

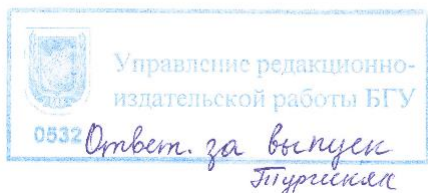
БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ  
ФАКУЛЬТЕТ РАДИОФИЗИКИ И КОМПЬЮТЕРНЫХ ТЕХНОЛОГИЙ  
Кафедра системного анализа и компьютерного моделирования

# МАТЕМАТИЧЕСКОЕ МОДЕЛИРОВАНИЕ

Методические указания  
к лабораторным работам

Для студентов специальностей  
1-31 04 02 «Радиофизика»,  
1-31 04 03 «Физическая электроника»,  
1-31 04 04 «Аэрокосмические радиоэлектронные  
и информационные системы и технологии»

МИНСК  
2017



Главное управление учебной и научно-методической работы БГУ	
<b>В ПЕЧАТЬ</b>	
Тираж	20 экз
Нач. (зам.) ГУУНМР	<i>[Signature]</i>
« 02 »	02 20 17 г.

УДК 519.216(076.5)  
ББК 22.172я73-5  
М34

А в т о р ы :

**А. В. Дигрис, С. В. Гилевский, Ю. Л. Крученок,  
В. М. Лутковский, В. В. Скакун, В. В. Апанасович**

Рекомендовано советом  
факультета радиофизики и компьютерных технологий  
20 декабря 2016 г., протокол № 4

Рецензент

кандидат физико-математических наук,  
доцент *Н. Н. Яцков*

**Математическое** моделирование : метод. указания к ла-  
М34 бораторным работам / А. В. Дигрис [и др.]. – Минск : БГУ,  
2017. – 59 с.

Методические указания предназначены для проведения лабора-  
торных работ по курсу «Математическое моделирование». Содержат  
учебный материал по методам моделирования случайных величин и  
потоков случайных событий, а также статистической проверке полу-  
ченных результатов.

Предназначено для студентов факультета радиофизики и компь-  
ютерных технологий БГУ.

**УДК 519.216(076.5)  
ББК 22.172я73-5**

© БГУ, 2017

## ВВЕДЕНИЕ

Настоящее издание является руководством к лабораторному практикуму по курсу «Математическое моделирование». Цель практикума – обучение студентов разработке, программной реализации и тестированию алгоритмов моделирования непрерывных и дискретных случайных величин, а также простейшего потока случайных событий. Студентам предоставляется возможность овладеть навыками программирования в интегрированной вычислительной среде MATLAB и изучить основные ее возможности. Выполняя задания к лабораторным работам, студенты знакомятся со встроенными функциями среды MATLAB для моделирования наиболее широко используемых на практике непрерывных равномерного и нормального распределений, а также дискретного распределения Пуассона. Среди методов, предназначенных для разработки собственных алгоритмов моделирования, студентам предлагается изучить методы обратных функций и Неймана для построения генераторов непрерывных случайных величин, а также единый подход для создания генераторов дискретных случайных величин независимо от типа их распределений.

Приведенные теоретические сведения содержат достаточный объем информации, исключая необходимость прибегать для выполнения заданий к другим литературным источникам. Наличие значительного набора вариантов заданий к каждой лабораторной работе позволяет индивидуально оценить уровень знаний каждого отдельного студента.

## ЛАБОРАТОРНАЯ РАБОТА № 1

### ИЗУЧЕНИЕ СРЕДЫ MATLAB И SIMULINK

**Цель работы.** Изучение основных функций системы MATLAB и элементов программирования на М-языке. Освоение матричных вычислений. Создание новых функций и построение их графиков в среде MATLAB. Знакомство с пакетом визуального динамического моделирования SIMULINK, и создание на его основе моделей радиофизических систем.

**Краткие сведения.** Система MATLAB (MATrix LABoratory) предназначена для выполнения научных и инженерных расчетов на компьютере [1]. С ее помощью эффективно решаются задачи вычислительной математики, линейной алгебры, математической статистики и математического моделирования (символьная математика, уравнения в частных производных, численное интегрирование, аппроксимация, решение оптимизационных задач). В ее состав входит несколько десятков специализированных пакетов (Toolbox), предназначенных для обработки данных в различных областях науки и техники (системы управления, нечеткие системы, нейронные сети, цифровая обработка сигналов и изображений).

MATLAB – это интегрированная среда разработки приложений, включающая редактор для создания исходного кода программ, средства для их отладки и просмотра результатов их работы.

В рамках данного пособия материал изложен в предположении, что используется версия MATLAB 2013b. При использовании других версий MATLAB корректность выполнения приведенного кода не гарантируется.

Для написания программ применяется специально разработанный язык программирования, состоящий из команд MATLAB. Отдельные команды, составляющие программу, выполняются системой в режиме интерпретатора, однако пакет MATLAB так же включает компилятор, с помощью которого программа на языке MATLAB может быть преобразована в отдельное приложение.

Оболочка среды MATLAB включает ряд окон, каждое из которых предоставляет определенные возможности при разработке программ:

- **Current folder** – встроенный мини-проводник для работы с файлами в текущей папке, которая может быть выбрана в поле ниже основной панели инструментов (рис. 1.1);
- **Workspace** – окно для отображения имен и диапазонов значений для переменных, которые создаются в памяти по мере выполнения кода программы. Данное окно позволяет просматривать значения

переменных либо непосредственно (для отдельных значений), либо в дополнительном окне, которое может быть открыто путем двойного нажатия левой кнопки мыши на имени переменной;

- **Command History** – окно для просмотра ранее выполненных команд MATLAB;
- **Command Window** – окно для ввода и запуска на выполнение отдельных команд MATLAB в режиме командной строки, а так же для просмотра текстовых результатов выполнения программного кода. Для выполнения какой-либо команды необходимо набрать ее имя в данном окне после символа '`>>`' и нажать «Enter» (рис. 1.2). Если результатом выполнения набранной команды является какая-либо текстовая информация, она будет отображена в окне «Command window». Для предотвращения вывода текстовых результатов в «Command window» необходимо после набранной команды поставить ';'.

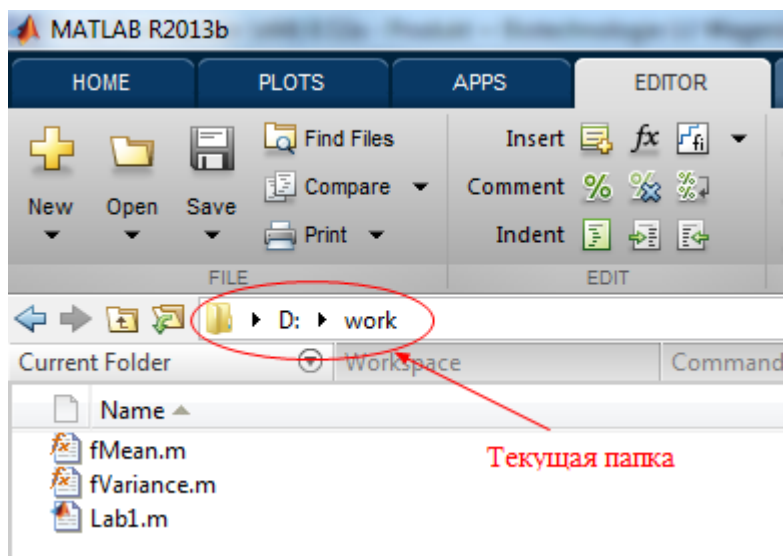


Рис. 1.1. Текущая папка среды MATLAB

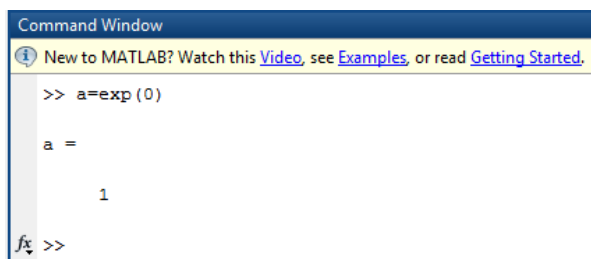


Рис. 1.2. Окно “Command window” среды MATLAB

Для обеспечения индивидуальной настройки среды разработки под требования конкретного пользователя любое из перечисленных выше окон может быть перемещено при помощи мыши в наиболее удобную позицию в рамках оболочки MATLAB, а также скрыто или отображено на экране. Для работы с данными окнами используется группа команд «Layout», расположенная на панели инструментов «Home».

**Написание исходного кода программ.** Несмотря на наличие возможности вводить отдельные команды MATLAB в окне «Command window», такая возможность используется скорее в отладочных целях для проверки результатов небольших промежуточных вычислений. Для создания программ с целью обеспечения сохранения написанного программного кода необходимо данный код размещать в **файлах исходного кода с расширением .m (m-файлах)**. Для создания нового m-файла можно использовать группу команд «New», расположенную на панели инструментов «Home» или «Editor», а для их последующего сохранения группу команд «Save» на панели инструментов «Editor». **Имя m-файла должно включать только латинские буквы, цифры и символ подчеркивания и начинаться обязательно с буквы.** Несоблюдение этих требований приведет к невозможности корректного запуска созданного файла на выполнение. Например, запуск файла с именем 5.m будет всегда приводить к результату «5» вне зависимости от написанного в нем кода.

В зависимости от назначения все m-файлы можно разделить на два вида:

1. **скрипты** – файлы, содержащие последовательность расположенных друг за другом команд языка MATLAB;
2. **функции** – расположенные в отдельных файлах именованные части кода, обычно предназначенные для многократного использования, и, в отличие от скриптов, способные, если это необходимо, принимать входные параметры и возвращать выходные значения.

Для запуска на выполнение кода программы из некоторого m-файла можно поступить одним из следующих способов:

- написать имя m-файла в окне «Command window», после чего нажать кнопку «Enter» на клавиатуре;
- если m-файл в настоящий момент открыт в активной вкладке встроенного редактора исходного кода, то для его запуска на выполнение можно воспользоваться кнопкой «Run» на панели инструментов «Editor» (рис. 1.3).

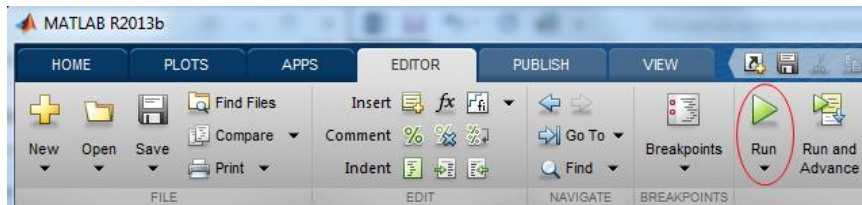


Рис. 1.3. Кнопка «Run» на панели инструментов «Editor»

Чтобы прервать досрочно выполнение программы необходимо перейти в окно «Command window» и нажать клавиши <Ctrl + C>.

При написании функций, которые в дальнейшем будут вызываться из других m-файлов, создается отдельный m-файл, название которого обязательно должно соответствовать имени создаваемой функции. Это правило необходимо соблюдать, поскольку поиск функций в MATLAB выполняется по имени содержащих их файлов. Первой строкой в m-файле, содержащем код функции, должно следовать ее определение, которое в общем виде выглядит следующим образом:

```
function [<возвращаемые_значения>] = <имя_функции>(<входные_параметры>)
```

где входные параметры и возвращаемые значения представляют собой следующие друг за другом через запятую имена переменных. Переменные, представляющие возвращаемые значения, должны быть определены в коде функции. Если функция возвращает только одно значение, то квадратные скобки перед знаком равно можно опустить. Пример функции, вычисляющей значений факториала, имеет следующий вид:

```
function k=fact(n)
    k=1;
    for i=1:n
        k=k*i;
    end
end
```

Пример программного кода, вызывающего функцию вычисления факториала (комментарии в М записываются после символа %):

```
...
%вызов функции 'fact':
% значение входного аргумента 5
% результат сохраняется в переменной 'v'
v = fact(5);
...
```

**Работа с матрицами.** Библиотека команд MATLAB содержит множество готовых функций для выполнения вычислений, данные для которых изначально представляются в виде матриц, а также для текстового и графического вывода полученных результатов на экран.

Необходимо понимать, что MATLAB каждую переменную рассматривает как матрицу и, если специально не указано иное, выполняет операции с данными в матричном виде. Поэлементное умножение, деление и возведение в степень производится путем добавления точки перед символом операции, например, поэлементное умножение матриц *a* и *b* имеет вид: *a.\*b*

Для создания матриц MATLAB предоставляет несколько способов:

- создание вектора-строки из набора значений:

```
a=[1 2 3]
```

- создание вектора-столбца из набора значений:

```
a=[1; 2; 3]
```

- создание прямоугольной матрицы из 2 строк и 3 столбцов:

```
a=[1 2 3; 4 5 6]
```

- построение вектора из значений начиная с 5 до 0 с шагом -2:

```
a=5:-2:0
```

Поскольку при использовании шага -2 в приведенном примере точно 0 получить не удастся, то последним значением в созданном векторе будет 1. Если шаг равен 1, то его можно явно не указывать, а использовать только начальное и конечное значения:

```
a=1:5
```

- вектора и матрицы могут быть так же построены с использованием некоторых встроенных функций, позволяющих одновременно заполнить созданную матрицу определенными значениями.



Получение матрицы 2x3, заполненной 0:

```
a=zeros(2,3)
```

Получение матрицы 2x2, заполненной 1:

```
a=ones(2)
```

Как видно из последнего примера, если в функцию, формирующую матрицу, передается один аргумент, то в результате будет получена квадратная матрица с соответствующим значению данного аргумента числом строк и столбцов.

