МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ
ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ
ВЫСШЕГО ОБРАЗОВАНИЯ
«ЕЛЕЦКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ ИМ. И.А. БУНИНА»

И. В. Пешков

МЕТОДИЧЕСКИЕ УКАЗАНИЯ К ВЫПОЛНЕНИЮ ЛАБОРАТОРНЫХ РАБОТ ПО КУРСУ «ЦИФРОВАЯ ОБРАБОТКА СИГНАЛОВ»

Елец – 2016

УДК 621.396.2
ББК 32.884.1
П 23

Печатается по решению редакционно-издательского совета
Елецкого государственного университета имени И.А. Бунина
от 29. 01. 2016 г., протокол № 1

Рецензенты:
С.В. Волобуев, к.п.н., доц. каф. «Физики и методики ее преподавания»
Елецкого государственного университета

Ю.Б. Нечаев, д.ф.-м.н., проф. каф. «Информационных систем»
Воронежского государственного университета

И.В. Пешков

П 23

Методические указания к выполнению лабораторных работ по курсу «Цифровая обработка сигналов». – Елец: Елецкий государственный университет им. И.А. Бунина, 2016. – 35 с.

В методических указаниях представлены требования к выполнению лабораторных работ и оформлению отчетов по ним, техника безопасности при выполнении экспериментов. По каждой теме исследования даются рекомендации по подготовке к лабораторному занятию, основные теоретические положения и порядок выполнения экспериментальных исследований.

Методические указания предназначены для студентов направлений подготовки бакалавров: Радиотехника.

Предназначено для студентов вышеуказанных направлений подготовки и специальности очной и заочной форм обучения.

УДК 621.396.2
ББК 32.884.1

Содержание

1. Octave как научный калькулятор

После вызова Octave из среды Windows на экране возникает изображение так называемого "командного окна" среды Octave (рис. 1).

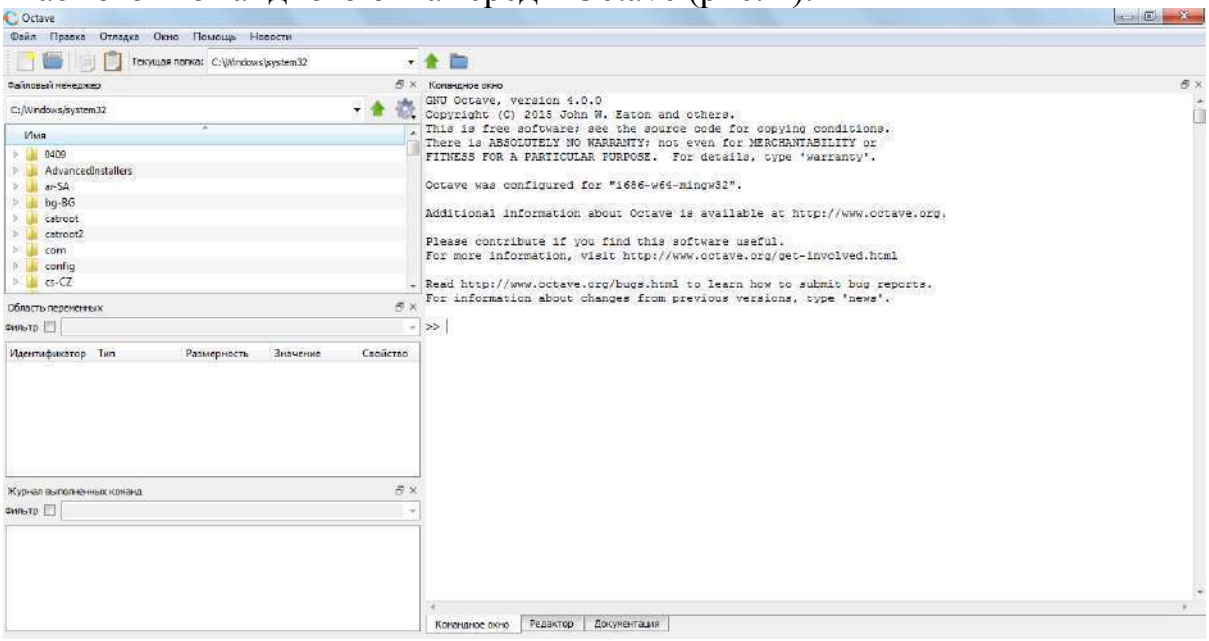


Рис. 1.

Это окно является основным в *Octave*. В нем появляются символы команд, которые набираются пользователем на клавиатуре дисплея, отображаются результаты выполнения этих команд, текст исполняемой программы и информация об ошибках выполнения программы, распознанных системой.

Признаком того, что Octave готова к восприятию и выполнению очередной команды, является возникновение в последней строке текстового поля окна знака приглашения '»', после которого расположена мигающая вертикальная черта.

Ввод чисел с клавиатуры осуществляется по общим правилам, принятым для языков программирования высокого уровня:

для отделения дробной части числа используется десятичная точка;

числа после символа «e» означают степень 10.

Если, например, ввести в командном окне Octave строку

1.20357651e-17

то после нажатия клавиши <Enter> в этом окне появится запись:

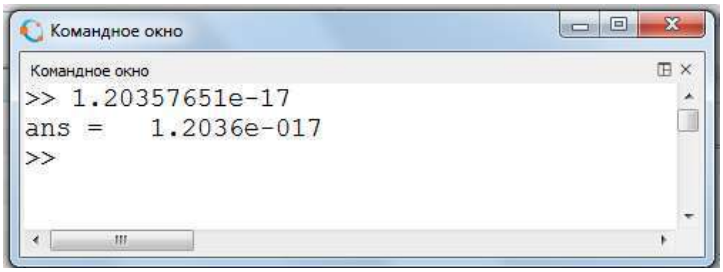


Рис. 2.

В арифметических выражениях языка Octave используются следующие знаки арифметических операций:

+ - сложение;
- - вычитание;
* - умножение;
/ - деление слева направо;
\ - деление справа налево;
^ - возведение в степень.

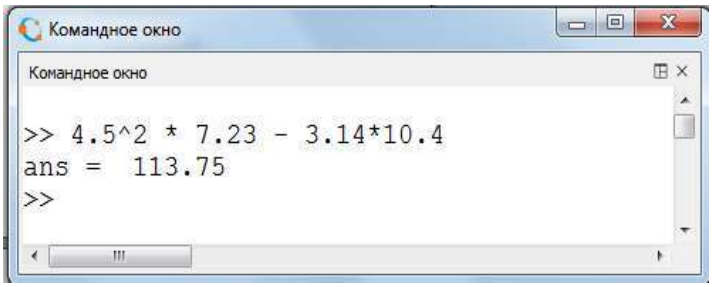


Рис. 3.

Использование Octave в режиме калькулятора может происходить путем простой записи в командную строку последовательности арифметических действий с числами, то есть обычного арифметического выражения, например:

4.5^2*7.23 - 3.14*10.4

Если после ввода с клавиатуры этой последовательности нажать клавишу <Enter>, в командном окне возникнет результат выполнения в виде, представленном на рис. 1.6, т. е. на экран под именем системной переменной ***ans*** выводится результат действия последнего выполненного оператора.

Вообще вывод промежуточной информации в командное окно подчиняется таким правилам:

- если запись оператора не заканчивается символом ';', результат действия этого оператора сразу же выводится в командное окно;
- если оператор заканчивается символом ';', результат этого действия не отображается в командном окне;
- если оператор не содержит знака присваивания (=), т. е. является просто записью некоторой последовательности действий над числами и переменными, значение результата присваивается специальной системной переменной по имени ans;
- полученное значение переменной ans можно использовать в следующих операторах вычислений, применяя это имя ans; при этом следует помнить, что значение системной переменной ans изменяется после действия очередного оператора без знака присваивания;

Пусть нужно вычислить выражение (25+17)*7. Это можно сделать таким образом. Сначала набираем последовательность 25+17 и нажимаем <Enter>. Получаем на экране результат в виде ans = 42. Теперь записываем последовательность ans*7 и нажимаем <Enter>. Получаем ans = 294 (рис. 1.7). Чтобы предотвратить выведение промежуточного результата действия 25+17, достаточно после записи этой последовательности добавить символ ' ; '. Тогда будем иметь результаты в виде, представленном на рис. 1.4

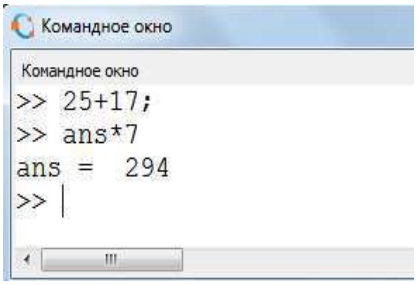


Рис. 4.

