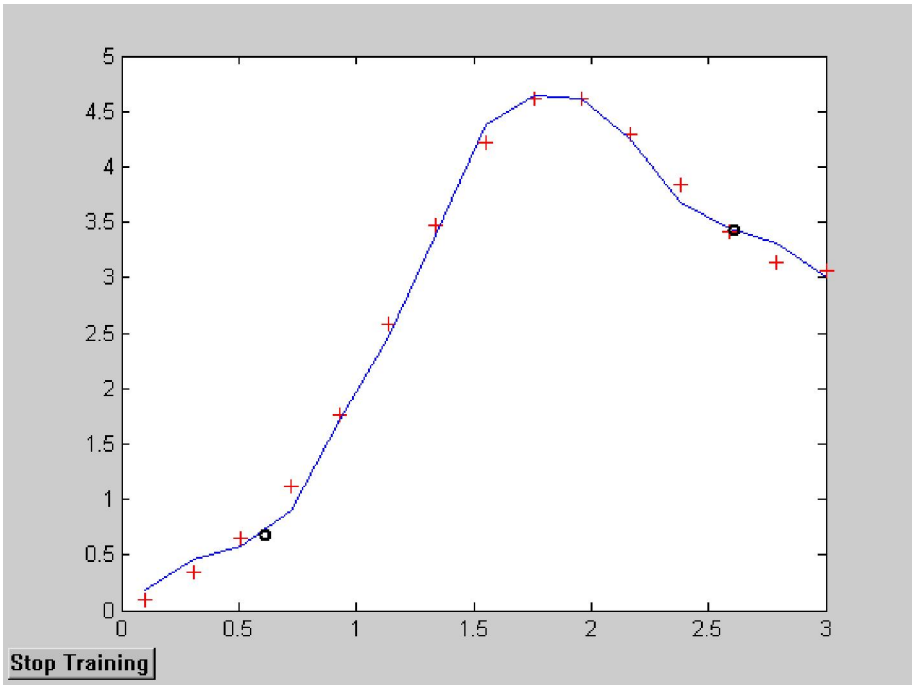


Рис. 4. – Результаты моделирования сети: + - исходные данные; сплошная линия и символ «о» – результаты моделирования всей зависимости и в контрольных точках

В массиве **v** содержатся приближённые значения для двух контрольных точек, указанных на графике (рис. 2) **xx=[0.61 2.61]**. При данных параметрах сети получены значения: **v = [1.05 3.35]**. Сравнив эти приближённые значения с точными значениями **[0.85 3.37]**, можно сделать вывод о корректности построения нейронной сети.

Содержание отчёта

Исходные данные своего варианта;

Текст программы с подробными комментариями;

Характеристику точности обучения и сравнения с результатами анализа и моделирования;

Сопоставление результатов в контрольных точках;

Краткие выводы о результатах работы.

Варианты заданий

Создайте таблицу 1 экспериментальных данных:

$xi = a + h\ i$, $i = 0,1, ,10$, $h = (b - a)\ /10$ K на отрезке $[a,b]$.

Вариант

$iy[a,b]$

1 2.86; 2.21; 2.96; 3.27; 3.58; 3.76; 3.93; 3.67; 3.90; 3.64; 4.09[0, 1]

1.14; 1.02; 1.64; 1.64; 1.96; 2.17; 2.64; 3.25; 3.47; 3.89; 3.36 [-1, 1]

4.70; 4.64; 4.57; 4.45; 4.40; 4.34; 4.27; 4.37; 4.42; 4.50; 4.62 [2, 4]

0.43; 0.99; 2.07; 2.54; 1.67; 1.29; 1.24; 0.66; 0.43; 0.35; 0.70 [2, 4]

1.55; 1.97; 1.29; 0.94; 0.88; 0.09; 0.02; 0.84; 0.81; 0.09; 0.15 [1, 4]

3.24; 1.72; 1.95; 2.77; 2.47; 0.97; 1.75; 1.55; 0.12; 0.70; 1.19[0, 4]

2.56; 1.92; 2.85; 2.94; 2.39; 2.16; 2.51; 2.10; 1.77; 2.28; 1.70 [-1, 2]

1.77; 0.92; 2.21; 1.50; 3.21; 3.46; 3.70; 4.02; 4.36; 4.82; 4.03 [-1, 3]

1.53; 0.45; 1.68; 0.12; 0.68; 2.36; 2.58; 2.53; 3.45; 2.70; 2.82 [4, 8]

2.50; 3.90; 3.54; 4.63; 3.87; 5.25; 4.83; 3.24; 3.08; 3.00; 4.70 [0, 5]

2.95; 3.38; 2.71; 2.37; 2.29; 2.75; 2.76; 2.74; 2.57; 2.40; 2.99 [1, 5]

-0.23; -0.03; -0.98; -0.97; -0.43; -0.91; -0.27; -0.19; 0.88; 1.06; 0.72 [2, 4]

2.36; 0.03; -0.38; -1.33; 0.25; -1.36; 0.95; 3.16; 4.03; 4.92; 4.20 [0, 2]

3.82; 4.07; 3.53; 4.83; 5.53; 5.04; 5.09; 5.87; 5.53; 4.72; 4.73 [3, 4]

2.35; 2.16; 2.39; 2.39; 2.18; 2.09; 2.44; 2.56; 3.35; 3.22; 2.65 [-3, 4]

1.77; 0.99; 2.21; 1.50; 2,77; 3.46; 3.70; 4.02; 4.77; 4.82; 4.03 [-1, 2]

Лабораторная работа № 6. Нечеткие модели для аппроксимации функций

Цель работы: изучение основных функций пакета Fuzzy Logic Toolbox программной среды MatLab для построения нечетких моделей аппроксимации функций..