Кроме представленных выше способов создания матриц, новые матрицы могут быть получены на основе ранее созданных матриц путем выполнения следующих операций:

- объединения строк или столбцов ранее созданных матриц в новую матрицу:

```
a=[1 2; 3 4] %матрица 2x2
```

```
b=[5 6] %матрица 1x2
```

```
c=[7;8;9] %матрица 3x1
```

```
%получаем матрицу r размером 3x2, объединяя строки матриц a и b  
r=[a; b]
```

```
%добавляем в r матрицу c как 3-й столбец, делая r размером 3x3  
r=[r c]
```

- удаления строк или столбцов из ранее созданной матрицы:

```
a=[1 2 3; 4 5 6] %матрица 2x3
```

```
a(:,2)=[] %удаление второго столбца
```

```
a(1,:)=[] %удаление первой строки
```

- выделения части элементов из ранее созданной матрицы в новую матрицу:

```
a=[1 2 3 4; 5 6 7 8; 9 10 11 12] %матрица 3x4
```

```
row2 = a(2,:) %получение второй строки матрицы a
```

```
col3 = a(:,3) %получение третьей колонки матрицы a
```

```
%получение элементов из 1-й, 3-й строк, 2-й, 4-й колонок
```

```
part = a([1 3], [2 4])
```

Из наиболее часто используемых математических операций над матрицами можно выделить следующие:

- транспонирование матрицы:

```
a=[1 2 3] %матрица 1x3
b=a' %b размером 3x1 - результат транспонирования a
```

- сложение и вычитание матриц, имеющих одинаковые размерности:

```
a=[1 2 3] %матрица 1x3
b=[4 5 6] %матрица 1x3

c=a+b %сложение матриц
d=a-b %вычитание матриц
```

- умножение, деление и возведение в степень матриц в соответствии с правилами матричных операций:

```
a=[1 2; 3 4] %матрица 2x2
b=[5 6; 7 8] %матрица 2x2

m=a*b %умножение матриц
d=a/b %деление матриц
p=a^2 %возведение матрицы в степень
```

- поэлементное умножение, деление и возведение в степень матриц, при котором соответствующая операция выполняется над отдельными элементами матриц, с одинаковыми индексами:

```
a=[1 2; 3 4] %матрица 2x2
b=[5 6; 7 8] %матрица 2x2

m=a.*b %умножение матриц
d=a./b %деление матриц
p=a.^2 %возведение матрицы в степень
```

**Работа со строками, вывод информации в Command Window.**  
Строковые константы в MATLAB указываются в одиночных кавычках:

```
str = 'результат:'
```

Поскольку строки рассматриваются как вектора символов, то для их объединения в новую строку, как и для других матриц, можно использовать квадратные скобки. Для преобразования матриц с численными значениями в строковое представление можно использовать функцию **num2str**, а для отображения значения переменной без указания ее имени применяется функция **disp**. Использование перечисленных выше возможностей позволяет, например, формировать текстовую информацию о полученных результатах работы программы с пояснениями:

```

a=[1 2 3];
b=[4 5 6];
disp(['результат: ' num2str(a+b)])

```

В приведенном выше примере символ ';' в конце первых двух строк **предотвращает вывод** полученных в них **результатов** (т.е. в данном примере векторов  $a$  и  $b$ ) в «**Command window**». Последняя строка выводит информацию о результатах вычисления суммы векторов  $a$  и  $b$ .

**Вывод информации в графическом виде.** Для построения простейшего графика на плоскости необходимо использовать функцию **plot** библиотеки MATLAB. В качестве первых двух аргументов данной функции можно использовать вектора одинаковой размерности, задающие координаты по осям  $X$  и  $Y$  для точек, отображаемых на графике. При необходимости можно также задействовать дополнительный третий аргумент, позволяющий задавать строку, содержащую в любой последовательности символы, определяющий цвет, вид маркеров и тип рисуемой линии (рис. 1.4).

```

x=[1 2 3];%значения по оси X
y=[4 5 6];%значения по оси Y

%Символы в строке 'r*--' определяют:
% r - красный цвет линии
% * - вид маркера точек
% -- - тип рисуемой линии
plot(x,y,'r*--');

```

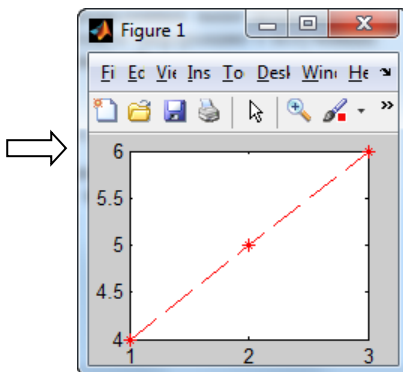


Рис. 1.4. Использование функции plot для построения графика на плоскости

В случае если необходимо на одном графике изобразить одновременно несколько линий, то можно воспользоваться одним из следующих способов:

- задействовать больше аргументов в функции plot, последовательно задающих информацию для рисуемых кривых:

```

%значения по осям X и Y для первой кривой
x1=[1 2 3];
y1=[4 5 6];
%значения по осям X и Y для второй кривой
x2=[2 4 6];
y2=[7 8 9];

% построение обеих кривых
plot(x1,y1,'r*--',x2,y2,'b');

```

- использовать несколько отдельных вызовов функции `plot`, между первым и вторым из которых, задействовать функцию **hold on**:

```

%значения по осям X и Y для первой кривой
x1=[1 2 3];
y1=[4 5 6];
%значения по осям X и Y для второй кривой
x2=[2 4 6];
y2=[7 8 9];

plot(x1,y1,'r*--');%построение первой кривой
hold on;
plot(x2,y2,'b');%построение второй кривой

```

После добавления последнего графика следует отключить режим наложения путем ввода команды **hold off**.

При последовательном рисовании нескольких графиков может возникнуть необходимость отобразить очередную рисуемую кривую в отдельном графическом окне. Для этой цели перед соответствующим вызовом функции `plot` должен быть выполнен вызов функции **figure**:

```

plot(x1,y1);%построение первой кривой
figure
plot(x2,y2);%построение второй кривой

```

Дополнительные сведения об оформлении графиков могут быть найдены в справочной документации.

**Вспомогательные функции и возможности.** Для добавления комментариев в код программы на языке MATLAB необходимо использовать символ `%`. Все, что будет следовать в той же строке за символом `%` будет являться текстом комментария. Для быстрого превращения части содержимого `m`-файла в текст комментария и наоборот можно воспользоваться набором кнопок «**Comment**» на панели инструментов «Editor» (рис. 1.5), предварительно выделив в редакторе кода необходимый фрагмент.

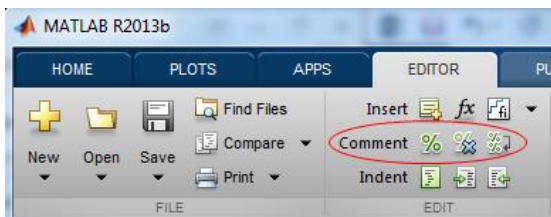


Рис. 1.5. Средства работы с комментариями в MATLAB

Для создания многострочных комментариев в MATLAB используется две пары символов, которые обязательно должны располагаться в отдельных строках программного кода, в которых не должно быть других символов, кроме пробелов. Пара символов `%{` должна располагаться перед началом комментария, а пара символов `%}` после его окончания:

```
%{
Первая строка комментария
Вторая строка комментария
...
%}
```

Для удаления всего содержимого ранее выведенного в окно «**Command window**» используется функция **clc**.

При запуске программ в MATLAB переменные, созданные в памяти в процессе выполнения программного кода, сохраняются и после окончания работы программы, что очень удобно для отладки кода. Для удаления ранее созданных переменных из памяти, например, для предотвращения использования старых значений при очередном запуске программного кода, используется функция **clear all**.

Для закрытия всех ранее открытых графических окон можно использовать функцию **close all**.

Перечисленные выше три функции полезно вызывать в начале программного кода скриптов для автоматического приведения оболочки MATLAB в исходное состояние.

Для работы с справочной системой среды MATLAB можно использовать следующие возможности:

- для быстрого отображения справки по интересующей команде в окне «**Command window**» используется функция **help**, после которой через пробел следует имя того, что необходимо найти. Например, справку по функции `plot` можно получить следующим образом:

```
help plot
```

- для вызова отдельного окна справки можно использовать кнопку **help** или строку поиска в верхней правой части окна среды разработки (рис. 1.6).



Рис. 1.6. Средства для вызова окна справки в оболочке MATLAB

### Краткие сведения по пакету SIMULINK.

SIMULINK представляет собой расширение системы MATLAB и предназначен для моделирования динамических систем, например, таких, как системы радиотехнического профиля, системы регистрации и обработки данных, управления, контроля и т.д. Системы могут быть непрерывными, дискретными или смешанными.

Для запуска SIMULINK можно использовать следующие способы:

- в окне «Command window» среды MATLAB набрать **Simulink** и нажать клавишу «Enter»;
- нажать кнопку «**Simulink Library**» на панели инструментов «Home».

Работа с SIMULINK состоит из трех этапов: создание модели, отладка и экспериментальная апробация модели.

Разработка моделей средствами SIMULINK (S-моделей) основана на технологии drag-and-drop (перетащить и оставить). В качестве «кирпичей» для построения S-моделей используются модули (или блоки), хранящиеся в библиотеке SIMULINK.

Модель представляет собой совокупность блоков, соединенных линиями передачи данных. Манипулируя компонентами библиотек только с помощью мыши, можно строить модели любой сложности. Копирование блоков в создаваемую модель допускается как из встроенных библиотек, так и с любых других библиотек или моделей. После создания и отладки модели запускать ее на выполнение можно как с помощью команд меню SIMULINK, так и из командной строки MATLAB.

Для построения моделей радиофизических систем будет использован узкополосный резонансный фильтр 2-го порядка, характеристики которого задаются коэффициентами блока Zero-Pole библиотеки Continuous. Симметричный фильтр 2-го порядка с единичным коэффициентом усиления на резонансной частоте  $f_0$  имеет передаточную характеристику вида:

$$H(s) = \frac{2\omega_0 \xi s}{(s - s_1)(s - s_2)} = \frac{ks}{(s - s_1)(s - s_2)},$$

где  $s_1 = -\omega_0 \xi + i\omega_0 \sqrt{1 - \xi^2}$ ,  $s_2 = -\omega_0 \xi - i\omega_0 \sqrt{1 - \xi^2}$  — комплексно сопряженные полюса фильтра (корни знаменателя передаточной функции);  $k = 2\omega_0 \xi$  — нормирующий коэффициент;  $\xi = 1/2Q$  — коэффициент затухания,  $Q$  — добротность фильтра;  $s = i\omega$  — комплексная переменная;  $\omega_0 = 2\pi f_0$ .

Представленный фильтр имеет один ноль (корень числителя передаточной характеристики) равный нулю.

### Порядок выполнения

1. Запустить среду разработки MATLAB.
2. С использованием справочной системы MATLAB изучить функции *disp*, *inv*, *plot*, *title*, *xlabel*, *ylabel*, *grid on/off*, *hold on/off*.
3. Получить решение системы линейных уравнений вида  $\mathbf{A} * \mathbf{X} = \mathbf{B}$ , где  $\mathbf{A}$  — квадратная матрица  $2 \times 2$ ,  $\mathbf{B}$  — вектор-столбец размерности 2. Коэффициенты матриц выбрать из табл. 1.1 в соответствии с вариантом задания, выданным преподавателем.


Решение данной системы уравнений можно получить путем умножения матрицы, обратной к  $\mathbf{A}$ , на вектор  $\mathbf{B}$ , причем предварительно следует задать значения матрицы  $\mathbf{A}$  и вектора  $\mathbf{B}$ .

4. Запрограммировать в отдельном m-файле функцию, вычисляющую значение полинома вида  $y = ax^3 + bx^2 + c$ . Функция должна принимать  $x$  в качестве входного аргумента и возвращать значения  $y$  и должна корректно работать в случае, если  $x$  и  $y$  являются матрицами. Коэффициенты полинома  $a$ ,  $b$ ,  $c$  выбрать из табл. 1.2 в соответствии с вариантом задания, выданным преподавателем.
5. В отдельном m-файле написать скрипт, который:
  - вызывает функцию, написанную при выполнении пункта 4, для расчёта значений полинома на интервале значений  $x$  [-5; 5]. Шаг изменения аргумента  $x$  выбрать самостоятельно;
  - строит график зависимости рассчитанных значений полинома от значений аргумента  $x$ ;
  - выполняет оформление внешнего вида построенного графика путем добавления заголовка, подписей для осей координат, нанесения координатной сетки.
6. Запустить SIMULINK и дождаться открытия окна просмотра библиотеки **SIMULINK** — «**Simulink Library Browser**».

7. Изучить содержимое стандартных библиотек блоков: в окне «**Simulink Library Browser**» выбрать подраздел **Simulink** и раскрыть библиотеки **Continuous**, **Math Operations**, **Sources** и **Sinks**.
8. Создать модель «Генератор сигнала и индикатор», приведенной на рис. 1.7:



Рис.1.7. Модель "Генератор сигнала и индикатор"

- Создать рабочее окно **untitled**: в окне «**Simulink Library Browser**» использовать пункт меню **File/ New**.
- В окне «**Simulink Library Browser**» раскрыть библиотеку **Sources**, выделить мышью блок **Signal Generator**. И перетащить его в окно **untitled**.
- В окне «**Simulink Library Browser**» раскрыть библиотеку **Sinks**, выделить мышью блок **Scope** и перетащить его в окно **untitled**.
- Соединить блоки в окне **untitled**, проведя мышью линию от выхода блока **Signal Generator** к входу блока **Scope**, при этом стрелка укажет направление передачи информации.
- Установить параметры отдельных блоков: двойной щелчок по блоку **Signal Generator**, выбрать **Wave form**: sine, установить **Amplitude**: 1 и **Frequency**: 10, выбрать **Units**: Hertz, нажать **OK**; двойной щелчок по блоку **Scope**, щелчок по кнопке , на вкладке **General** установить **Number of axes**: 1, **Time range**: 1, **Tick labels**: all, нажать **OK**.
- Установить параметры моделирования: **Simulation/ Configuration Parameters...** **Simulation time**: **Start Time**: 0.0, **Stop Time**: 20.0/ **Solver options**: **Max Step Size**: 1e-5, **Min Step Size**: 1e-6, **Initial Step Size**: 1e-6, **Relative Tolerance**: 1e-3/ **OK**.
- Запустить модель на выполнение: **Simulation/ Start**. Во время работы можно менять некоторые параметры блоков модели, не останавливая процесс моделирования. Например, дважды щелкнув по блоку **Signal Generator**, можно изменить амплитуду сигнала и/или его частоту. Автоматическое масштабирование амплитуды сигнала в блоке **Scope** реализуется щелчком правой кнопки мыши и выборе команды **Autoscale**.