Например, если ввести в командное окно строку 'x = 25 + 17', на экране появится запись (рис. 5):

Рис. 5.

Система Octave имеет несколько имен переменных, которые используются самой системой и входят в состав зарезервированных: i, j - мнимая единица (корень квадратный из -1); pi – число; inf - обозначение машинной бесконечности; Na- обозначение неопределенного результата; eps - погрешность операций над числами с плавающей запятой.

Ввод с клавиатуры значения комплексного числа осуществляется путем записи в командное окно строки вида:

<имя комплекс. переем.> = <действит. часть> + <i или j> *<мнимая часть>

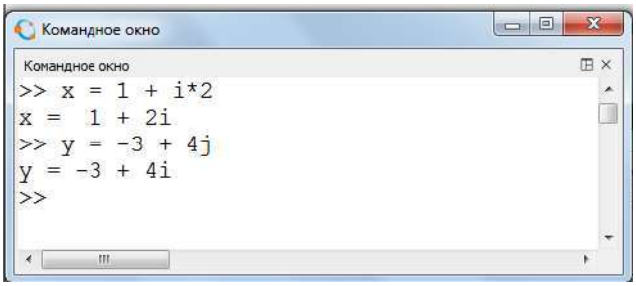


Рис. 6.

Простейшие действия с комплексными числами - сложение, вычитание, умножение, деление и возведение в степень - осуществляются при помощи обычных арифметических знаков +, -, *, /, \ и ^ соответственно.

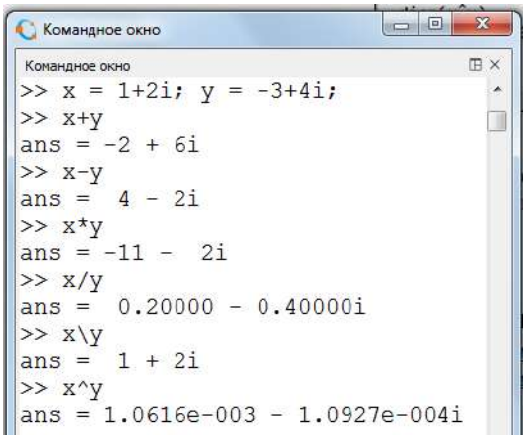


Рис. 7.

Практически все элементарные математические функции вычисляются при комплексных значениях аргумента и получают в результате этого комплексные значения результата.

Благодаря этому, например, функция sqrt вычисляет, в отличие от других языков программирования, квадратный корень из отрицательного аргумента, а функция abs при комплексном значении аргумента вычисляет модуль комплексного числа.

В Octave есть несколько дополнительных функций, рассчитанных только на комплексный аргумент:

real(Z) - выделяет действительную часть комплексного аргумента Z;

imag(Z) - выделяет мнимую часть комплексного аргумента;

angle(Z) - вычисляет значение аргумента комплексного числа Z (в радианах в диапазоне от -π до +π);

conj(Z) - выдает число, комплексно сопряженное относительно Z.

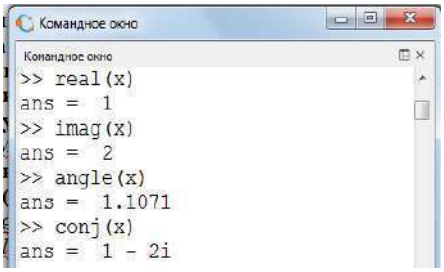


Рис. 8.

2. Простые матрично-векторные операции

Octave - система, специально предназначенная для осуществления сложных вычислений с векторами, матрицами и полиномами. Под вектором в Octave понимается одномерный массив чисел, а под матрицей - двумерный массив. При этом по умолчанию предполагается, что любая заданная переменная является вектором или матрицей. Например, отдельное заданное число система воспринимает как матрицу размером (1*1), а вектор-строку из N элементов - как матрицу размером (1*N).

Начальные значения векторов можно задавать с клавиатуры путем поэлементного ввода. Для этого в строке следует сначала указать имя вектора, потом поставить знак присваивания ' = ', затем, - открывающую квадратную скобку, а за ней ввести заданные значения элементов вектора, отделяя их пробелами или запятыми. Заканчивается строка записью закрывающей квадратной скобки.

Вектор-строку можно создать выполнением команды

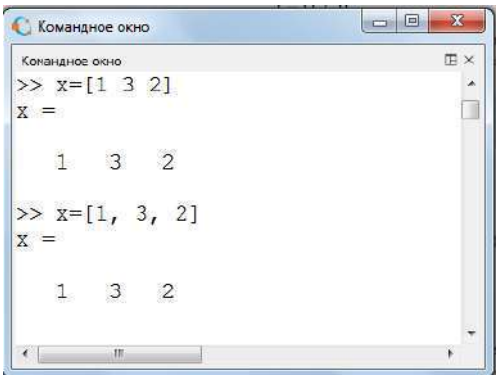


Рис. 9.

которые создают вектор $\vec{v} = [1, 2, 3]$. Векторы-строки ориентированы по горизонтали. В обратном случае, векторы-столбцы ориентированы по вертикали и создаются следующей командой

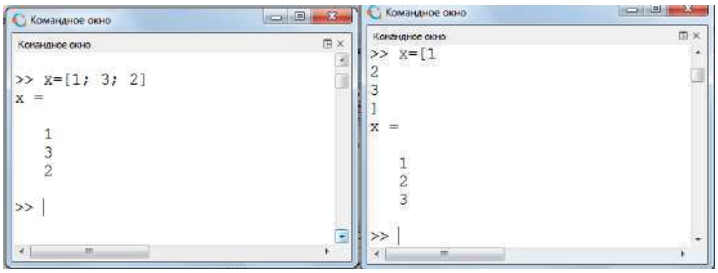


Рис. 10.

где символ «;» означает перенос на следующую строку. Кроме того, *Enter* может быть использован для индикации следующей строки. Тога эти две команды эквиваленты и создают один вектор

$$v = \begin{bmatrix} 1 \\ 2 \\ 3 \end{bmatrix}$$

Векторы могут быть созданы с помощью оператора двоеточие «:», как

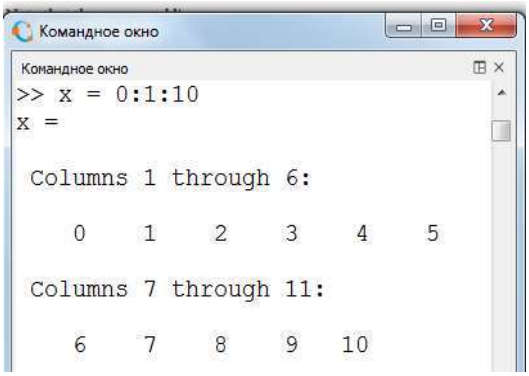


Рис. 11.

которая создает вектор с элементами от 1 до 10

$$\vec{v} = [0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10].$$

Команды

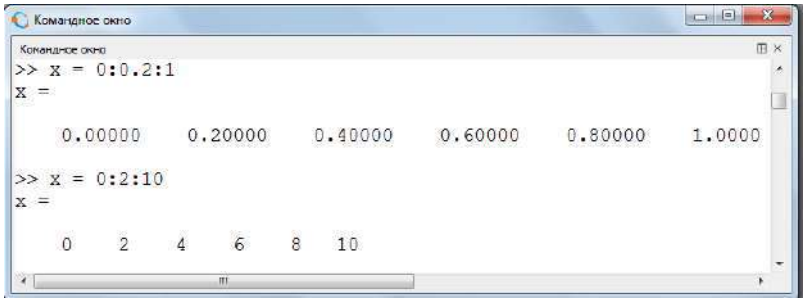


Рис. 12.