Теоретическая часть

Пакет FuzzyLogicToolbox (пакет нечеткой логики) – это совокупность прикладных программ, относящихся к теории *нечетких* множеств и позволяющих конструировать модели нечетких систем.

Основные возможности пакета:

построение систем нечеткого вывода (регуляторов, аппроксиматоров зависимостей);

построение адаптивных нечетких моделей систем;

интерактивное динамическое моделирование в Simulink.

Пакет позволяет работать:

в режиме графического интерфейса;

в режиме командной строки;

с использованием блоков и примеров пакета Simulink.

Графический интерфейс FuzzyLogicToolbox

В состав программных средств FuzzyLogicToolbox входят следующие основные программы, позволяющие работать в режиме графического интерфейса:

редактор нечеткой системы вывода FuzzyInferenceSystemEditor (FISEditor или FIS-редактор) вместе со вспомогательными программами – редактором функций принадлежности (MembershipFunctionEditor), редактором правил (RuleEditor), просмотрщиком правил (RuleViewer) и просмотрщиком поверхности отклика (SurfaceViewer);

редактор гибридных систем (ANFISEditor, ANFIS-редактор);

программа нахождения центров кластеров (программа Clustering).

Построение нечеткой модели аппроксимирующей функции.

Командой (функцией) **Fuzzy** из режима командной строки запускается основная интерфейсная программа пакета FuzzyLogic– редактор нечеткой системы вывода (FuzzyInferenceSystemEditor, FISEditor, FIS-редактор).

Вид открывающегося при этом окна приведен на рис. 1.

File – работа с файлами моделей (их создание, сохранение, считывание и печать);

Edit – операции редактирования (добавление и исключение входных и выходных переменных);

View – переход к дополнительному инструментарию.

Задание на лабораторную работу. Разработать нечеткую модель, отображающую зависимость между переменными x и y, заданную с помощью табл.1, представленные в таблице данные отражают зависимость $y = x^2$.

Таблица 1. Значения x и y

x	-1	- 0.6	0	0.4	1
y	1	0.36	0	0.16	1

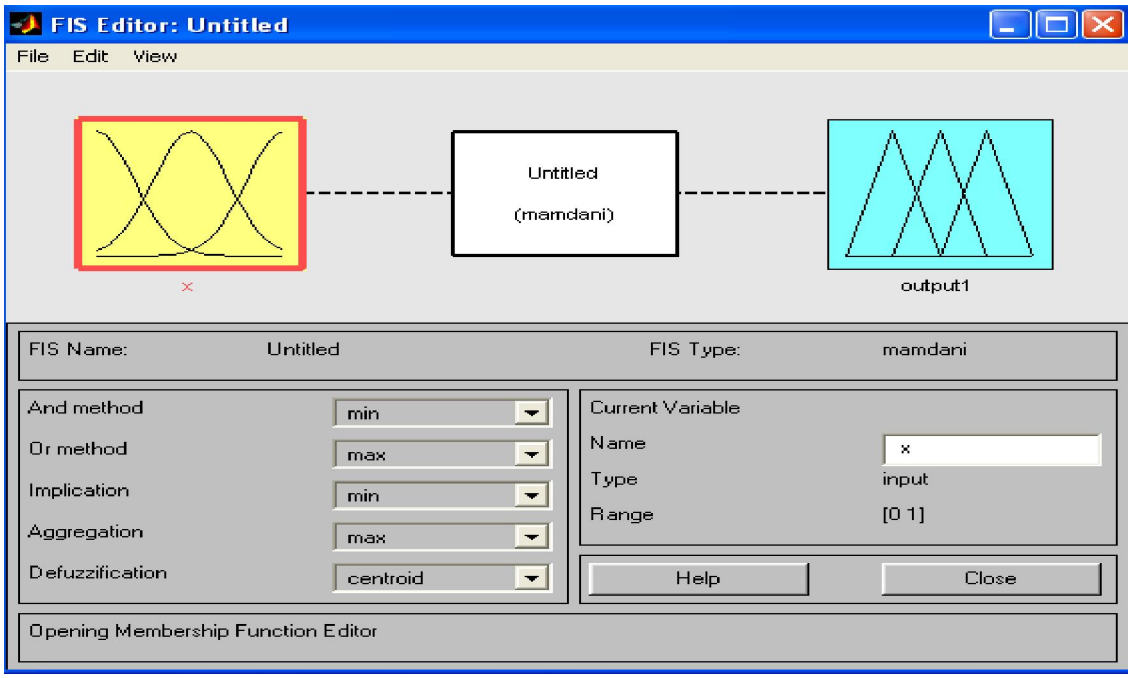


Рис. 1. Вид окна FisEditor

Требуемые действия отобразим следующими пунктами.