- Наблюдать график сигнала в зависимости от времени моделирования можно в окне, для отображения которого необходимо дважды щелкнуть левой кнопкой мыши по блоку **Scope**.
- Сохранение модели: **File/Save/** в окне **untitled/** выбрать папку и ввести **Имя файла: Model1/Сохранить**. Автоматически присваивается расширение **.slx**

9. Создание модели «Анализатор спектра периодических сигналов», представленной на рис. 1.8.

Анализатор спектра периодических сигналов производит разложение в ряд Фурье (по пяти составляющим) периодических сигналов (меандр или пилообразный сигнал), затем восстанавливает его по пяти гармоникам.

- Создать новое рабочее окно **untitled: File/ New**.
- Копировать из библиотек нужных блоков, их дублирование и соединение в окне **untitled** производится так же, как и при создании предыдущей модели. При этом дублирование рациональнее производить после установки параметров первого дублируемого блока.

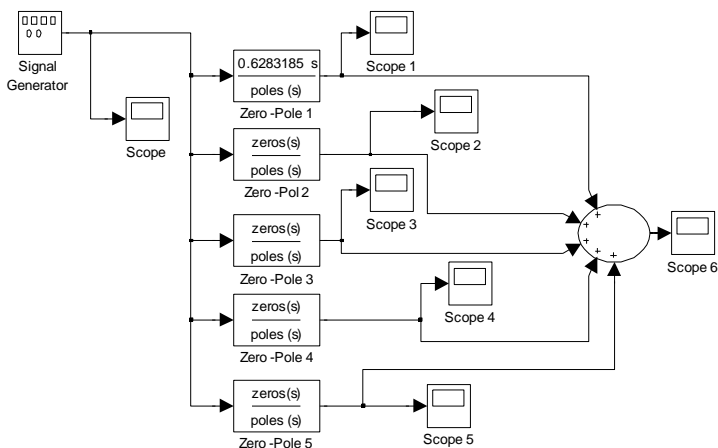


Рис. 1.8. Модель «Анализатор спектра периодических сигналов»

- Установить параметры отдельных блоков модели: для блока **Signal Generator** выбрать **Wave form:** square (меандр), установить **Frequency:**  $f_0$  (выбираются из табл. 1.3 в соответствии с выполня-

емым вариантом), **|Amplitude: 1**; для всех индикаторов **Scope** устанавливаем параметры аналогичные параметрам предыдущей модели; для всех блоков **Zero-Pole** вводим значения параметров при  $\omega_0 = 2\pi f_0$  и  $\xi=0.005$ ; **Zeros:**  $[0]$ , **Poles:**  $[-n * \omega_0 * 0.005 + i * n * \omega_0 * 0.9999875, -n * \omega_0 * 0.005 - i * n * \omega_0 * 0.9999875]$ , **Gain:**  $[n * \omega_0 * 0.01]$ , где  $n = 1, 2, \dots, 5$  – номер блока.

- Установить параметры моделирования, аналогичные параметрам предыдущей модели.
- Запустить на выполнение (**Simulation/ Start**) модель анализатора для сигнала меандр, затем для пилообразного сигнала (**Wave form: sawtooth** в блоке **Signal Generator**). Не забывать про автоматическое масштабирование в блоках **Scope**. Для лучшей визуализации результатов динамического моделирования все шесть графических окон можно разместить без их перекрытия на экране монитора.
- Сохранение модели: **File/Save/** в окне **untitled/** выбрать папку и ввести **Имя файла: Model2/ Сохранить**

**Форма отчета.** Формой отчета являются М-файлы, реализующие задания, представленные в разделе “Порядок выполнения”, а также файлы, содержащие созданные модели SIMULINK.

Таблица 1.1

**Коэффициенты систем линейных уравнений**

Вариант	A	B	Вариант	A	B	Вариант	A	B
1	2 1 3 4	4 11	9	1 4 2 -2	13 6	17	2 3 2 1	5 3
2	2 2 2 4	16 26	10	-1 1 3 1	2 6	18	2 2 1 2	4 3
3	4 4 4 8	24 40	11	-5 2 1 1	20 3	19	1 -4 3 2	-5 13
4	5 1 4 4	10 24	12	4 1 8 -1	7 29	20	1 3 2 1	6 7
5	4 1 1 1	24 9	13	2 3 1 1	5 2	21	4 3 2 1	15 7
6	1 2 1 1	10 9	14	2 3 3 2	8 7	22	7 2 4 1	23 11
7	4 -1 2 1	-6 0	15	2 3 1 1	2 1	23	-2 2 1 6	-4 23
8	5 -1	14	16	3 -3	6	24	9 -5	28

Вариант	A	B	Вариант	A	B	Вариант	A	B
	1 -1	6		-2 1	-5		3 7	-8

Таблица 1.2

### Параметры функций

Вариант	$a$	$b=c$	Вариант	$a$	$b=c$	Вариант	$a$	$b=c$
1	-0.5	2	9	5	-2	17	11	3
2	-0.5	-2	10	3	10	18	3	8
3	-0.5	10	11	8	-10	19	7	5
4	-0.5	-10	12	5	-40	20	2	9
5	-0.5	-40	13	9	-20	21	0.8	7
6	-0.5	-20	14	7	-60	22	0.5	-10
7	0.5	-60	15	12	1	23	1	-20
8	1	2	16	14	5	24	5	-40

Таблица 1.3

### Частота генерируемых сигналов

Вариант	$f_0$	Вариант	$f_0$
1	10	13	24
2	12	14	9
3	15	15	14
4	20	16	16
5	25	17	3
6	8	18	18
7	5	19	21
8	7	20	3
9	6	21	8
10	17	22	13
11	4	23	22
12	2	24	18

## ЛАБОРАТОРНАЯ РАБОТА № 2

### ИЗУЧЕНИЕ ВСТРОЕННЫХ СРЕДСТВ MATLAB ДЛЯ МОДЕЛИРОВАНИЯ И ОЦЕНКИ СЛУЧАЙНЫХ ВЕЛИЧИН

**Цель работы.** Практическое освоение встроенных средств MATLAB для моделирования случайных величин, изучение алгоритмов первичной обработки статистических данных, исследование основных законов распределения случайных величин.

**Краткие сведения.** Под случайной величиной понимается величина, которая в результате опыта со случайным исходом принимает то или иное значение [2].

Случайные величины могут быть дискретными и непрерывными. Дискретной случайной величиной называется случайная величина, заданная на счетном наборе значений, каждому из которых поставлена в соответствие некоторая вероятность его появления. Вероятности, определенные для множества значений дискретной случайной величины, подчиняются условию нормировки:  $\sum_1^N p_i = 1$ , где  $N$  количество возможных значений (может быть бесконечным) а  $p_i$  вероятность  $i$ -го значения.

Общим способом задания закона распределения вероятностей для случайной величины является использование функции распределения, которая представляет собой вероятность того, что случайная величина  $\xi$  примет значение меньшее, чем некоторое заданное  $x$  (выражение 2.1):

$$F(x_i) = P\{\xi < x_i\}. \quad (2.1)$$

Функция распределения  $F(x)$  произвольной случайной величины есть неубывающая функция своего аргумента, значения которой заключены между нулем и единицей; причем  $F(-\infty) = 0$ , а  $F(+\infty) = 1$  [2]. Функция распределения дискретной случайной величины представлена разрывной ступенчатой функцией, скачки которой происходят в точках, соответствующих возможным значениям случайной величины, и равны вероятностям этих значений. Сумма всех скачков функции распределения равна единице. Значение данной функции для аргумента  $x$  это сумма вероятностей всех значений дискретной случайной величины, меньших  $x$ .

Распределение дискретной случайной величины однозначно задаётся рядом распределения вероятностей:

$$P\{\xi = x_i\} = p_i, \quad i = 1, 2, \dots \quad (2.2)$$

Ряд распределения вероятностей случайной величины  $\xi$  может быть задан в виде таблицы, в верхней строке которой перечислены в порядке возрастания все возможные значения случайной величины, а в нижней – вероятности этих значений.

$$\xi: \begin{array}{c|ccccc} & x_1 & x_2 & \dots & x_n & \dots \\ \hline & p_1 & p_2 & \dots & p_n & \dots \end{array}$$

Случайная величина  $\xi$  называется непрерывной, если она принимает значения на некоторой непрерывной, в общем случае бесконечной, области, на которой задано распределение вероятностей. Для задания закона распределения непрерывной случайной величины может использоваться ее функция распределения  $F(x)$  (формула 2.1) (другие названия: интегральная, или кумулятивная функция распределения), которая в данном случае является непрерывной, монотонно возрастающей функцией (смотри 2.3)

$$F(x) = P\{\xi < x\} = P\{-\infty < \xi < x\}. \quad (2.3)$$

Функция распределения непрерывной случайной величины не только непрерывна в любой точке, но и дифференцируема всюду, кроме, может быть, отдельных точек, где она терпит излом.

Кроме интегральной функции распределения для задания закона распределения непрерывной случайной величины можно использовать функцию плотности распределения. Плотностью распределения  $f(x)$  непрерывной случайной величины  $\xi$  в точке  $x$  называется производная ее функции распределения в этой точке:

$$f(x) = F'(x) = \frac{dF(x)}{dx}. \quad (2.4)$$

Для нормированной непрерывной случайной величины выполняется следующее соотношение (условие нормировки):

$$\int_{-\infty}^{\infty} f(x)dx = 1. \quad (2.5)$$

Для вычисления интегральной функции распределения на базе известной плотности  $f(x)$  используется следующее выражение:

$$F(x) = \int_{-\infty}^x f(x)dx. \quad (2.6)$$

Закон распределения полностью описывает случайную величину с вероятностной точки зрения. Но часто достаточно указать только отдельные числовые характеристики случайной величины, которые позволяют в сжатой форме выразить наиболее существенные черты распределения. Среди

таких характеристик можно выделить математическое ожидание, моду, медиану, дисперсию и среднеквадратическое отклонение [2].

*Математическое ожидание*  $\mu$  – это среднее взвешенное значение случайной величины, в которое ее значение входит с "весом", равным соответствующей вероятности. Математическое ожидание дискретной случайной величины вычисляется по формуле:

$$\mu = \sum_i x_i p_i. \quad (2.7)$$

Математическое ожидание непрерывной случайной величины – соответственно по формуле:

$$\mu = \int_{-\infty}^{\infty} x f(x) dx. \quad (2.8)$$

Для непрерывных случайных величин, определенных на конечном интервале значений, плотность  $f(x)$  вне этого интервала равна 0, и по этой причине интегрирование в выражении (2.8) выполняется на конечной области определения.

*Модой* случайной величины называют ее наиболее вероятное значение, т. е. значение, для которого вероятность  $p_i$  или плотность распределения  $f(x)$  достигает максимального значения.

*Медиана* – это значение случайной величины, для которого вероятности появления значений меньших и больших медианы одинаковы и равны по 1/2.

*Дисперсия*  $\sigma^2$  случайной величины есть математическое ожидание квадрата разности случайной величины и ее математического ожидания. Дисперсия имеет размерность квадрата случайной величины и характеризует степень разброса случайной величины относительно ее математического ожидания. Дисперсия дискретной случайной величины вычисляется по формуле

$$\sigma^2 = \sum_i (x_i - \mu)^2 p_i. \quad (2.9)$$

Для непрерывной случайной величины используется формула

$$\sigma^2 = \int_{-\infty}^{\infty} (x - \mu)^2 f(x) dx. \quad (2.10)$$

Для непрерывных случайных величин, определенных на конечном интервале значений, по тем же причинам, что и для математического ожидания интегрирование в выражении (2.10) выполняется на конечной области определения.

*Среднеквадратическим отклонением  $\sigma$*  случайной величины называется квадратный корень из ее дисперсии. Среднеквадратическое отклонение имеет размерность случайной величины.

Из непрерывных случайных величин наиболее известны и применяются в практике нормально распределенная и равномерно распределенная в интервале  $[0, 1]$  случайные величины. Важная роль нормального распределения во многих областях науки и техники следует из центральной предельной теоремы теории вероятностей. Если результат наблюдения является суммой многих случайных слабо взаимосвязанных величин, каждая из которых вносит малый вклад относительно общей суммы, то при увеличении числа слагаемых распределение результата стремится к нормальному. Равномерно распределенная на интервале  $[0, 1]$  случайная величина называется иначе базовой случайной величиной, так как для нее разработаны генераторы псевдослучайных последовательностей, доступные в большинстве инструментальных средств разработки программных продуктов, и на ее основе можно получить широкий спектр других случайных величин (а также случайных векторов, процессов и потоков). Из дискретных случайных величин наибольшее распространение (например, в теории массового обслуживания) получила случайная величина, распределённая по закону Пуассона. Ниже эти случайные величины рассматриваются подробнее.