создают векторы от 0 до 1 и от 0 до 10 с шагом 0.2 и 2 соответственно

$$\vec{v} = [0, 0.2, 0.4, 0.6, 0.8, 1].$$

$$\vec{v} = [0, 2, 4, 6, 8, 10].$$

Таким образом, структуры, связанные с оператором «:» формируют векторы. Общий вид команды *x = a:h:b,* где a – начальное значение, b – конечное значение, h – шаг. Стоить отметить, что выбрав начальное значение a и шаг h конечное значение не будет достигнуто. Пример

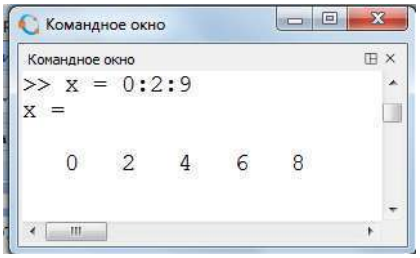


Рис. 13.

Данная команда создаст вектор $\vec{v} = [0, 2, 4, 6, 8]$. Конечное значение b=9 не будет создано в этом случае, поскольку стартовое значение a=0 и шаг h=2 не соразмерен со значением b=9. Если шаг не указан, то предполагается, что он равен 1. Например, команда

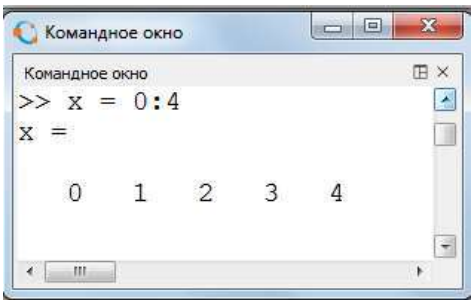


Рис. 14.

Создаст вектор $\vec{v} = [0, 1, 2, 3, 4]$.

Матрицы также просто генерировать. В матрице имеется определенное число строк *N* и столбцов *M*. Такая матрица будет иметь размерность *N*×*M,* например 3×3 матрица

$$A = \begin{bmatrix} 1 & 3 & 2 \\ 5 & 6 & 7 \\ 8 & 3 & 1 \end{bmatrix}$$

может быть создана оператором «;»

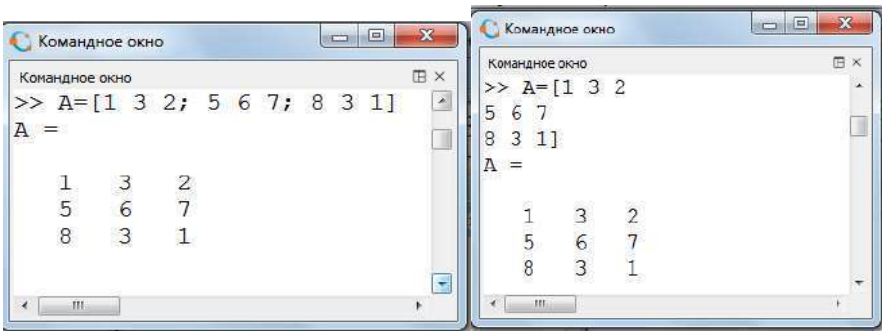


Рис. 15.

или с помощью клавиши *Enter*.

Обращение к любому элементу заданной матрицы в Octave осуществляется путем указания (в скобках, через запятую) после имени матрицы двух целых положительных чисел, которые определяют соответственно номера строки и столбца матрицы, на пересечении которых расположен этот элемент: **A**(*i*,*j*), где i – обозначает номер строки, j – обозначает номер столбца. Чтобы получить доступ ко второй строке и третьему столбцу используется команда A(2,3), которая возвратит элемент 7.

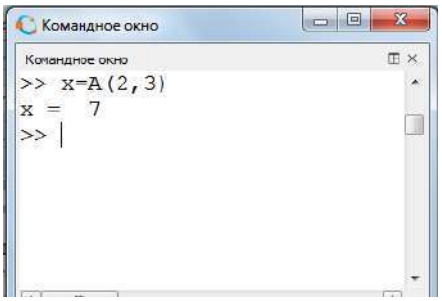


Рис. 16.

Если нужно, наоборот, установить на это место некоторое число, например, π, то это можно сделать так:

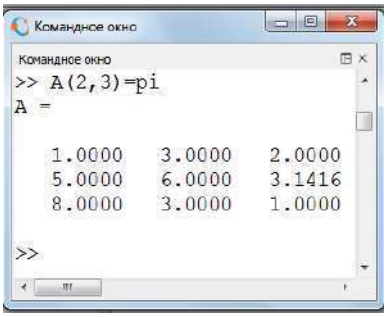


Рис. 17.

Чтобы создать вектор, состоящий из элементов второй строки матрицы **A**, поступают так:

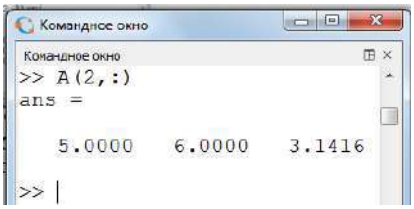


Рис. 18.

Пусть нужно создать вектор $\overrightarrow{v_1}$, состоящий из элементов третьего столбца последней матрицы **A**. Для этого произведем такие действия:

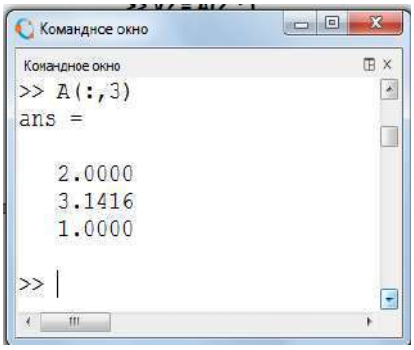


Рис. 19. Рис.

Допустим, что необходимо из матрицы **A** образовать матрицу **B** размером (2×2), которая состоит из элементов левого нижнего угла матрицы **A**. Тогда делают так:

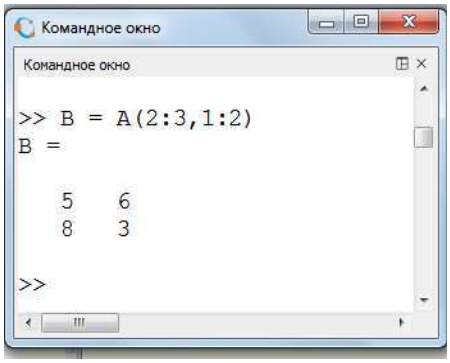


Рис. 20. Рис.

Аналогично можно вставить матрицу **B** в верхнюю середину матрицы **A**:

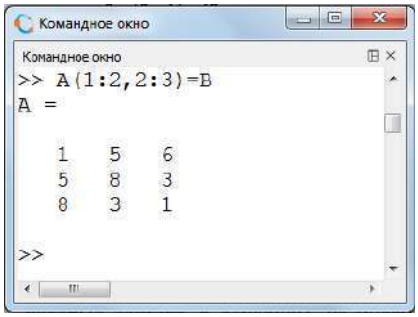


Рис. 21. Рис.

Если верхней границей изменения номеров элементов матрицы является ее размер в этом измерении, вместо него можно использовать служебное слово **end.** Например:

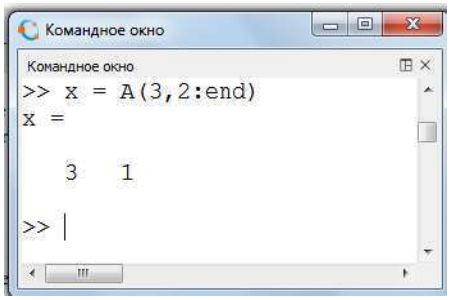


Рис. 22. Рис.

Такая команда извлекает значения из третьей строки и начиная со 2 и до последнего (третьего) столбца и присваивает их вектору $\vec{x} = [2, 3]$.

В качестве последнего примера, рассмотрим, оператор транспонирования «’», который переводит строки в столбцы и наоборот, столбцы в строки. В этом примере команда

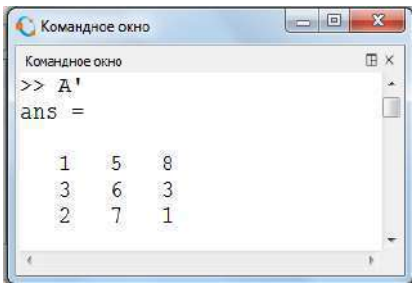


Рис. 23.

Записывает первую строку матрицы **A** записывает первым столбцом, вторую строку – вторым столбцам и т.д.

Стоить отметить важный комментарий о функции транспонирования. В частности, когда транспонируется вектор комплексных чисел с помощью оператора «’», то происходит комплексное сопряжение чисел. Если провести транспонирование с помощью оператора поэлементного «.’», то комплексного сопряжения не происходит.