В позиции меню File выбираем опцию NewSugenoFIS (новая система типа Sugeno), при этом в блоке, отображаемом белым квадратом, в верхней части окна редактора появится надпись Untitled2 (sugeno).

Щелкнем левой кнопкой мыши по блоку, озаглавленному inputl (вход 1). Затем в правой части редактора в поле, озаглавленном Name (Имя), вместо inputl введем обозначение нашего аргумента, т.е. *x*. Обратим внимание, что если теперь сделать где-нибудь (вне блоков редактора) однократный щелчок

мыши, то имя отмеченного блока изменится на x; то же достигается нажатием после ввода клавиши Enter.

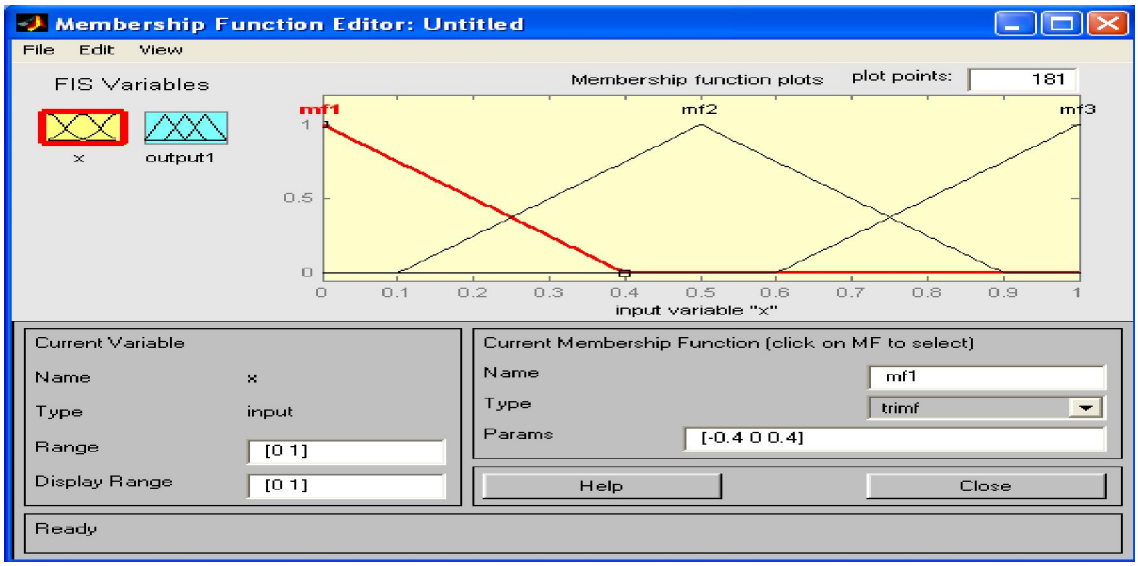


Рис. 2 Окно редактора функций принадлежности

Задание на лабораторную работу

Разработать нечеткую систему, отображающую зависимость между переменными x и y, заданную с помощью табл. 2. По результатам работы определите тип кривой.

Таблица 2

Вариант	Значения						
1	x	-1	-0.5	0	0.2	1	
	y	1	0.25	0	0.4	1	
2	x	-1	-0.6	0.2	0.4	1	
	y	-1	-1.67	5	2.5	1	
3	x	-1	-0.5	0	0.3	1	
	y	-1	-0.13	0	0.27	1	
4	x	-1	-0.6	0	0.3	1	
	y	0	0.8	1	0.95	0	

5	x	-2	-0.35	0	0.56	2	
	y	1	0.56	0.1	0.3	1	
6	x	-1	-0.5	0.2	0.3	1	
	y	-1	-1.34	5	2.12	1	
7	x	-2	-0.3	0	0.4	2	
	y	-1	-0.12	-0.23	0.24	1	
8	x	-1	-0.6	0	0.3	1	
	y	0	0.8	1	0.95	0	
9	x	-1	-0.6	0	0.3	1	
	y	1	0.36	0	0.9	1	

Контрольные вопросы

1. Дайте определение нечеткого множества

2. В каких пределах определяется функция принадлежности?

3. Что такое лингвистическая переменная?

4. Что такое нечеткое правило?

5. В чем суть логического вывода?

Содержание отчета

1. Цель работы.
2. Задание.
3. Краткое описание действий.
4. Графики по всем пунктам программы.
5. Выводы по работе.

Список литературы

1. Дьяконов В.П. MATLAB 6/6.1/6.5 + SIMULINK 4/5 в математике и моделировании.– М.: СОЛОН-Пресс, 2003 – 565 с.

2. Круглов В. В., Дли М.И., Голунов Р. Ю. Нечёткая логика и искусственные нейронные сети: Учеб. Пособие. – М.: Издательство Физико-математической литературы.