**Равномерно распределенная в интервале  $[a, b]$  случайная величина.** Случайная величина имеет непрерывное равномерное распределение на отрезке  $[a, b]$ , в частном случае на отрезке  $[0, 1]$ , если её плотность  $f(x)$  имеет вид:

$$f(x) = \begin{cases} 1/(b - a), & x \in [a, b] \\ 0, & x \notin [a, b] \end{cases} \quad (2.11)$$

а функция распределения – соответственно:

$$F(x) = \begin{cases} 0, & x < a \\ (x - a)/(b - a), & a \leq x < b. \\ 1, & x \geq b \end{cases} \quad (2.12)$$

Числовые характеристики равномерно распределенная непрерывной случайной величины на интервале  $[a, b]$  представлены ниже в табл. 2.1.

Таблица 2.1

**Числовые характеристики равномерного распределения непрерывной случайной величины на интервале  $[a, b]$**

Числовая характеристика	$[a, b]$	$[0, 1]$
Математическое ожидание	$\mu = (b + a)/2$	1/2
Дисперсия	$\sigma^2 = (b - a)^2/12$	1/12
Медиана	$(b + a)/2$	1/2
Мода	Любое значение из $[a, b]$	Любое значение из $[0, 1]$

В рамках библиотеки MATLAB можно выделить следующие функции, связанные с равномерным распределением:

- **rand** генерирует случайные числа, равномерно распределенные на интервале  $[0, 1]$ , так называемую базовую случайную величину. **rand(M, N)** генерирует массив размера  $M \times N$ , элементами которого являются реализации базовой случайной величины. Функция **rand** без аргументов формирует одну такую реализацию, которая изменяется при каждом последующем вызове.
- **unifrnd(A,B,M,N)** генерирует  $M \times N$  равномерно распределенных на интервале  $[A, B]$  случайных чисел.
- **unifcdf(x,A,B)** возвращает значение (кумулятивной, в английском cumulative distribution function, cdf) функции распределения для непрерывного равномерного распределения на интервале  $[A, B]$ .
- **unifpdf(x,A,B)** возвращает значение функции плотности распределения (в английском probability distribution function, pdf) для непрерывного равномерного распределения на интервале  $[A, B]$ .
- **unifinv(p,A,B)** возвращает значение квантиля равномерно распределенной на интервале  $[A, B]$  случайной величины для вероятности  $p$ .

**Нормальная случайная величина.** Случайная величина имеет нормальное распределение с математическим ожиданием  $\mu$  и среднеквадратическим отклонением  $\sigma$ , если её плотность  $f(x)$  задается следующим выражением (функцией Гаусса):

$$f(x) = \frac{1}{\sigma\sqrt{2\pi}} \exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right), -\infty \leq x \leq \infty. \quad (2.13)$$

Функция распределения нормальной случайной величины записывается следующим образом



$$F(x) = \frac{1}{2} \left[ 1 + \operatorname{erf} \left( \frac{x-\mu}{\sqrt{2}\sigma} \right) \right], \quad (2.14)$$

где  $\operatorname{erf}$  – функция ошибок.

Числовые характеристики нормального распределения представлены в табл. 2.2.

Таблица 2.2

**Числовые характеристики нормально распределенной непрерывной случайной величины с математическим ожиданием  $\mu$  и среднеквадратическим отклонением  $\sigma$**

Числовая характеристика	
Математическое ожидание	$\mu$
Дисперсия	$\sigma^2$
Медиана	$\mu$
Мода	$\mu$

Функции MATLAB, связанные с нормальным распределением:

- **randn** генерирует случайные числа, распределенные по нормальному закону с математическим ожиданием 0 и среднеквадратическим отклонением 1 (так называемого стандартизированного нормального распределения  $N(0,1)$ ). **randn(M, N)** генерирует матрицу реализаций стандартизированной нормально распределенной случайной величины размера  $M \times N$ . Функция **randn** без аргументов формирует одно такое случайное число;
- **normrnd(MU,SIGMA,M,N)** генерирует матрицу реализаций нормально распределенной случайной величины с математическим ожиданием MU и среднеквадратическим отклонением SIGMA размера  $M \times N$ .
- **normcdf(x,MU,SIGMA)** возвращает значение (кумулятивной) функции распределения нормально распределенной случайной величины с математическим ожиданием MU и среднеквадратическим отклонением SIGMA;
- **normpdf(x,MU,SIGMA)** возвращает значение функции плотности распределения нормально распределенной случайной величины с математическим ожиданием MU и среднеквадратическим отклонением SIGMA;
- **norminv(p,MU,SIGMA)** возвращает значение квантиля нормально распределенной случайной величины с математическим

ожиданием  $\mu$  и среднеквадратическим отклонением для вероятности  $p$ .

**Для справки:** реализации нормальной случайной величины с математическим ожиданием  $\mu$  и среднеквадратическим отклонением  $\sigma$  можно получить из реализаций стандартизированной нормальной случайной величины с математическим ожиданием 0 и среднеквадратическим отклонением 1 с помощью простого соотношения:

$$N(\mu, \sigma) = N(0,1) * \sigma + \mu. \quad (2.15)$$

**Дискретная случайная величина, распределённая по закону Пуассона.** Пуассоновская случайная величина  $\xi$  представляет собой число событий, произошедших за фиксированное время, при условии, что данные события происходят независимо друг от друга с некоторой фиксированной средней интенсивностью  $\lambda$ . Вероятности  $P(k) = P\{\xi = k\}$  для значений случайной величины, распределённой по закону Пуассона, имеют вид:

$$P(k) = P\{\xi = k\} = \frac{\lambda^k}{k!} e^{-\lambda}, \quad k = 0, 1, 2, \dots \quad (2.16)$$

Функция распределения Пуассоновской случайной величины записывается следующим образом

$$F(k) = P\{\xi < k\} = \frac{\Gamma(k+1, \lambda)}{k!}, \quad (2.17)$$

где  $\Gamma$  – гамма функция.

Числовые характеристики распределения Пуассона представлены в табл. 2.3.

Таблица 2.3

**Числовые характеристики дискретного распределения Пуассона с параметром  $\lambda$**

Числовая характеристика	
Математическое ожидание	$\lambda$
Дисперсия	$\lambda$
Медиана	$\approx \lambda + \frac{1}{3} - 0.02\lambda$
Мода	$\lambda$

Функции MATLAB, связанные с распределением Пуассона:

- **poissrnd** генерирует случайные числа, распределенные по закону Пуассона. **poissrnd(LAMBDA,M,N)** генерирует массив размера  $M \times N$  реализаций случайной величины, распределенной по закону Пуассона с параметром LAMBDA.
- **poisscdf(x,LAMBDA)** возвращает значение (кумулятивной) функции распределения Пуассоновской случайной величины с параметром LAMBDA.
- **poisspdf(x,LAMBDA)** возвращает вероятность значения  $x$  для Пуассоновской случайной величины с параметром LAMBDA.
- **poissinv(p,LAMBDA)** возвращает значение Пуассоновской случайной величины с параметром LAMBDA, для которого функция распределения равна  $p$ .

### Некоторые другие функции MATLAB, связанные с генерацией случайных величин:

- **rng** функция контроля генераторов RAND, RANDI, RANDN. Позволяет задать стартовое значение (seed) и тип генератора.  
**rng('default')** сбрасывает стартовое значение генератора в значение по умолчанию (0).  
**rng('shuffle')** устанавливает стартовое значение генератора, вычисленное на основе текущего времени.  
**rng(seed,generator)** позволяет дополнительно к стартовому значению определить и тип генератора. Аргумент generator может принять следующие значения:
 

'twister'	Mersenne Twister
'combRecursive'	Combined Multiple Recursive
'multFibonacci'	Multiplicative Lagged Fibonacci
'v5uniform'	MATLAB 5.0 uniform generator
'v5normal'	MATLAB 5.0 normal generator
'v4'	MATLAB 4.0 generator
- **random** генерирует случайные числа, распределенные по заданному закону распределения. **random(NAME,A,M,N)** генерирует массив размера  $M \times N$  реализаций случайной величины с заданным однопараметрическим законом распределения. **random(NAME,A,B,M,N)** генерирует массив размера  $M \times N$  реализаций случайной величины с заданным двухпараметрическим законом распределения. **random(NAME,A,B,C,M,N)** генерирует массив размера  $M \times N$  реализаций случайной величины с заданным

трехпараметрическим законом распределения размера  $M \times N$ . Аргумент NAME может принять следующие значения:

'beta'	или	'Beta',
'bino'	или	'Binomial',
'chi2'	или	'Chisquare',
'exp'	или	'Exponential',
'ev'	или	'ExtremeValue',
'f'	или	'F',
'gam'	или	'Gamma',
'gp'	или	'GeneralizedPareto',
'geo'	или	'Geometric',
'hyge'	или	'Hypergeometric',
'logn'	или	Lognormal',
'nbin'	или	'NegativeBinomial',
'poiss'	или	'Poisson',
'rayl'	или	'Rayleigh',
't'	или	'T',
'unif'	или	'Uniform',
'unid'	или	'Discrete Uniform',
'wbl'	или	'Weibull'.

- **randi** генерирует дискретные числа с равномерным распределением вероятностей. **randi**([I<sub>MIN</sub> I<sub>MAX</sub>],M,N) генерирует  $M \times N$  дискретных чисел в диапазоне [I<sub>MIN</sub> I<sub>MAX</sub>].
- **unidrnd**(A,M,N) генерирует  $M \times N$  дискретных случайных чисел с равномерным распределением вероятностей в интервале [1,A].
- **unidcdf**(x,K) возвращает значение функции распределения для дискретного равномерного распределения, принимающего значения 1,2,...,K.
- **unidpdf**(x,K) возвращает вероятность значения  $x$  для дискретного равномерного распределения, принимающего значения 1,2,...,K.
- **unidinv**(p,K) возвращает значение дискретной равномерно распределённой случайной величины, принимающей значения 1,2,...,K, для которого функция распределения равна  $p$ .

### Функции MATLAB раздела анализ данных

- **max** максимальное значение;
- **min** минимальное значение;
- **mean** среднее значение;
- **median** медиана;

- **std** среднеквадратическое отклонение;
- **sort** сортировка;
- **sum** сумма элементов;
- **prod** произведение элементов;
- **cumsum** кумулятивная сумма;
- **cumprod** кумулятивное произведение;
- **diff** попарные разности  $n$  и  $(n-1)$  элементов;
- **corrcoef** матрица корреляционных коэффициентов;
- **cov** для вектора - дисперсия, для матрицы - ковариационная матрица.

Если аргументом этих функций является матрица, то вычисления производятся по столбцам. Например,  $\min(A)$  дает вектор-строку минимальных значений для каждого столбца матрицы  $A$ . Если же аргумент - вектор, то не делается различия по его ориентации.

### Порядок выполнения

1. С использованием справочной системы MATLAB изучить функции генерации случайных величин и анализа данных (см. выше).
2. Провести моделирование вектора (размера  $1 \times M$ ) из  $M$  реализаций непрерывной случайной величины в соответствии с вариантом задания (смотри табл. 2.4). Для выполнения задания воспользоваться встроенными в MATLAB функциями, приведенными в разделе «краткие сведения» для соответствующей случайной величины.
3. Провести моделирование вектора (размера  $1 \times M$ ) из  $M$  реализаций дискретной случайной величины, распределенной по закону Пуассона с параметром  $\lambda$ , определенным вариантом задания (смотри табл. 2.4). Для выполнения задания воспользоваться встроенной в MATLAB функцией `poissrnd`.
4. Провести первичную статистическую проверку для каждой из сгенерированных случайных последовательностей. Для этого:
  - Вывести на экран  $K$  (определено вариантом в табл. 2.4) реализаций случайной величины. В MATLAB для этого требуется определить диапазон индексов вектора, содержащего полученные реализации случайной величины. Например, 100 чисел с 20-го до 119-го, хранящихся в переменной `x_rand`, можно вывести на экран следующей командой: `x_rand(20:119)`.
  - Вычислить минимальное и максимальное значение. Для выполнения воспользуйтесь функциями `min`, `max`.

- Получить оценку математического ожидания тестируемой случайной величины и сравнить ее с теоретическим математическим ожиданием. Для выполнения воспользуйтесь функцией `mean`.
- Исследовать для непрерывной случайной величины, как по мере увеличения статистики (объема выборки  $L$ ) оценка математического ожидания стремится к теоретическому математическому ожиданию. Для этого необходимо построить график зависимости оценки математического ожидания от  $\log_{10}(L)$ , на который нанести соответствующие теоретические значения (смотри раздел «краткие сведения»). Выборки случайных величин объемом  $L = 32, 316, 1000, 3162, M$  значений необходимо получать из ранее сгенерированной генеральной совокупности, после чего необходимо вычислить оценки математического ожидания для каждой выборки и построить график.
- Получить оценку среднеквадратического отклонения тестируемой случайной величины и сравнить ее с теоретическим среднеквадратическим отклонением. Для выполнения воспользуйтесь функцией `std`.
- На одном графике построить нормированную гистограмму тестируемой случайной величины и теоретическую плотность распределения вероятности (для дискретной случайной величины просто распределение вероятностей). Для расчета гистограммы воспользуйтесь функцией `hist`, а для построения графика нормированной гистограммы – функцией `bar`. Для расчета теоретической плотности (для дискретной случайной величины просто распределения вероятностей) воспользуйтесь соответствующей функцией библиотеки MATLAB (смотри раздел «краткие сведения»), а для построения ее графика – функцией `plot`.
- Для непрерывной случайной величины на одном графике построить нормированную кумулятивную гистограмму (функцию накопленных вероятностей) тестируемой случайной величины и теоретическую функцию распределения вероятностей. Для расчета и построения кумулятивной гистограммы воспользуйтесь функциями `hist`, `cumsum` и `stairs`. Для расчета теоретической функции распределения вероятностей воспользуйтесь соответствующей функцией библиотеки MATLAB (смотри раздел «краткие сведения»), а для построения ее графика – функцией `plot`.