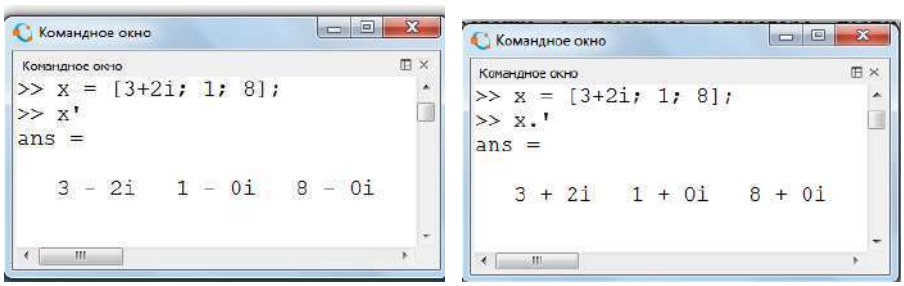


Рис. 24.

В качестве последнего примера манипуляций векторами/матрицами, рассмотрим следующий вектор $x = [-1\ 2\ 3\ 5\ -2]$. Этот вектор содержит положительные и отрицательные элементы. Следующие команды могут быть весьма полезными для манипулирования такими векторами и определения границ значений

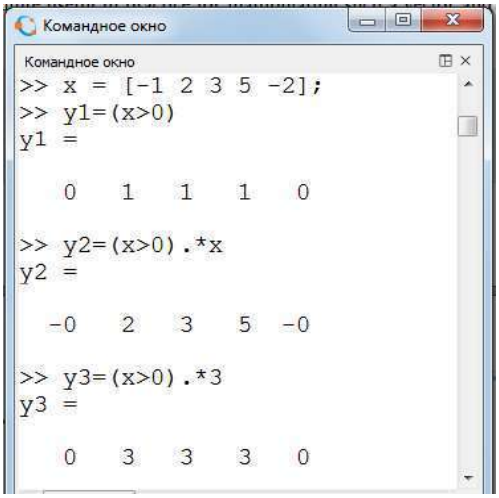


Рис. 25.

Эти команды получают векторы

y_1 = [0 1 1 1 0]
y_2 = [0 2 3 5 0]
y_3 = [0 3 3 3 0]

Данные команды определяют очень удобный способ быстрого поиска в векторах и матрицах при логических операциях.

3. Формирование векторов и матриц

Octave имеет несколько функций, которые позволяют формировать векторы и матрицы некоторого определенного вида. К таким функциям относятся:

zeros(M, N) - создает матрицу размером (M*N) с нулевыми элементами, например:

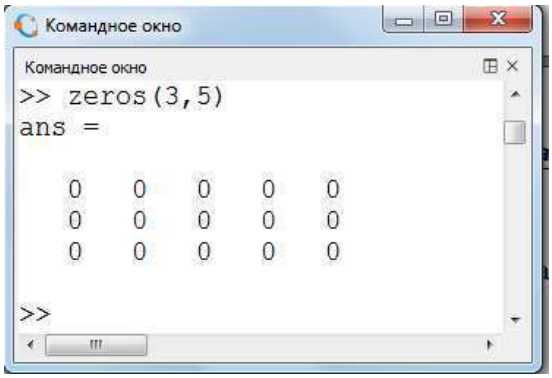


Рис. 26.

ones(M,N) - создает матрицу размером (M*N) с единичными элементами, например:

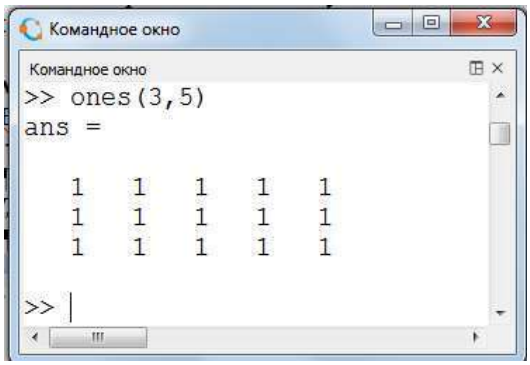


Рис. 27.

eye(M,N) - создает единичную матрицу размером (M*N), т. е. с единицами по главной диагонали и остальными нулевыми элементами, например:

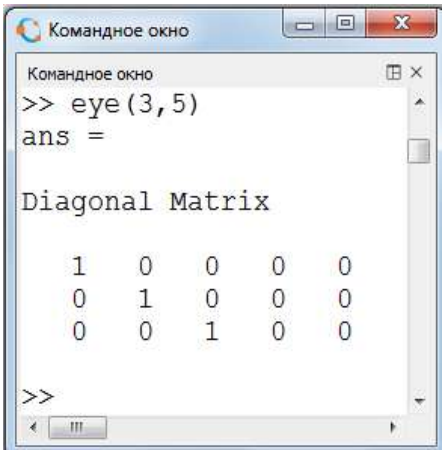


Рис. 28. Рис.

rand(M,N) - создает матрицу размером (M*N) из случайных чисел, равномерно распределенных в диапазоне от 0 до 1, например:

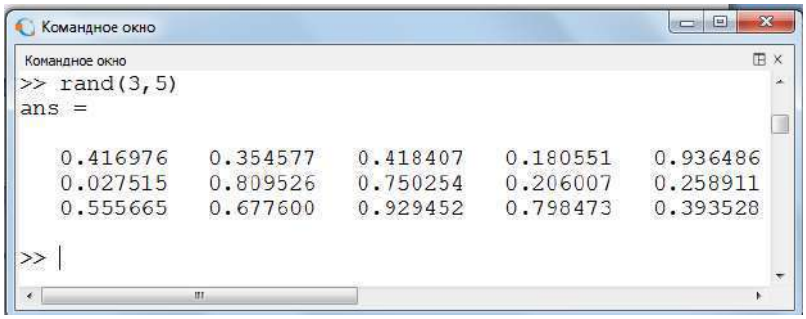


Рис. 29. Рис.

randn(M,N) - создает матрицу размером (M*N) из случайных чисел, распределенных по нормальному (гауссовому) закону с нулевым математическим ожиданием и стандартным (среднеквадратичным) отклонением, равным единице, например:

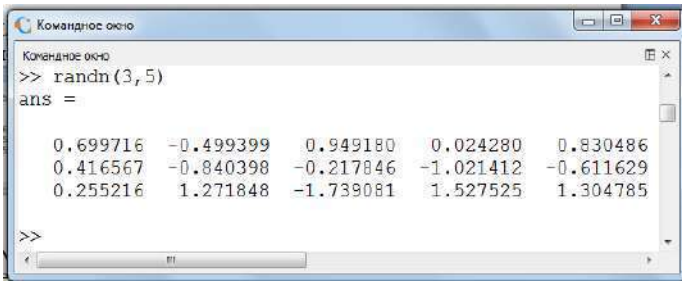


Рис. 30. Рис.

Функция **zeros**(1,N) формирует (создает) вектор-строку из N нулевых элементов. Аналогично **zeros(N,1)** создает вектор-столбец из N нулей. Векторы, значения элементов которых являются случайными равномерно распределенными, формируются таким образом: **rand(1,N)** - для вектора-строки и **rand(M,1)** - для вектора-столбца.

4. Действия над векторами и матрицами

Будем различать две группы действий над векторами:

а) векторные действия - т. е. такие, которые предусмотрены векторным исчислением в математике;

б) *действия по преобразованию элементов* - это действия, которые преобразуют элементы вектора, но не являются операциями, разрешенными математикой.

Векторные действия над векторами

Сложение векторов. Как известно, суммироваться могут только векторы одинакового типа (т. е. такие, которые являются или векторами-строками, или векторами-столбцами), имеющие одинаковую длину (т. е. одинаковое количество элементов). Если X и Y - именно такие векторы, то их сумму Z можно получить, введя команду Z = X + Y, например:

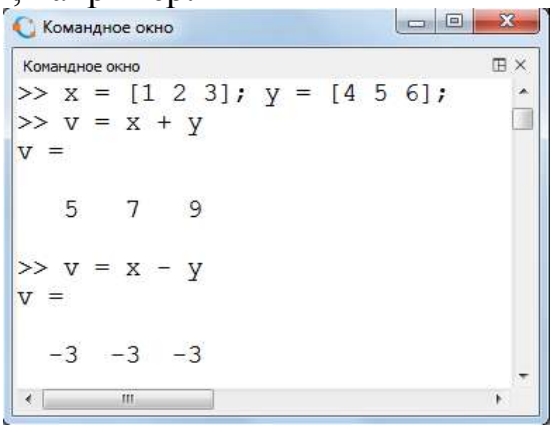


Рис. 31.