**Форма отчета.** М-файлы, реализующие задание.

Таблица 2.4

## Информация к вариантам заданий

Вариант	М	К	Закон распределения	$\lambda$
1	8500	10	$N(-10, 0.5)$	9.5
2	8000	12	Равномерное на $[0, 1]$	0.5
3	11500	12	$N(9, 2)$	4.5
4	12250	14	Равномерное на $[0, 1]$	3.5
5	11000	12	$N(-3, 2)$	3.0
6	12000	10	Равномерное на $[0, 1]$	5.5
7	7000	16	$N(-1, 1)$	7.5
8	6500	10	Равномерное на $[0, 1]$	12.5
9	10500	14	$N(4, 1.5)$	6.5
10	9750	18	Равномерное на $[0, 1]$	12.0
11	10750	20	$N(-5, 1)$	7.0
12	10000	16	Равномерное на $[0, 1]$	11.5
13	11250	20	$N(13, 2)$	11.0
14	8250	12	Равномерное на $[0, 1]$	8.5
15	6750	18	$N(12, 1.2)$	5.0
16	7250	18	Равномерное на $[0, 1]$	2.0
17	9250	16	$N(11, 1)$	2.5
18	9000	10	Равномерное на $[0, 1]$	10.5
19	11750	18	$N(0, 1.5)$	8.0
20	6250	14	Равномерное на $[0, 1]$	1.0
21	10250	20	$N(7, 1)$	9.0
22	7500	15	Равномерное на $[0, 1]$	1.5
23	7750	20	$N(1, 2)$	4.0

Вариант	М	К	Закон распределения	$\lambda$
24	9500	14	Равномерное на [0, 1]	10.0
25	8750	16	$N(-7, 2)$	6.0

**Пример выполнения задания в Matlab.**

Генерация случайной величины и проведение первичной статистической проверки:

```

clc
clear all
close all

x = unidrnd(100,1,10000); %генерация случайной величины
disp('First 20 values:');
disp(x(1:20));           %отображение первых 20 значений

disp(['Range [min max]: [' num2str(min(x)) ' '
num2str(max(x)) '];']); %Вычисление минимального и макси-
мального значений

```

Расчет основных числовых характеристик дискретной случайной величины (принимающей значения от 1 до 100 с равной вероятностью) и их оценок

```

x_val = 1:100;           %генерация вектора возможных зна-
чений случайной величины
p = 1/100;               %вычисление вероятности

mu = sum(x_val*p);       %расчет математического ожидания
avg_val = mean(x);       %вычисление оценки математиче-
ского ожидания (выборочного среднего)
disp(['Mean: theory = ' num2str(mu) '; estimation = '
num2str(avg_val) '']);

sigma = sqrt(sum((x_val-mu).^2*p)); %расчет среднеквад-
ратического отклонения

std_val = std(x);        %получение оценки среднеквадратического
отклонения (или std_val = sqrt(var(x)))
disp(['St. dev.: theory = ' num2str(sigma) '; estimation = '
' num2str(std_val) '']);

median_theor = floor((100+1)/2); %вычисление медианы
med_val = median(x);      %получение оценки медианы

```



```
disp(['Median: theory = ' num2str(median_theor) ' ; estimation = ' num2str(med_val) ' ;']);

disp(['Mode: theory = any value from 0 - 100; ' 'estimation = ' num2str(mode(x)) ' ;']); %получение оценки моды
```

Построение гистограммы. Для дискретной случайной величины необходимо задать вектор карманов в виде дискретных значений, соответствующий ряду возможных значений случайной величины.

```
figure; %указание Matlab строить график в новом окне
hist_x = 1:max(x); %генерация вектора карманов
hist(x, hist_x) %построение гистограммы.
```

Для непрерывной случайной величины необходимо предоставить вектор центров карманов (для функции hist). Например:

```
a=floor(min(x)); %задание минимального значения для гистограммы (если известна конечная левая граница области определения непрерывной случайной величины, то необходимо использовать ее)
b=ceil(max(x)); %задание максимального значения для гистограммы (если известна конечная правая граница области определения непрерывной случайной величины, то необходимо использовать ее). Для бесконечных распределений следует использовать min(), max() с осторожностью, так как не исключена генерация очень маленького и очень большого значений, что приведет к попаданию большинства сгенерированных значений в один (или несколько) крайних карманов гистограммы. В этом случае следует самостоятельно задавать a и b, исходя из сведений о возможных значениях случайной величины (например, применяя правило 3σ для нормального распределения).
x_step = (b-a)/20; %вычисление величины кармана (шага)
x_hist = a+x_step/2:x_step:b-x_step/2; %получение вектора центров карманов
hist(x, x_hist) %построение гистограммы
```

Построение нормированной гистограммы случайной величины и распределения вероятности (на примере равномерного дискретного распределения)

```
figure;
hist_estim = hist(x, x_hist); %получение вектора высот гистограммы (график не выводится, если слева от hist встречается оператор присвоения)
hist_estim_norm = hist_estim/length(x); %пересчет частоты выпадения события к вероятности (нормировка)
```

```

pdf_theory = unidpdf(x_hist, 100);      %получение вектора
теоретических значений вероятностей
bar(x_hist,hist_estim_norm)             %построение
столбиковой диаграммы
hold on;
plot(x_hist, pdf_theory, 'b');           %построение распре-
деления вероятностей
hold off;
title('PDF for uniform discrete distribution');
xlabel('value');
ylabel('PDF');
grid on;

```

Построение оценки функции распределения (эмпирического закона распределения, на примере равномерного дискретного распределения)

```

figure;
hist_cum = cumsum(hist_estim_norm);      вычисление функции
накопленных вероятностей. Для непрерывной случайной вели-
чины ее необходимо домножить на величину кармана.
cdf_theory = unidcdf(x_hist, 100);       получение вектора
теоретических значений
hold on
stairs(x_hist, hist_cum, 'r');
plot(x_hist, cdf_theory, 'b');
hold off;
title('CDF for uniform discrete distribution');
xlabel('value');
ylabel('CDF');
grid on;

```

## ЛАБОРАТОРНАЯ РАБОТА № 3

### МОДЕЛИРОВАНИЕ НЕПРЕРЫВНОЙ СЛУЧАЙНОЙ ВЕЛИЧИНЫ МЕТОДОМ ОБРАТНЫХ ФУНКЦИЙ

**Цель работы.** Практическое освоение алгоритма программной генерации непрерывной случайной величины методом обратных функций и методов статистической проверки разработанного генератора.

#### Краткие сведения

**Метод обратных функций.** Предположим, что случайная величина  $\xi$  определена на конечном или бесконечном интервале  $[a, b]$  и имеет на этом интервале плотность распределения  $f(x) > 0$ . Пусть

$$F(x) = \int_a^x f(u) du, x \in [a; b] \quad (3.1)$$

функция распределения случайной величины  $\xi$ . Тогда, если получена реализация  $z$  случайной величины  $\alpha$ , равномерно распределенной на  $[0; 1]$ , выборочное значение случайной величины  $\xi$  может быть определено как корень уравнения [3, 4]:

$$F(x) = z. \quad (3.2)$$

В тех случаях, когда уравнение (3.2) аналитически разрешимо относительно  $x$ , получается явная формула:

$$x = G(z) \quad (3.3)$$

для разыгрывания выборочного значения  $\xi$ , где функция  $G$  является обратной для функции  $F$ . В других случаях решение (3.2) может быть найдено приближенно или для разыгрывания  $\xi$  следует применить другие методы.

#### Примеры:

1. Случайная величина  $\xi$  распределена равномерно на интервале  $[a; b]$ , т.е. ее функция распределения имеет вид:

$$F(x) = (x - a)/(b - a), x \in [a; b].$$

Для получения алгоритма ее генерации в соответствии с выражением (3.2) необходимо решить уравнение:

$$(x - a)/(b - a) = z,$$

откуда следует окончательная формула для генератора:

$$x = a + (b - a)z.$$

2. Случайная величина  $\xi$  определена при  $x > a$  с плотностью:

$$f(x) = \lambda \cdot \exp(-\lambda(x - a)).$$

В этом случае, с учетом выражения (3.1), уравнение (3.2) принимает вид:

$$1 - \exp(-\lambda(x - a)) = z,$$

откуда получаем:

$$x = a - \ln(1 - z)/\lambda.$$

Поскольку случайные величины  $\alpha$  и  $(1 - \alpha)$  распределены одинаково, то последнее выражение приобретает следующий окончательный вид:

$$x = a - \ln(z)/\lambda.$$

**Оценка математического ожидания.** Для построения оценки математического ожидания случайной величины  $\xi$  по сгенерированному набору ее реализаций  $x_i, i = 1, 2, \dots, N$  используется выражение вида:

$$m_N^* = \frac{1}{N} \sum_{i=1}^N x_i. \quad (3.4)$$

**Доверительный интервал для математического ожидания.** Доверительным интервалом для математического ожидания  $\mu$  случайной величины  $\xi$  является интервал  $[\varphi_1; \varphi_2]$ , в который значение  $\mu$  попадает с некоторой доверительной вероятностью  $\beta$ , т.е.  $P\{\varphi_1 < \mu < \varphi_2\} = \beta$ . Если для случайной величины  $\xi$  сгенерирована ее выборка достаточно большого объема  $N$  ( $N > 30$ ), то границы доверительного интервала для ее математического ожидания можно приближенно рассчитать следующим образом:

$$[m_N^* - \frac{y_\beta \sigma_\xi}{\sqrt{N}}; m_N^* + \frac{y_\beta \sigma_\xi}{\sqrt{N}}], \quad (3.5)$$

где  $m_N^*$  – оценка математического ожидания, полученная по формуле (3.4),  $\sigma_\xi$  – среднеквадратическое отклонение, рассчитанное на базе известного закона распределения случайной величины  $\xi$  (для непрерывных случайных величин смотри выражение (2.10)),  $y_\beta$  – корень интегрального уравнения вида:

$$\frac{2}{\sqrt{2\pi}} \int_0^{y_\beta} \exp(-\frac{t^2}{2}) dt = \beta. \quad (3.6)$$

Найти  $y_\beta$  для любого  $\beta$  можно из обратной функции стандартизированного нормального распределения *norminv*, реализованной в MATLAB:  $y_\beta = \text{norminv}(1 - \alpha/2)$ , где  $\alpha = 1 - \beta$ . Для доверительной вероятности  $\beta = 0.95$  значение  $y_\beta = 1.96$ .

**Оценка дисперсии.** В качестве оценки дисперсии случайной величины  $\xi$  по сгенерированному набору ее реализаций  $x_i, i = 1, 2, \dots, N$  можно использовать выражение:

$$D_N^* = \frac{1}{N-1} \sum_{i=1}^N x_i^2 - \frac{1}{N(N-1)} (\sum_{i=1}^N x_i)^2 \quad (3.7)$$

**Построение эмпирической плотности распределения.** Эмпирическая плотность распределения строится по сгенерированной выборке для случайной величины  $\xi$ . Расчет значений эмпирической плотности распределения выполняется в соответствии со следующим выражением:

$$f_j^e = \frac{v_j}{N\Delta x}, \quad (3.8)$$

где  $v_j$  – значение гистограммы, построенной по сгенерированной выборке, для  $j$ -го кармана (количество реализаций из сгенерированной выборки, попавших в  $j$ -й карман),  $N$  количество сгенерированных значений (объем выборки),  $\Delta x$  – ширина кармана,  $j = 1, \dots, M$ ,  $M$  – количество карманов, выделяемых с шагом  $\Delta x$  на области определения  $[a, b]$  случайной величины  $\xi$ . Другими словами, оценкой плотности распределения случайной величины является ее нормированная гистограмма, вычисленная по сгенерированной выборке.

### Порядок выполнения

1. С использованием справочной системы MATLAB изучить функции *var*, *integral*, *norminv*.
2. Разработать алгоритм моделирования заданной (см. ниже табл. 3.1) непрерывной случайной величины и программно его реализовать в среде MATLAB.
3. Провести тестовую проверку созданного генератора:
  - получить оценку математического ожидания генерируемой случайной величины двумя способами: с использованием выражения (3.4) и с использованием встроенной функции *mean*; сравнить полученные оценки с математическим ожиданием генерируемой случайной величины  $\xi$ , рассчитанным теоретически с использованием ее функции плотности распределения (смотри табл. 3.1);
  - построить доверительный интервал для математического ожидания в соответствии с выражением (3.5);
  - получить оценку дисперсии генерируемой случайной величины двумя способами: с использованием выражения (3.7) и с использованием встроенной функции *var*; сравнить полученные оценки с дисперсией генерируемой случайной величины  $\xi$ , рассчитанной

теоретически с использованием ее функции плотности распределения (смотри табл. 3.1);

- построить в одном графическом окне графики эмпирической плотности распределения случайной величины и (теоретической) плотности распределения (смотри табл. 3.1)

**Форма отчета.** М-файлы, реализующие задание.

Таблица 3.1

**Непрерывные случайные величины для программной генерации**

Вариант	Плотность распределения	Область задания
1	$1/(x \ln 2)$	$[1; 2]$
2	$1/(2x^{1/2})$	$[1; 4]$
3	$3(x - 1)^2$	$[1; 2]$
4	$\sin(2x)$	$[0; \pi/2]$
5	$0.2 \exp(-x/5)$	$[0; \infty]$
6	$x/2$	$[0; 2]$
7	$2\cos(2x)$	$[0; \pi/4]$
8	0.1	$[0; 10]$
9	$0.5 \exp(-x/2)$	$[0; \infty]$
10	$3\sin(x) \cos^2(x)$	$[0; \pi/2]$
11	$\cos(x)$	$[-\pi/2; 0]$
12	$3x^2/8$	$[0; 2]$
13	$5 \exp(-5x)$	$[0; \infty]$
14	$1/(x \ln 5)$	$[1; 5]$
15	$2(x + 1)$	$[-1; 0]$
16	0.5	$[0; 2]$

Окончание табл. 3.1

Вариант	Плотность распределения	Область задания
---------	-------------------------	-----------------

17	$1.5x^{1/2}$	$[0; 1]$
18	$\sin(x)$	$[0; \pi/2]$
19	$1/(1 + \cos(x))$	$[0; \pi/2]$
20	$2 \exp(-2x)$	$[0; \infty]$
21	$2x^{-3}$	$[1; \infty]$
22	$\cos(x)$	$[0; \pi/2]$
23	$\exp(-x)$	$[0; \infty]$
24	$3 \sin^2(x) \cos(x)$	$[0; \pi/2]$
25	$24x^{-4}$	$[2; \infty]$

## ЛАБОРАТОРНАЯ РАБОТА № 4

### МОДЕЛИРОВАНИЕ НЕПРЕРЫВНОЙ СЛУЧАЙНОЙ ВЕЛИЧИНЫ МЕТОДОМ НЕЙМАНА

**Цель работы.** Практическое освоение алгоритма программной генерации непрерывной случайной величины методом Неймана и методов статистической проверки разработанного генератора.

#### Краткие сведения

**Метод Неймана.** Предположим, что случайная величина  $\xi$  определена на интервале  $[a, b]$  и имеет на этом интервале плотность распределения  $f(x)$ . Пусть существует некоторая функция  $g(x)$ , такая, что для любого  $x$  выполняется неравенство  $f(x) \leq g(x)$ , и кроме того, площадь под кривой  $g(x)$  конечна, т.е.  $S = \int_a^b g(x)dx < \infty$ . Определим дополнительно две вспомогательные непрерывные случайные величины:

1.  $\eta$ , которая имеет плотность распределения  $r(x) = g(x)/S$  на интервале  $[a, b]$ . Для случайной величины  $\eta$  должен существовать удобный алгоритм ее моделирования, например, по методу обратных функций.
2.  $\psi$ , которая равномерно распределена на отрезке  $[0; g(x)]$  и имеет плотность распределения  $l(y) = 1/g(x)$ . Стоит отметить, что плотность распределения  $l(y)$  зависит от значения переменной  $x$  как от параметра, а сама случайная величина  $\psi$  независима по отношению к случайной величине  $\eta$ , на что указывает использование для нее отдельной переменной  $y$ .

Если область определения случайной величины  $\xi$  конечна, то в качестве  $\eta$  можно взять равномерно распределенную на интервале  $[a, b]$  случайную величину, что соответствует выбору  $g(x) = g = \text{const}$  (например:  $g(x) = g = \max f(x)$ ).

Реализации случайной величины  $\xi$  могут быть получены в соответствии со следующим алгоритмом [4]:

1. получаем реализацию  $X'$  случайной величины  $\eta$  (блок 1 на рис. 4.1);
2. получаем независимо реализацию  $Y$  случайной величины  $\psi$  (блок 2 на рис. 4.1);
3. если  $Y \leq f(X')$  (блок 3 на рис. 4.1), то реализация случайной величины  $\xi: X = X'$  (блок 4 на рис. 4.1), иначе  $X'$  отбрасывается и попытка сгенерировать  $X$  повторяется путем перехода к пункту 1.



Графическое представление процесса моделирования методом Неймана приведено на рисунке 4.1.

Блок-схема алгоритма для получения одной реализации непрерывной случайной величины методом Неймана представлена на рисунке 4.2.



Рис. 4.1. Графическое представление процесса моделирования методом Неймана

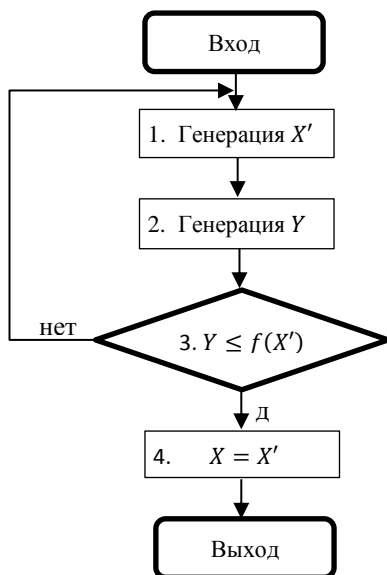


Рис. 4.2. Блок-схема моделирования отдельной реализации непрерывной случайной величины методом Неймана

### Пример:

Случайная величина  $\xi$  определена на отрезке  $[-R, R]$  и имеет плотность распределения:

$$f(x) = \frac{2}{\pi R^2} \sqrt{R^2 - x^2}.$$

Выберем функцию  $g(x)$ , принимающую на всей области  $[-R, R]$  одинаковые значения, равные максимальному значению функции  $f(x)$  на  $[-R, R]$ :

$$g(x) = \max_{-R \leq x \leq R} f(x) = 2/(\pi R).$$

Тогда плотность распределения вспомогательной случайной величины  $\eta$  представим в виде:

$$r(x) = \frac{2}{\pi RS} = \left\{ S = 2R \cdot \frac{2}{\pi R} \right\} = \frac{1}{2R}.$$

Данную случайную величину можно моделировать методом обратных функций:

$$\int_{-R}^x r(u) du = \int_{-R}^x \frac{1}{2R} du = \frac{x + R}{2R} = z_1 \Rightarrow X' = R(2z_1 - 1),$$

где  $z_1$  – реализация случайной величины, равномерно распределенной на отрезке  $[0, 1]$ . Для моделирования второй вспомогательной случайной величины  $\psi$  так же воспользуемся методом обратных функций:

$$\int_0^y l(v) dv = \int_0^y \frac{\pi R}{2} dv = \frac{\pi R y}{2} = z_2 \Rightarrow Y = \frac{2z_2}{\pi R},$$

где  $z_2$  – отдельная независимая реализация случайной величины, равномерно распределенной на отрезке  $[0, 1]$ . Использование отдельно сгенерированного значения  $z_2$  позволяет получить  $Y$  независимо от  $X'$ . Затем для того чтобы проверить, будет ли  $X'$  реализацией случайной величины  $\xi$ , проверяем выполнение неравенства

$$Y \leq f(X').$$

Таким образом, моделирование случайной величины  $\xi$  заключается в следующем. Разыгрываем две независимые реализации  $z_1$  и  $z_2$  базовой случайной величины. Если выполняется последнее неравенство, считаем, что  $X = X' = R(2z_1 - 1)$ . В противном случае разыгрываем следующую пару случайных чисел  $z_1$  и  $z_2$  и т.д.

**Критерий согласия  $\chi^2$ .** Критерий согласия  $\chi^2$  используется для проверки соответствия сгенерированной выборки теоретическому закону распределения некоторой случайной величины  $\xi$ , которая может быть как непрерывной, так и дискретной. Для выполнения вышеуказанной проверки область определения  $\Omega$  случайной величины  $\xi$  разбивается на  $r$  попарно непересекающихся подобластей  $M_1, \dots, M_r$ , таких, что сумма вероятностей  $p_j$  попадания значений случайной величины  $\xi$  в подобласть  $M_j$ , рассчитанных на базе ее закона распределения, равна 1:

$$\sum_{j=1}^r p_j = P\{\xi \in \Omega\} = 1. \quad (4.1)$$

Если  $\xi$  является непрерывной случайной величиной,  $M_j$  будут представлять собой подинтервалы, на которые разбита область ее определения. Отдельные подинтервалы могут иметь различную длину и совместно должны полностью заполнять область определения  $\xi$ . В случае, если  $\xi$  является дискретной случайной величиной, то в качестве  $M_j$  могут выступать либо ее отдельные значения, либо группы соседних значений. Если область определения  $\xi$  бесконечна, то крайние подобласти ( $M_1$  и/или  $M_r$ ) со стороны бесконечности будут включать бесконечное число значений.

Пусть тестируемая выборка содержит  $N$  реализаций. Тогда теоретически ожидаемое среднее число значений из данной выборки, которые должны попадать в подобласть  $M_j$ , вычисляется как  $Np_j$ . Обозначим через  $v_j$  количество значений из тестируемой выборки, действительно попавших в подобласть  $M_j$ . С использованием представленных выше обозначений можно построить величину

$$\chi_N^2 = \sum_{j=1}^r \frac{(v_j - Np_j)^2}{Np_j}, \quad (4.2)$$

отражающую степень отклонения распределения, которому в действительности подчиняются значения в тестируемой выборке, от закона распределения случайной величины  $\xi$ . Значения практических частот  $v_j$  попадания реализаций случайной величины в подобласть  $M_j$  являются элементами гистограммы, построенной на этом наборе подобластей по сгенерированной выборке. В рамках пакета MATLAB для построения гистограммы можно использовать функцию *hist*. Способ расчета теоретических вероятностей  $p_j$  зависит от вида случайной величины  $\xi$ . Если  $\xi$  является непрерывной случайной величиной, то  $p_j$  вычисляется как разность интегральной функции распределения случайной величины  $\xi$ , вычисленной соответственно в правой и левой границе интервала  $M_j$ , или как интеграл от функции плотности

распределения вероятности на интервале  $M_j$ . Если  $\xi$  является дискретной случайной величиной, то вероятности  $p_j$  вычисляются как сумма вероятностей отдельных ее дискретных значений, включенных в  $M_j$ .

Если область определения  $\xi$  бесконечна, то количество значений и вероятность попадания в крайнюю подобласть ( $M_1$  или  $M_r$ ) со стороны бесконечности находятся из условия нормировки случайной величины. Например, число попаданий  $M_r = N - \sum_{j=1}^{r-1} v_j$ , а вероятность  $p_r = 1 - \sum_{j=1}^{r-1} p_j$ .

Величина  $\chi_N^2$  при достаточно больших  $N$  должна подчиняться распределению  $\chi^2$  с  $r - 1$  степенью свободы. Поэтому для проверки соответствия сгенерированной выборки закону распределения случайной величины  $\xi$  необходимо зафиксировать доверительную вероятность  $\beta$  (на практике ее обычно выбирают равной 0.68, 0.95 или 0.997) и определить величину  $\chi^2(r - 1, 1 - \beta)$ , которая является корнем следующего интегрального уравнения:

$$\int_{\chi^2(r-1, 1-\beta)}^{\infty} \varphi_{r-1}(x) dx = 1 - \beta \quad (4.3)$$

где  $\varphi_{r-1}(x)$  – функция плотности распределения  $\chi^2$  с  $r - 1$  степенью свободы. Другими словами, нужно найти квантиль распределения  $\chi^2$  с вероятностью  $\beta$  и  $r - 1$  степенями свободы. Для того чтобы избежать необходимости каждый раз напрямую решать уравнение (4.3), на практике для нахождения величины  $\chi^2(r - 1, 1 - \beta)$  часто используют специально подготовленные таблицы (смотри табл. 4.2) или решают (4.3) численно. Например, в MATLAB для вычисления значения  $\chi^2(r - 1, 1 - \beta)$ , при любом  $r$  и  $\beta$  можно использовать функцию  $chi2inv(\beta, r-1)$ .

Если вычисленное по формуле (4.2) значение  $\chi_N^2$  удовлетворяет неравенству  $\chi_N^2 < \chi^2(r - 1, 1 - \beta)$ , то с вероятностью  $\beta$  гипотеза о совпадении законов распределения заданной случайной величины и случайной величины, моделируемой программным генератором, принимается. Этот вывод, конечно, нельзя считать абсолютным. Если получаемая последовательность удовлетворяет критерию  $\chi^2$ , это еще не означает, что она успешно пройдет проверку другими тестами.

В качестве замечания о применении критерия  $\chi^2$  необходимо отметить, что если  $N$  заранее ограничено, то нельзя выбирать  $r$  слишком большим, так как тогда будут малы величины  $Np_j$  и неустойчивыми значениями  $v_j$ , что может приводить к неоправданно большим значениям отдельных слагаемых в формуле (4.2). Обычно рекомендуют выбирать подобласти  $M_j$

так, чтобы минимальное  $Np_j$  было не меньше 5. В противном случае рекомендуется прибегать к объединению соседних областей  $M_j$ .

**Критерий согласия  $\omega^2$ .** Данный критерий позволяет проверить гипотезы о соответствии сгенерированной выборки некоторой случайной величине с известной функцией распределения  $F(x)$ .

Для выполнения проверки по критерию  $\omega^2$  значения  $x_i, i = 1, \dots, N$  из сгенерированной выборки сортируются в порядке возрастания. Отсортированную последовательность будем обозначать  $Q_i, i = 1, \dots, N$ . С использованием этой последовательности рассчитывается величина  $\omega_N^2$ :

$$\omega_N^2 = \frac{1}{12N} + \sum_{i=1}^N \left[ F(Q_i) - \frac{i-0.5}{N} \right]^2. \quad (4.4)$$

Отметим, что в формуле (4.4) используется функция распределения, а не плотность. Далее фиксируется доверительная вероятность  $\beta$ , в соответствии с которой определяется критическое значение критерия  $x_\beta$  (для  $\beta = 0.95$  значение  $x_\beta = 0.46$ ).

Если  $\omega_N^2 < x_\beta$  то гипотеза о соответствии функции  $F(x)$  и функции распределения генерируемой случайной величины принимается. Применять критерий  $\omega^2$  можно при объеме тестируемой выборки  $N > 100$ .