Аналогично с помощью арифметического знака ' - ' осуществляется *вычитание векторов*, имеющих одинаковую структуру (Z = X - Y).

Умножение вектора на число осуществляется в Octave с помощью знака арифметического умножения ' * ' таким образом: $\vec{Z} = \vec{X} * r$ или $\vec{Z} = r * \vec{X}$, где r - некоторое действительное число.

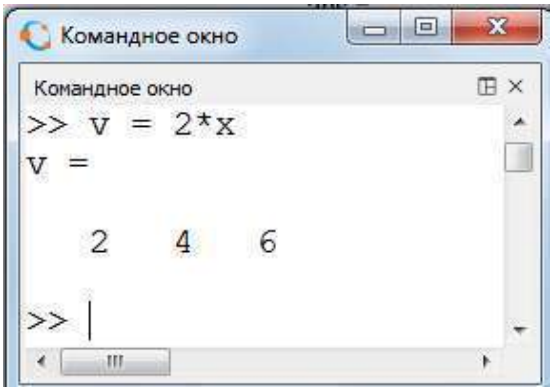


Рис. 32.

Умножение двух векторов определено в математике только для векторов одинакового размера (длины) и лишь тогда, когда один из векторов-множителей

строка, а второй - столбец. Иначе говоря, если векторы $\vec{X}$ и $\vec{Y}$ являются строками, то математическое смысл имеют лишь две формы умножения этих векторов: $\vec{U} = \vec{X}' * \vec{Y}$ и $\vec{V} = \vec{X} * \vec{Y}'$. При этом в первом случае результатом будет квадратная матрица, а во втором - число. В MatLAB умножение векторов осуществляется применением обычного знака умножения ' * ', который записывается между множителями-векторами.

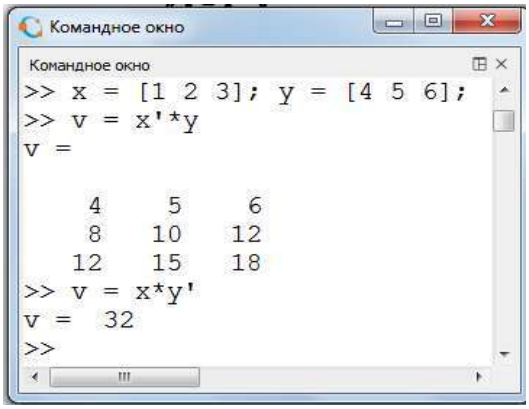


Рис. 33.

Кроме этих операций в MatLAB предусмотрено несколько операций поэлементного преобразования, осуществляемых с помощью знаков обычных арифметических действий. *Эти операции применяются к векторам одинакового типа и размера*. *Результатом их есть вектор того же типа и размера*.

Добавление (отнимание) числа к (из) каждому элемента вектора. Осуществляется с помощью знака ' + ' (' - ').

Поэлементное умножение векторов. Проводится с помощью совокупности знаков ' .* ', которая записывается между именами перемножаемых векторов. В результате получается вектор, каждый элемент которого является произведением соответствующих элементов векторов - "сомножителей".

Поэлементное деление векторов. Осуществляется с помощью совокупности знаков './ '. Результат - вектор, каждый элемент которого является частным от деления соответствующего элемента первого вектора на соответствующий элемент второго вектора.

Поэлементное деление векторов в обратном направлении. Осуществляется с помощью совокупности знаков '.\ '. В результате получают вектор, каждый элемент которого является частным от деления соответствующего элемента второго вектора на соответствующий элемент первого вектора.

Поэлементное возведение в степень. Осуществляется с помощью совокупности знаков '.^ '. Результат - вектор, каждый элемент которого является соответствующим элементом первого вектора, возведенным в степень, величина которой равняется значению соответствующего элемента второго вектора. Примеры:

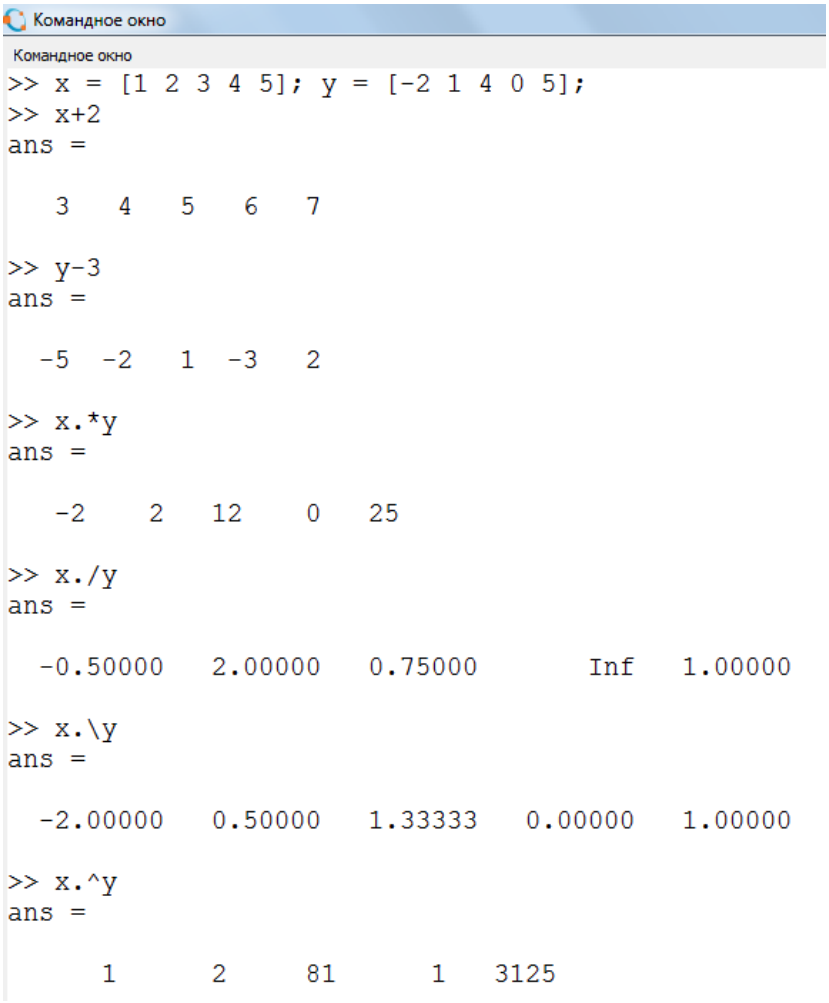


Рис. 34.

Вышеуказанные операции позволяют очень просто вычислять (а затем - строить графики) сложные математические функции, не используя при этом операторы цикла, т. е. осуществлять построение графиков в режиме калькулятора. Для этого достаточно задать значение аргумента как арифметическую прогрессию, а потом записать нужную функцию, используя знаки поэлементного преобразования векторов. Например, пусть нужно вычислить значения функции:

$$y = ae^{-hx}\sin(x)$$

при значениях аргумента x от 0 до 10 с шагом 1. Вычисление массива значений этой функции в указанных условиях можно осуществить с помощью лишь двух простых операторов:

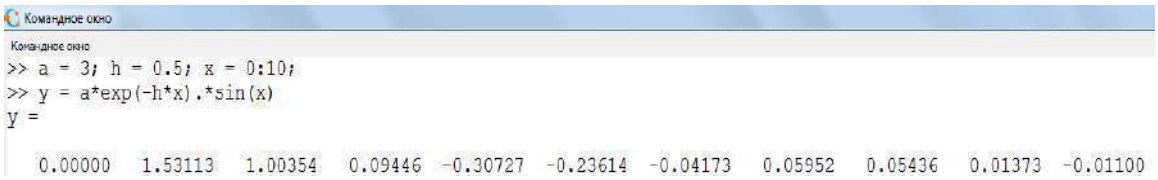


Рис. 35.

Кроме этого, в Octave определены операции *поэлементного умножения* матриц одинакового размера (совокупностью знаков ' .* ', записываемой между именами перемножаемых матриц), *поэлементного деления* (совокупности './ ' и '.\'), *поэлементного возведения в степень* (совокупность '.^'), когда каждый элемент

первой матрицы возводится в степень, равную значению соответствующего элемента второй матрицы.

К матричным действиям над матрицами относят такие операции, которые используются в матричном исчислении в математике и не противоречат ему.