### Порядок выполнения

1. С использованием справочной системы MATLAB изучить функцию *chi2inv*.
2. Разработать алгоритм моделирования заданной (см. ниже табл. 4.1) непрерывной случайной величины с использованием метода Неймана и программно реализовать его в среде MATLAB.
3. Провести тестовую проверку созданного генератора:
  - построить в одном графическом окне графики эмпирической плотности распределения случайной величины и плотности, заданной в табл. 4.1 в соответствии с полученным вариантом;
  - проверить гипотезу о соответствии генерируемой случайной величины заданному распределению по критерию  $\chi^2$ ;
  - проверить гипотезу о соответствии генерируемой случайной величины заданному распределению по критерию  $\omega^2$ .

**Форма отчета.** М-файлы, реализующие задание.

Таблица 4.1

**Непрерывные случайные величины для программной генерации**

Вариант	Плотность распределения	Область задания
1	$\cos(x)$	$[0; \pi/2]$
2	$\exp(-x)/(1 - \exp(-1))$	$[0; 1]$
3	$3\sin(x) \cos^2(x)$	$[0; \pi/2]$
4	$2.5x^{3/2}$	$[0; 1]$
5	$1/(x \ln 2)$	$[1; 2]$
6	$2\exp(-2x)/(1 - \exp(-1))$	$[0; 0.5]$
7	$4x^{1/3}/3$	$[0; 1]$
8	$1/(1 + \cos(x))$	$[0; \pi/2]$
9	$2(x + 1)$	$[-1; 0]$
10	$x/2$	$[0; 2]$
11	$2x^{-3}$	$[0.6; 0.75]$
12	$5\exp(-5x)/(1 - \exp(-1))$	$[0; 0.2]$
13	$3x^2/8$	$[0; 2]$
14	$\cos(x)$	$[-\pi/2; 0]$
15	$0.5\exp(-x/2)/(1 - \exp(-1))$	$[0; 2]$
16	$3 \sin^2(x) \cos(x)$	$[0; \pi/2]$
17	$0.2\exp(-x/5)/(1 - \exp(-1))$	$[0; 5]$
18	$x^{-2}$	$[0.5; 1]$
19	$3(x - 1)^2$	$[1; 2]$
20	$1/(x \ln 5)$	$[1; 5]$
21	$\sin(x)$	$[0; \pi/2]$
22	$1.5x^{1/2}$	$[0; 1]$
23	$1/(2x^{1/2})$	$[1; 4]$
24	$x/8$	$[3; 5]$
25	$2\cos(2x)$	$[0; \pi/4]$

Таблица 4.2

Процентные точки распределения  $\chi^2$ 

$\begin{matrix} 1-\beta \\ r-1 \end{matrix}$	0,990	0,975	0,950	0,900	0,10	0,05	0,025	0,010
1	0,00016	0,00098	0,0039	0,158	2,71	3,84	5,02	6,63
2	0,0201	0,0506	0,103	0,211	4,61	5,99	7,38	9,21
3	0,115	0,216	0,352	0,584	6,25	7,81	9,35	11,34
4	0,297	0,484	0,711	1,06	7,78	9,49	11,14	13,28
5	0,554	0,831	1,15	1,61	9,24	11,07	12,83	15,09
6	0,872	1,24	1,64	2,20	10,64	12,59	14,45	16,81
7	1,24	1,69	2,17	2,83	12,02	14,07	16,01	18,48
8	1,65	2,18	2,73	3,49	13,36	15,51	17,53	20,09
9	2,09	2,70	3,33	4,17	14,68	16,92	19,02	21,67
10	2,56	3,25	3,94	4,87	15,99	18,31	20,48	23,21
11	3,05	3,82	4,57	5,58	17,28	19,68	21,92	24,73
12	4,57	4,40	5,23	6,30	18,55	21,03	23,34	26,22
13	4,11	5,01	5,89	7,04	19,81	22,36	24,74	27,69
14	4,66	5,63	6,57	7,79	21,06	23,68	26,12	29,14
15	5,23	6,26	7,26	8,55	22,31	25,00	27,49	30,58
16	5,81	6,91	7,96	9,31	23,54	26,30	28,85	32,00
17	6,41	7,56	8,67	10,08	24,77	27,59	30,19	33,41
18	7,01	8,23	9,39	10,86	25,99	28,87	31,53	34,81
19	7,63	8,91	10,12	11,65	27,20	30,14	32,85	36,19
20	8,26	9,59	10,85	12,44	28,41	31,41	34,17	37,57
21	8,90	10,28	11,59	13,24	29,62	32,67	35,48	38,93
22	9,54	10,98	12,34	14,04	30,81	33,92	36,78	40,29
23	10,20	11,69	13,09	14,85	32,01	35,17	38,08	41,64
24	10,86	12,40	13,85	15,66	33,20	36,42	39,36	42,98
25	11,52	13,12	14,61	16,47	34,38	37,65	40,65	44,31
26	12,20	13,84	15,38	17,29	35,56	38,88	41,92	45,64
27	12,88	14,57	16,15	18,11	36,74	40,11	43,19	46,96
28	13,56	15,31	16,93	18,94	37,92	41,34	44,46	48,28
29	14,26	16,05	17,71	19,77	39,09	42,56	45,72	49,59
30	14,95	16,79	18,49	20,60	40,26	43,77	46,98	50,89
40	22,16	24,43	26,51	29,05	51,81	55,76	59,34	63,69
60	37,48	40,48	43,19	46,46	74,40	79,08	83,30	88,38
120	86,92	91,58	95,70	100,62	140,23	146,57	152,21	158,95

## ЛАБОРАТОРНАЯ РАБОТА № 5

### МОДЕЛИРОВАНИЕ ДИСКРЕТНОЙ СЛУЧАЙНОЙ ВЕЛИЧИНЫ

**Цель работы.** Практическое освоение алгоритма программной генерации дискретной случайной величины и методов статистической проверки разработанного генератора.

#### Краткие сведения

**Моделирование дискретных случайных величин.** Пусть задана дискретная случайная величина  $\xi$ , принимающая конечное или счетное множество значений  $x_i, i = 1, 2, \dots, n$ , с вероятностями  $p_i = P\{\xi = x_i\}, \sum_{i=1}^n p_i = 1$  (случай  $n = \infty$  не исключается). Алгоритм моделирования дискретной случайной величины основан на использовании равномерно распределенной на отрезке  $[0; 1]$  непрерывной (так называемой базовой) случайной величины  $\alpha$  и факте, что вероятности выпадения значений  $x_i$  дискретной случайной величины также принимают значения из этого интервала (сумма вероятностей равна 1) [3].

Разделим отрезок  $[0; 1]$  на интервалы  $a_i$  длины  $p_i$ . Количество интервалов  $a_i$  равно количеству  $n$  возможных значений моделируемой дискретной случайной величины  $\xi$  (случай  $n = \infty$  не исключается). Вероятность попадания реализации  $z$  базовой случайной величины  $\alpha$  в некоторый интервал равна длине этого интервала и, следовательно, совпадает с вероятностью  $p_i$  того, что случайная величина  $\xi$  примет значение  $x_i$ . Поэтому для получения реализации  $x$  случайной величины  $\xi$  необходимо разыграть значение  $z$  случайной величины  $\alpha$  и определить интервал, в который оно попало. Если значение  $z$  попадает в интервал  $a_i$ , величина  $x$  полагается равной  $x_i$ .

На рисунке 5.1 приведена блок-схема алгоритма моделирования случайной величины  $\xi$ . Блок 1 генерирует значение  $z$  случайной величины  $\alpha$ . В блоке 2 значение  $z$  присваивается переменной  $S$ , а целочисленная переменная  $i$  получает значение 1. В блоках 3, 4 величина  $S$  модифицируется путем вычитания из нее значения  $p_i$ . Такое вычитание производится с последовательным увеличением значения  $i$  на единицу в блоке 5 до тех пор, пока при некотором  $i \leq n$  в блоке 4 не получится  $S < 0$ . Это и означает, что значение  $z$  попадает в интервал  $a_i$ , а следовательно,  $x$  принимает значение  $x_i$  (блок 6), т.е. реализацией дискретной случайной величины  $\xi$  будет значение  $x$  с индексом  $i$  (если по заданию  $x_i = i$ , то самому значению  $i$ ).



Отметим, что часто для вычисления  $p_i$  задаются формулы (например, при  $n = \infty$ ). В этом случае для повышения скорости моделирования несколько первых значений  $p_1, p_2, \dots, p_k$  можно подсчитать заранее. Тогда в программе расчет по формуле для  $p_i$  потребует лишь для  $i > k$ . Если величина  $\sum_{i=1}^k p_i$  близка к единице, то это будет случаться весьма редко, поскольку вероятность события  $i > k$  равна  $1 - \sum_{i=1}^k p_i$ .

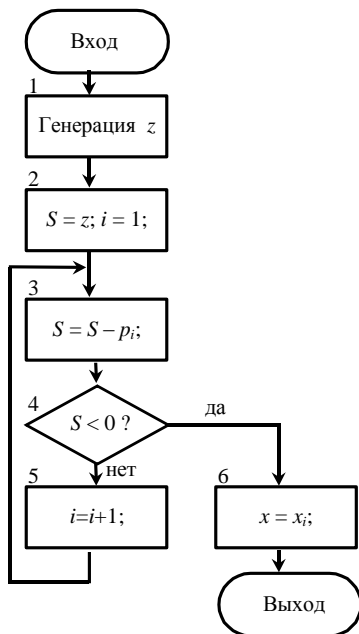


Рис. 5.1. Блок-схема алгоритма моделирования дискретной случайной величины

### Пример:

Рассмотрим алгоритм моделирования дискретной случайной величины, принимающей значения 1 и 0 с вероятностями  $p$  и  $1 - p$ , соответственно (распределение Бернулли). Разделим отрезок  $[0; 1]$  на два интервала длиной  $p$  и  $1 - p$ , смотри рис. 5.2. Получим реализацию  $z$  базовой случайной величины  $\alpha$ , воспользовавшись, к примеру, функцией *rand* среды MATLAB. Если реализация  $z < p$  (попадает внутрь первого отрезка), то случайная величина примет значение 1, в противном случае – значение 0.

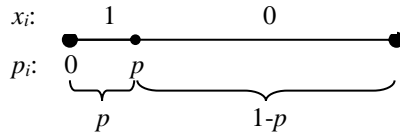


Рис. 5.2. Схема генерации распределения Бернулли

**Расчет математического ожидания и дисперсии дискретной случайной величины.** Математическое ожидание дискретной случайной величины рассчитывается по формуле  $\mu = \sum_{i=1}^n x_i p_i$ . Если  $n = \infty$ , то можно воспользоваться следующим приближенным способом расчета: задают погрешность вычисления  $\varepsilon$  (например,  $10^{-6}$ ) и накапливают в цикле сумму до тех пор, пока следующее слагаемое не станет меньше заданной погрешности.

Дисперсия дискретной случайной величины рассчитывается по формуле  $\sigma^2 = \sum_{i=1}^n (x_i - \mu)^2 p_i = \sum_{i=1}^n x_i^2 p_i - \mu^2$ . При  $n = \infty$  можно воспользоваться изложенным выше приближенным способом расчета. В приближенном способе расчета использование формулы  $\sigma^2 = \sum_{i=1}^n x_i^2 p_i - \mu^2$  является предпочтительным, так как при целом значении  $\mu$  возможен преждевременный выход из цикла при  $x_i = \mu$ .

**Построение распределения вероятностей.** Оценка распределения вероятностей дискретной случайной величины рассчитывается по следующей формуле:

$$\hat{p}_i = \frac{v_i}{N}, \quad (5.1)$$

где  $v_i$  частота выпадения в сгенерированной выборке  $i$ -го значения дискретной случайной величины  $x_i$ ,  $N$  - общее количество сгенерированных значений. Частоты выпадения значений можно получить, воспользовавшись функцией *hist* среды MATLAB, имеющей два аргумента: вектор реализаций случайной величины и вектор центров карманов. Для дискретной случайной величины в качестве второго аргумента следует передать вектор самих значений  $x_i$  случайной величины (для бесконечных распределений последним значением в данном векторе будет максимальное значение из сгенерированной выборки).

### Порядок выполнения

1. Разработать алгоритм моделирования заданной (см. табл. 5.1) дискретной случайной величины и программно его реализовать в среде MATLAB.

2. Провести тестовую проверку созданного генератора:
- вычислить математическое ожидание заданной случайной величины. Получить оценку математического ожидания по сгенерированной выборке и сравнить ее с теоретическим значением;
  - вычислить дисперсию заданной случайной величины. Получить оценку дисперсии по сгенерированной выборке и сравнить ее с теоретическим значением;
  - построить графики заданного распределения вероятностей случайной величины и эмпирического распределения, полученного по сгенерированной выборке;
  - проверить гипотезу о соответствии сгенерированной выборки заданному распределению случайной величины по критерию  $\chi^2$ .