Базовые действия с матрицами - сложение, вычитание, транспонирование, умножение матрицы на число, умножение матрицы на матрицу, возведение матрицы в целую степень - осуществляются в языке Octave с помощью обычных знаков арифметических операций. При использовании этих операций *важно помнить* условия, при которых эти операции являются возможными:

при сложении или вычитании матрицы должны иметь одинаковые размеры;

при умножении матриц количество столбцов первой матрицы должно совпадать с количеством строк второй матрицы.

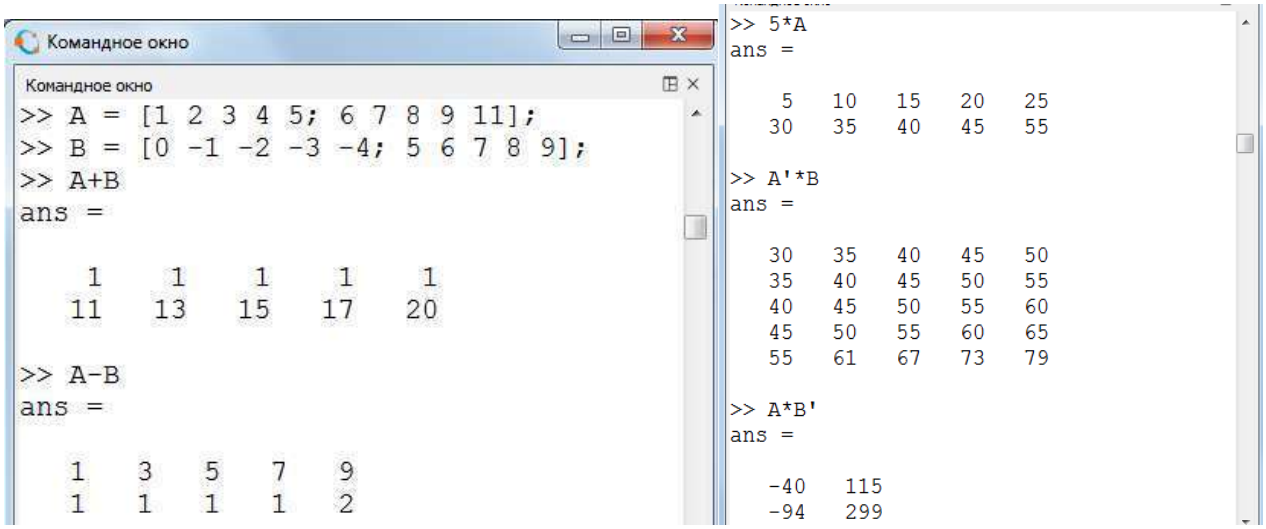


Рис. 36.

5. Логические операции, циклы и итерации

Основные строительные блоки любой программы, написанной на языке Octave/Octave или на другом языке программирования – это операторы ***for*** и ***if***. Они формируют каркас для выполнения сложный операций. Данный параграф сфокусирован на использовании этих двух логических структур в программировании.

Для иллюстрации использования цикла for, рассмотрим несколько несложных программ. Начнем конструирование циклов, которые рекурсивно суммируют числа. Общий формат таких циклов:

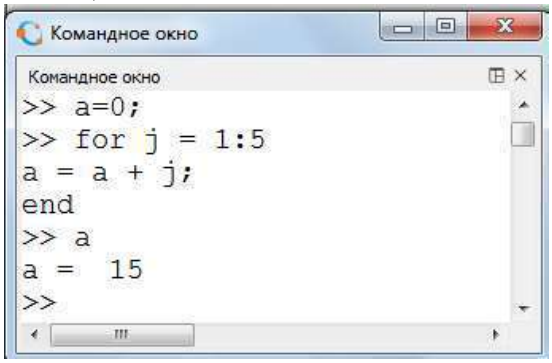


Рис. 37.

Последняя программа начинается инициализацией нулем переменной *a*. Затем происходит проход пять раз тело цикла for, т.е. счетчик j последовательно принимает значения 1, 2, 3, 4 и 5. В цикле значение переменной *a* обновляется путем прибавления текущего значения j. Т.О. начиная со стартового значения равного 0, оказывается, что переменная *a* последовательно приобретает значения 1 (j=1), 3 (j=2), 6 (j=3), 10 (j=4) и 15 (j=5).

По-умолчание увеличение переменной j происходит на один. Однако инкремент может быть изменен. Программа

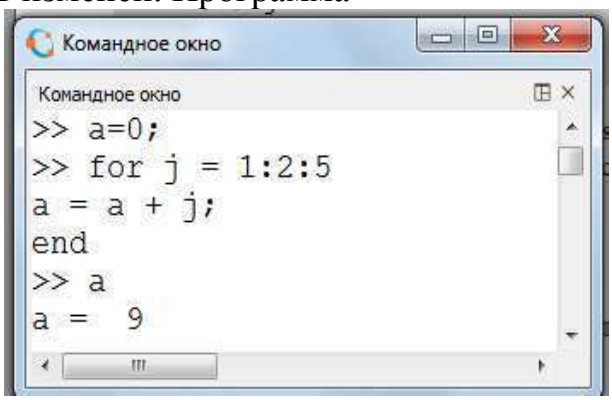


Рис. 38.

подобна предыдущей. Однако в последнем случае шаг инкремента изменен на 2. Т.О. начиная со значения a=0, переменная a приобретает значения 1 (j=4), 4 (j=3) и 9 (j=5). В более общем случае цикл for может быть инициализирован вектором-строкой.

Оператор if реализован подобным образом. Общий формат для оператора логического перехода следующий

if (логическое выражение)
(выражение для выполнения)
elseif (логическое выражение)
(выражение для выполнения)
elseif (логическое выражение)
(выражение для выполнения)
else
(выражение для выполнения)
end

Если *(логическое выражение)* является истинным, то выполняется *(выражение для выполнения)*, если ложно, то происходит на следующую ветвь *elseif (логическое выражение)* и т.д. Если все предыдущие логические оператор ложны, то выполняется последнее *(выражение для выполнения)*.

В качестве условия используются выражения типа:
<имя переменной1> <операция сравнения> <имя переменной2>

Операции сравнения в языке Octave могут быть такими:
< - меньше;
> - больше;
<= - меньше или равно;
>= - больше или равно;

= = - равно;
~ = - не равно.
Например:

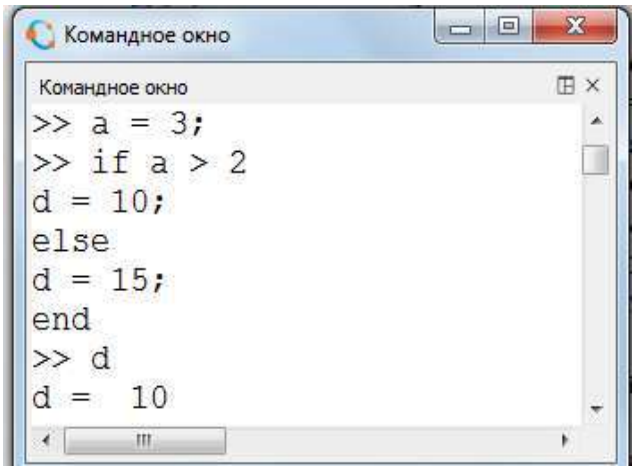


Рис. 39.

6. Вызов функций

Общая форма использования функции в Octave такова:
<*имя результата*> = <*имя функции*>(<*перечень аргументов или их значений*>).

В языке Octave предусмотрены некоторые элементарные арифметические функции.
sin(v) - синус каждого элемента вектора v;
cos(v) – косинус каждого элемента вектора v;
tan(v) – тангенс каждого элемента вектора v;
atan(v) - арктангенс (в диапазоне от -π/2 к +π /2);
exp(v) - экспонента каждого элемента вектора v;
log(v) - натуральный логарифм каждого элемента вектора v;
log10(v) - десятичный логарифм каждого элемента вектора v;
sqrt(v) - квадратный корень каждого элемента вектора v;
abs(v) - модуль каждого элемента вектора v.

7. Построение графиков

Вывод графиков в системе Octave - настолько простая и удобная операция, что ее можно использовать даже в режиме калькулятора. Основной функцией, обеспечивающей построение графиков на экране дисплея, является функция *plot*. Общая форма обращения к ней такова: *plot*(x1, y1, s1, x2, y2,...).

Здесь x1, y1 - заданные векторы, элементами которых являются массивы значений аргумента (x1) и функции (y1), отвечающие первой кривой графика; x2, y2 - массивы значений аргумента и функции второй кривой и т.д. При этом предполагается, что значения аргумента откладываются вдоль горизонтальной оси графика, а значения функции - вдоль вертикальной оси.

Приведем пример. Пусть нужно вывести график функции
y = 3sin(x +π/3) на промежутке от -3π до +3π с шагом π /100.
Сначала надо сформировать массив значений аргумента x:

x = -3*pi : pi/100 : 3*pi,
потом вычислить массив соответствующих значений функции:y = 3*sin(x+pi/3)
и, наконец, построить график зависимости y(x). В командном окне последовательность операций будет выглядеть так:

>> x = -3*pi : pi/100 : 3*pi;
>> y = 3*sin(x + pi/3);
>> plot(x, y)

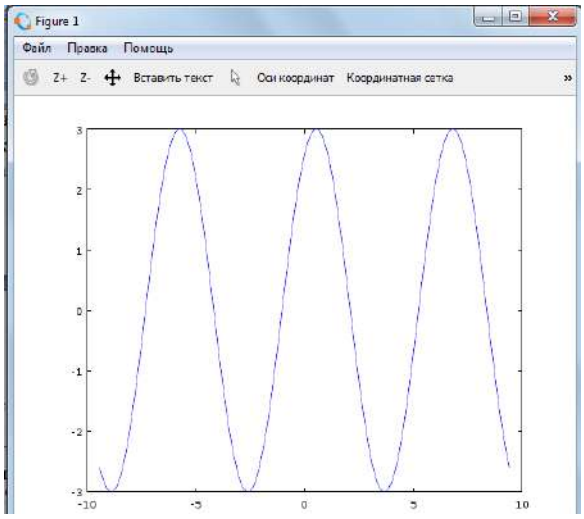


Рис. 40.

8. Методические материалы

Методические указания для студентов по подготовке к лабораторным работам

Лабораторная работа – небольшой научный отчёт, обобщающий проведенную студентом работу, которую представляют для защиты преподавателю. К лабораторным работам предъявляется ряд требований, основным из которых является исчерпывающее описание всей проделанной работы, позволяющее судить о полученных результатах, степени выполнения заданий и профессиональной подготовке студентов.

В отчёт по лабораторной работе должны быть включены следующие пункты:
1. титульный лист;
2. цель работы;
3. краткие теоретические сведения;
4. описание экспериментальной установки и методики эксперимента;
5. экспериментальные результаты;
6. анализ результатов работы;
7. выводы.

Лабораторная работа № 1
Представление сигналов во времени с помощью языка программирования Octave

Цель работы: Изучение основных конструкций, принципов работы и функций языка программирования Octave.

Краткие теоретические сведения: Octave представляет интерактивный командный интерфейс для решения линейных и нелинейных математических задач, а также проведения других численных экспериментов. Кроме того, Octave можно использовать для пакетной обработки. Язык Octave оперирует арифметикой вещественных и комплексных скаляров и матриц, имеет расширения для решения линейных алгебраических задач, нахождения корней систем нелинейных алгебраических уравнений, работы с полиномами, решения различных дифференциальных уравнений, интегрирования систем дифференциальных и дифференциально-алгебраических уравнений первого порядка, интегрирования функций на конечных и бесконечных интервалах. Этот список можно легко расширить, используя язык Octave (или используя динамически загружаемые модули, созданные на языках C, C++, Фортран и др.).

Задание на выполнение работы.

Задание 1. Вычислите указанное арифметическое выражение. Укажите последовательность нажатия клавиш. Сравните полученный результат с приведенным ответом.

1. $\dfrac{\left(12\frac{1}{6}-6\frac{1}{27}-5{,}25\right)13{,}5+0{,}111}{0{,}02}$.

2. $\dfrac{\left(1\frac{1}{12}+2\frac{5}{32}+\frac{1}{24}\right):9{,}6+2{,}13}{0{,}0004}$.

3. $\dfrac{\left(6{,}6-3\frac{3}{14}\right)5\frac{5}{6}}{(21-1{,}25):2{,}5}$.

4. $\dfrac{2{,}625-\frac{2}{3}\cdot 2\frac{5}{14}}{\left(3\frac{1}{12}+4{,}375\right):19\frac{8}{9}}$.

5. $\dfrac{0{,}134+0{,}05}{18\frac{1}{6}-1\frac{11}{14}-\frac{2}{15}\cdot 2\frac{6}{7}}$.

6. $\dfrac{\left(58\frac{4}{15}-56\frac{7}{24}\right):0{,}8+2\frac{1}{9}\cdot 0{,}225}{8{,}75\cdot 0{,}6}$.

Задание 2. Проведите вычисления по заданной формуле при заданных значениях параметров. Укажите необходимую последовательность действий. Сравните полученный результат с приведенным ответом.

1. $3m^2 + \sqrt[3]{2n^2} : m$; а) $m = -\frac{14}{5}$, $n = tg\frac{\pi}{8}$; б) $m = 2{,}2 \cdot 10^{-2}$, $n = \frac{1}{3{,}1}$.

ОТВЕТ: а) 23,27; б) 26,938.

2. $\frac{4}{3}l^3 \sin^2\frac{\alpha}{2}\sqrt{\cos\alpha}$; а) $l = 1{,}7 \cdot 10^3$, $\alpha = 18°$; б) $l = \frac{16}{21}$, $\alpha = \frac{\pi}{5}$.

ОТВЕТ: а) 1. 5633e+008; б) 5. 0651e-002.

3. $\sqrt{\frac{a\sqrt{b}}{\sqrt[3]{tg\alpha}}}$; а) $a = 1{,}5$, $b = 0{,}8$, $\alpha = 61°$; б) $a = 3 \cdot 10^{-2}$, $b = 0{,}71$, $\alpha = \frac{3}{7}\pi$.

ОТВЕТ: а) 1. 0498e+000; б) 1. 2429e-001.

4. $\frac{3a^2\sqrt{6{,}8 \cdot (a-b)}}{4(a+b)^3}$; а) $a = 4{,}13 \cdot 10^{-1}$, $b = \frac{1}{261}$;

б) $a = \sin\frac{5\pi}{8}$, $b = -tg12°$

ОТВЕТ: а) 2. 9464e+000; б) 4. 9445e+000.

5. $\frac{c^3}{6}\cos\frac{\alpha}{2}\sqrt{\sin\alpha}$; а) $c = \lg 2{,}38$, $\alpha = \frac{\pi}{5}$; б) $c = e^{-0{,}3}$, $\alpha = 65°$.

ОТВЕТ: а) 3. 4657e-004; б) 2. 2120e-002.

6. $\sqrt{\frac{n^3}{16{,}3\sin\alpha\sin 2\alpha}}$; а) $n = 3{,}1516 \cdot 10^{-2}$, $\alpha = 5°$; б) $n = e^{3{,}5}$, $\alpha = \frac{2\pi}{13}$.

ОТВЕТ: а) 1. 1265e-002; б) 7. 6324e+001.

7. $5\sin 35° \sqrt{\frac{S^3 \cos 36°}{\pi^3 tg\alpha}}$; а) $S = \ln 3$, $\alpha = 44°$; б) $S = \frac{18}{25}$, $\alpha = \frac{7}{12}\pi$.

ОТВЕТ: а) 5. 4283e-001; б) 8. 9703e-018+ 1. 4650e-001i.

8. $\left|\lg(1+\sin\alpha) + \ln(1-\sin\beta)\right|$; а) $\alpha = \frac{3\pi}{7}$, $\beta = 83°$; б) $\alpha = \frac{2}{3}\pi$, $\beta = 16°$.

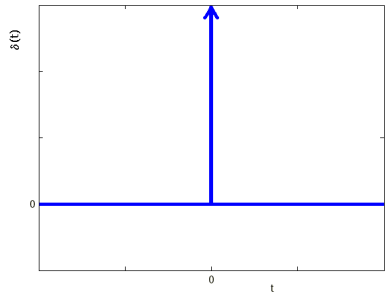
Задание 3. Вычислите значения функции f(x) на отрезке[a; b] с шагом h. Вывести график функции f(x) на экран.