**Форма отчета.** М-файлы, реализующие задание.

Таблица 5.1

**Дискретные случайные величины для программной генерации**

Вариант	Множество значений	Распределение вероятностей
1	$k = 0, 1, \dots$	$P\{\xi = k\} = e^{-1}/k!$
2	$k = 0, 1, \dots$	$P\{\xi = k\} = 2^{-k-1}$
3	$k = \{3; 5; 7; 9\}$	$P\{\xi = k\} = \{0.35; 0.15; 0.35; 0.15\}$
4	$k = 1, 2, \dots$	$P\{\xi = k\} = (1 - 1/e)^k/k$
5	$k = 0, 1, \dots$	$P\{\xi = k\} = (\ln 2)^k/(2k!)$
6	$k = \{1; 2; 3; 4; 5\}$	$P\{\xi = k\} = \{0.05; 0.45; 0.25; 0.15; 0.1\}$
7	$k = 0, 1, \dots$	$P\{\xi = k\} = p^k(1 - p); p = 0.7$
8	$k = 1, 2, \dots$	$P\{\xi = k\} = -(1 - p)^k/(k \ln p); p = 0.25$
9	$k = 0, 1, \dots$	$P\{\xi = k\} = C_{k+m-1}^{m-1} 0.6^m 0.4^k; m = 12$
10	$k = 0, 1, \dots$	$P\{\xi = k\} = p(1 - p)^k; p = 0.25$
11	$k = 0, 1$	$P\{\xi = k\} = \{0.3; 0.7\}$
12	$k = 0, 1, \dots$	$P\{\xi = k\} = 10^k e^{-10}/k!$
13	$k = 0, 1, \dots$	$P\{\xi = k\} = \frac{(k+1)(k+2)(k+3)0.7^4 0.3^k}{6}$

Вариант	Множество значений	Распределение вероятностей
14	$k = 0, 1, \dots$	$P\{\xi = k\} = 5^k e^{-5} / k!$
15	$k = 0, 1, \dots$	$P\{\xi = k\} = 3^k / 4^{k+1}$
16	$k = \{1; 2; 3; 100\}$	$P\{\xi = k\} = \{0.33; 0.33; 0.33; 0.01\}$
17	$k = 0, 1, \dots, 10$	$P\{\xi = k\} = C_{10}^k 0.3^k 0.7^{10-k}$
18	$k = 0, 1, \dots$	$P\{\xi = k\} = (k + 1) 0.3^2 0.7^k$
19	$k = 0, 1, \dots$	$P\{\xi = k\} = 3 / 4^{k+1}$
20	$k = 1, 2, \dots$	$P\{\xi = k\} = p(1 - p)^{k-1}; p = 0.5$
21	$k = 0, 1, \dots$	$P\{\xi = k\} = 2^k / 3^{k+1}$
22	$k = 1, 2, \dots, 10$	$P\{\xi = k\} = 0.1$
23	$k = 1, 2, \dots$	$P\{\xi = k\} = 1 / (2^k k \ln 2)$
24	$k = 0, 1, \dots$	$P\{\xi = k\} = \frac{(k+1)(k+2)p^3(1-p)^k}{2}; p = 0.5$
25	$k = 0, 1, \dots$	$P\{\xi = k\} = 2 / 3^{k+1}$

## ЛАБОРАТОРНАЯ РАБОТА № 6

### МОДЕЛИРОВАНИЕ ПОТОКА ПУАССОНА

**Цель работы.** Практическое освоение алгоритма программной генерации стационарного потока Пуассона и методов статистической проверки разработанного генератора.

#### Краткие сведения

**Случайные потоки.** При моделировании стохастических систем типичны ситуации, когда случайные воздействия, возникающие в процессе функционирования сложной системы, представляют собой последовательность событий, наступающих в отдельные случайные моменты времени из интервала моделирования  $\Omega = [T_0; T]$ . Каждое событие определяется указанием лишь одной координаты – момента времени появления. Такую последовательность называют случайным потоком [3, 4].

Обозначим через  $\xi_i$  момент наступления  $i$ -го события потока на  $\Omega$ ,  $i = 1, 2, \dots$  Будем предполагать, что моменты наступления событий упорядочены по возрастанию, т. е.  $T_0 = \xi_1 \leq \xi_2 \leq \dots \leq \xi_i \leq \dots \leq T$ . Геометрически случайный поток можно изобразить в виде следующих друг за другом точек на временной оси (рис. 6.1).

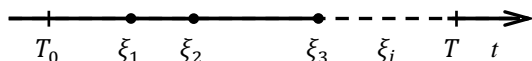


Рис. 6.1. Поток случайных событий

Поток называется *ординарным*, если вероятность попадания на элементарный отрезок  $\Delta t$  двух или более событий пренебрежимо мала по сравнению с вероятностью попадания на него ровно одного события.

Поток событий называется потоком *без последствия*, если для любых двух непересекающихся промежутков времени число событий, попадающих в один из них не зависит от того, сколько событий попало в другой.

Поток событий называется *стационарным*, если вероятность выпадения того или иного числа событий на интервале  $\Omega$  зависит только от длины этого интервала и не зависит от того, где  $\Omega$  расположен на оси времени.

Можно провести аналогию между случайным вектором и случайным потоком. И в том и в другом случае речь идет о наборе компонент, являющихся реализациями в общем случае зависимых случайных величин. Но если у вектора число компонент фиксировано и не меняется от реализации

к реализации, то поток характеризуется случайным числом компонент, которое может обращаться в бесконечность, если  $\Omega$  имеет бесконечную протяженность. Другими словами, о случайном потоке можно говорить как о случайном векторе со случайной размерностью. Задача моделирования случайного потока заключается в последовательной генерации упорядоченных по возрастанию моментов наступления событий.

**Моделирование потока Пуассона.** Пуассоновские потоки случайных событий широко используются для описания случайных воздействий в системах обеспечения экспериментов в оптике, ядерной физике, радиофизике.

*Потоком Пуассона* называют поток случайных событий, характеризующийся свойствами ординарности и отсутствия последействия. Потоком Пуассона данный поток назван потому, что вероятность появления на интервале наблюдения  $\Omega$   $k$  событий подчиняется дискретному распределению Пуассона

$$P\{n = k\} = \frac{\Delta^k}{k!} \exp(-\Delta), k = 0, 1, 2, \dots, \quad (6.1)$$

где  $n$  – случайная величина, характеризующая количество событий на интервале  $\Omega$ ,  $\Delta = \int_{\Omega} \lambda(t)dt$ ,  $\lambda(t)$  – интенсивность потока Пуассона.

Поток Пуассона полностью определяется своей интенсивностью  $\lambda(t)$ , которую следует рассматривать как мгновенную интенсивность потока в некоторый момент времени  $t$ . С учетом этого можно определить смысл параметра  $\Delta$  в выражении (6.1). Из формулы  $\Delta = \int_{\Omega} \lambda(t)dt$  следует, что  $\Delta$  – это среднее число событий потока Пуассона на интервале наблюдения  $\Omega$ .

Алгоритм моделирования стационарного потока Пуассона ( $\lambda(t) = \lambda = \text{const}$ ) непосредственно следует из его основных свойств: ординарности и отсутствия последействия. Можно показать, что распределение интервала между событиями  $\tau_i = t_i - t_{i-1}$  у стационарного потока Пуассона подчиняется экспоненциальному распределению с функцией плотности вероятности  $f(\tau) = \lambda e^{-\lambda\tau}$ . Следовательно, моделирование стационарного потока Пуассона осуществляется по следующему рекуррентному алгоритму [3, 4]:

$$t_i = t_{i-1} - \frac{\ln(z_i)}{\lambda}, \quad (6.2)$$

где  $z_i$  – реализация случайной величины равномерно распределенной на  $[0; 1]$ ;  $t_0 = T_0$ . Процесс моделирования продолжается до тех пор, пока очередное значение  $t_i$  не превысит правую границу области  $\Omega$ . После окончания моделирования сгенерированная реализация стационарного потока

Пуассона будет представлять собой совокупность событий, полученных в пределах интервала моделирования  $\Omega$ .

### **Построение статистических характеристик случайного потока.**

Для тестовой проверки генератора случайного потока необходимо выполнить сравнение оценок статистических характеристик потока, построенных по набору его смоделированных реализаций, с соответствующими им теоретическими зависимостями. Для качественного построения оценок статистических характеристик случайного потока необходимо использовать достаточно большое количество его реализаций, полученных при помощи тестируемого генератора. В данной лабораторной работе будем рассматривать две статистических характеристики потока: *интенсивность* и *распределение числа событий случайного потока на интервале наблюдения*.

Для построения *оценки интенсивности случайного потока* интервал моделирования потока  $\Omega$  разобьем на  $m$  неперекрывающихся подинтервалов. В качестве оценки интенсивности потока на  $i$ -м подинтервале можно использовать среднее число событий потока в единицу времени, рассчитываемое в соответствии с выражением:

$$\lambda_i^* = \frac{n_i}{N \cdot dt}, i = 1, 2, \dots, m, \quad (6.3)$$

где  $n_i$  – количество событий, попавших в  $i$ -й подинтервал при  $N$ -кратном моделировании потока,  $dt$  – ширина подинтервала. Теоретическая интенсивность потока Пуассона задается функцией  $\lambda(t)$ . Для нахождения  $n_i$  можно применить функцию MATLAB hist().

*Распределение числа событий случайного потока* на интервале наблюдения представляет собой зависимость вероятности появления определенного числа событий рассматриваемого потока на области  $\Omega$  от возможного числа событий. Оценка данной статистической характеристики случайного потока строится в соответствии с выражением

$$P_k^* = \frac{N_k}{N}, k = 0, 1, 2, \dots, \quad (6.4)$$

где  $N_k$  – количество смоделированных реализаций случайного потока, в которых число событий оказалось равным  $k$ . Теоретическое распределение числа событий на интервале наблюдения для потока Пуассона задается распределением Пуассона (6.1). Количество реализаций случайного потока  $N_k$ , в которых число событий оказалось равным  $k$  есть гистограмма числа событий, полученных при  $N$ -кратном моделировании потока. Для ее нахождения можно применить функцию MATLAB hist( $n, k$ ), где  $n$  – вектор, содержащий число событий, полученных в каждой отдельной реализации потока,

а  $k$  – вектор, содержащий возможное число событий в потоке, генерируемый как  $k = 0: k_{max}$ , где  $k_{max}$  – максимальное число выпавших событий.

### Порядок выполнения

1. Разработать алгоритм моделирования заданного (см. ниже табл. 6.1) потока Пуассона и программно реализовать его в среде MATLAB.
2. Провести тестовую проверку созданного программного генератора:
  - построить на интервале  $\Omega$  графики заданной интенсивности потока Пуассона и оценки интенсивности генерируемого потока;
  - построить графики распределения числа событий потока Пуассона (6.1) на интервале  $\Omega$  и оценки распределения числа событий на  $\Omega$  генерируемого потока

**Форма отчета.** М-файлы, реализующие задание.

Таблица 6.1

**Параметры генерируемых потоков**

Вариант	Интенсивность	Интервал моделирования	Вариант	Интенсивность	Интервал моделирования
1	$\lambda=0.5$	$\Omega=[0; 1]$	14	$\lambda=2$	$\Omega=[0; 4]$
2	$\lambda=0.5$	$\Omega=[0; 2]$	15	$\lambda=2$	$\Omega=[0; 5]$
3	$\lambda=0.5$	$\Omega=[0; 3]$	16	$\lambda=5$	$\Omega=[0; 0.5]$
4	$\lambda=0.5$	$\Omega=[0; 4]$	17	$\lambda=5$	$\Omega=[0; 1]$
5	$\lambda=0.5$	$\Omega=[0; 5]$	18	$\lambda=5$	$\Omega=[0; 1.5]$
6	$\lambda=1$	$\Omega=[0; 1]$	19	$\lambda=5$	$\Omega=[0; 2]$
7	$\lambda=1$	$\Omega=[0; 2]$	20	$\lambda=5$	$\Omega=[0; 2.5]$
8	$\lambda=1$	$\Omega=[0; 3]$	21	$\lambda=10$	$\Omega=[0; 0.2]$
9	$\lambda=1$	$\Omega=[0; 4]$	22	$\lambda=10$	$\Omega=[0; 0.4]$
10	$\lambda=1$	$\Omega=[0; 5]$	23	$\lambda=10$	$\Omega=[0; 0.6]$
11	$\lambda=2$	$\Omega=[0; 1]$	24	$\lambda=10$	$\Omega=[0; 0.8]$
12	$\lambda=2$	$\Omega=[0; 2]$	25	$\lambda=10$	$\Omega=[0; 1]$
13	$\lambda=2$	$\Omega=[0; 3]$			



## БИБЛИОГРАФИЧЕСКИЕ ССЫЛКИ

1. *Гилат А.* MATLAB. Теория и практика. М.: ДМК Пресс, 2016. 416 с.
2. *Гилевский С.В., Молофеев В.М.* Теория вероятностей и математическая статистика. Минск: БГУ, 2015. 175 с.
3. *Апанасович В.В., Тихоненко О.М.* Цифровое моделирование стохастических систем. Минск: Университетское, 1986. 127 с.
4. *Дигрис А.В.* Дискретно-событийное моделирование. Минск: БГУ, 2011. 201 с.

## СОДЕРЖАНИЕ

ВВЕДЕНИЕ .....	3
Лабораторная работа № 1 Изучение среды matlab и simulink .....	4
Лабораторная работа № 2 Изучение встроенных средств Matlab для моделирования и оценки случайных величин .....	20
Лабораторная работа № 3 Моделирование непрерывной случайной величины методом обратных функций .....	35
Лабораторная работа № 4 Моделирование непрерывной случайной величины методом НЕЙМАНА .....	40
Лабораторная работа № 5 Моделирование дискретной случайной величины .....	48
Лабораторная работа № 6 Моделирование потока Пуассона .....	53
Библиографические ссылки .....	57

Учебное издание

## **МАТЕМАТИЧЕСКОЕ МОДЕЛИРОВАНИЕ**

**Методические указания  
к лабораторным работам**

**Для студентов специальностей  
1-31 04 02 «Радиофизика»,  
1-31 04 03 «Физическая электроника»,  
1-31 04 04 «Аэрокосмические радиоэлектронные  
и информационные системы и технологии»**

В авторской редакции

Ответственный за выпуск *А. В. Дигрис*

Подписано в печать 02.02.2017. Формат 60×84/16. Бумага офсетная.

Печать офсетная. Усл. печ. л. 3,49. Уч.-изд. л. 2,57.

Тираж 20 экз. Заказ

Белорусский государственный университет.

Свидетельство о государственной регистрации издателя, изготовителя,  
распространителя печатных изданий № 1/270 от 03.04.2014.

Пр. Независимости, 4, 220030, Минск.

Отпечатано с оригинал-макета заказчика  
на копировально-множительной технике  
факультета радиофизики и компьютерных технологий  
Белорусского государственного университета.  
Ул. Курчатова, 5, 220064, Минск.