Вариант	$f(x)$	a	b	h
1	$\frac{x^2}{1+0{,}25\sqrt{x}}$	1,1	3,1	0,2
2	$\frac{x^3-0{,}3x}{\sqrt{1+2x}}$	2,05	3,05	0,1
3	$\frac{2e^{-x}}{2\pi+x^3}$	0	1,6	0,16
4	$\frac{\cos \pi x^2}{\sqrt{1-3x}}$	-1	0	0,1
5	$\sqrt{1+4x}\sin \pi x$	0,1	0,8	0,07
6	$\frac{e^{x/3}}{1+x^2}$	1,4	2,4	0,1
7	$e^{-2x}+x^2-1$	0,25	2,25	0,2
8	$(e+x)\sin\left(\pi\sqrt{x-1}\right)$	1,8	2,8	0,1
9	$\sqrt{3+2x}\cdot tg\frac{\pi x^3}{2}$	0,1	0,9	0,08

Лабораторная работа № 2
Представление дискретных во времени основных сигналов

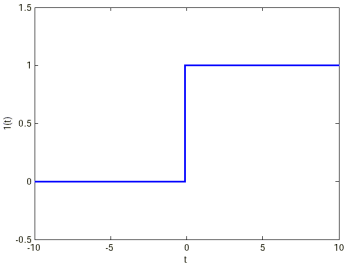
Цель работы: Представление и задание с помощью функций языка программирования Octave основных сигналов: дельта-импульс, функция Хевисайда, гармонический сигнал.

Краткие теоретические сведения: Де́льта-фу́нкция (или δ-функция, δ-функция Дирака, дираковская дельта, единичная импульсная функция) — обобщённая функция, которая позволяет записать точечное воздействие, а также пространственную плотность физических величин (масса, заряд, интенсивность источника тепла, сила и т. п.), сосредоточенной или приложенной в одной точке.



Фу́нкция Хевиса́йда (едини́чная ступе́нчатая функция, функция едини́чного скачка, включённая едини́ца) —кусочно-постоянная функция, равная нулю для отрицательных значений аргумента и единице — для положительных. В нуле эта функция, вообще говоря, не определена, однако её обычно доопределяют в этой точке некоторым числом, чтобы область определения функции содержала все точки действительной оси. Чаще всего неважно, какое значение функция принимает в нуле, поэтому могут использоваться различные определения функции Хевисайда, удобные по тем или иным соображениям, например[1]

$$\theta(x) = \begin{cases} 0, & x < 0; \\ 1, & x \geqslant 0. \end{cases}$$



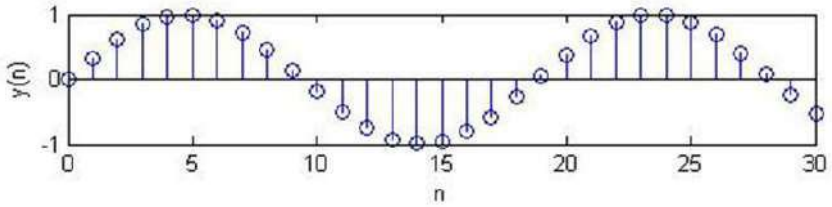
Гармонический сигнал — это гармонические колебания, со временем распространяющиеся в пространстве, которые несут в себе информацию или какие-то данные и описываются уравнением:

$y = A\cos(\omega t + \varphi_0)$

где A — амплитуда сигнала;

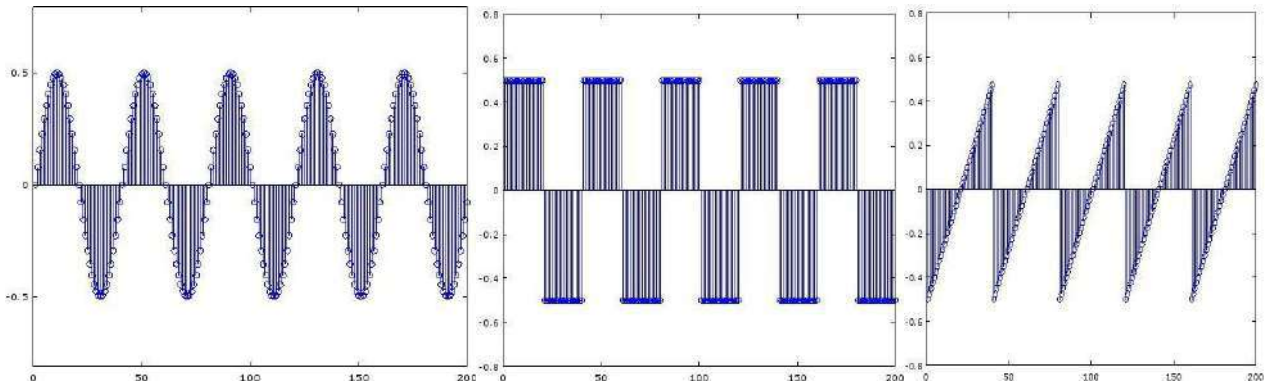
$\varphi = \omega t + \varphi_0$ — фаза гармонического сигнала;
t — время;
ω — циклическая частота сигнала;
Тем не менее, часто используют комплексную запись сигнала[1]:

$$y = A\exp[j(\omega t + \varphi_0)]$$



Задание на выполнение работы.

Задание 1. Необходимо написать Octave код, чтобы сгенерировать следующий дискретный во времени сигнал длиной N: x[n]=1/2sin(ωn),n=0...N−1 где N=8000 and ω=π/20.
Затем необходимо вывести график функции на экран. Сигнал x очень длинный, поэтому необходимо вывести только первые 200 отсчетов.
Задание 2. Необходимо написать Octave код, чтобы сгенерировать дискретный во времени сигнал прямоугольной формы, длиной N и частотой как в задании 1 (т.е. число периодов должно совпадать). Прямоугольная волна должна принимать значения от 1 до -1: 1 для первых 20 отсчетов, затем -1 для 20 отсчетов и т.д. Затем необходимо вывести график функции на экран. Сигнал x очень длинный, поэтому необходимо вывести только первые 200 отсчетов.
Задание 3. Необходимо написать Octave код, чтобы сгенерировать дискретный во времени сигнал треугольной формы, длиной N и частотой как в задании 1 (т.е. число периодов должно совпадать). Треугольная волна должна линейно увеличиваться от -1 до 0.9 для первых 20 отсчетов, затем снижаться от 1 до -0.9 для следующих 20 отсчетов и т.д.
Затем необходимо вывести график функции на экран. Сигнал x очень длинный, поэтому необходимо вывести на экран только первые 200 отсчетов.
Результатом работы должны быть следующего вида графики



Необходимые функций:

sin(Z) - вычисляет синус от значений элементов массива Z;

square(t) – генерирует прямоугольный сигнал с периодом 2π для значений времени, заданных входным вектором t. Функция square(t) работает аналогично функции sin(t), но вместо синусоидального сигнала генерирует прямоугольный сигнал с пиковыми значениями ±1.

sawtooth(t) - генерирует пилообразный сигнал с периодом 2π для значений времени, заданных входным вектором t. Функция sawtooth(t) работаете аналогично sin(t), но вместо синусоидального сигнала генерирует пилообразный, минимумы и максимумы которого равны -1 и 1 соответственно.

stem(y) – выводит график вектора $\vec{y}$ на экран.

Лабораторная работа № 3
Скалярное произведение сигналов. Корреляция

Цель работы: С помощью функции корреляции найти положение заданного отрывка в самом сигнале.

Краткие теоретические сведения:

Часто возникает задача обнаружения одного сигнала в другом. Например, может быть известно, что некоторое внешнее событие генерирует в датчике сигнал определенной формы. Однако события могут приходить почти одновременно, а сигналы от них – перекрываться. Кроме того, на выходе датчика может присутствовать шум, затрудняющий нахождение нужных сигналов. Для надежного обнаружения таких сигналов применяется метод корреляции (correlation).

Пусть датчик генерирует сигнал x[n], и мы хотим обнаружить в нем последовательность g[n] некоторой конечной длины. Для поиска этой последовательности вычисляются скалярные произведения сигналов x[n] и g[n-k] для различных k. То есть мы как бы пытаемся «приложить» искомый сигнал во всех возможных положениях к сигналу с датчика и найти их «степень похожести» (скалярное произведение) для каждого положения. Таким образом, на выходе мы получаем сигнал y[k], показывающий, насколько сигнал с датчика x[n] в позиции k похож на искомый сигнал g[n]. Формула вычисления корреляции такова:

$$y[k] = \sum_{i=-\infty}^{+\infty} g[i] \cdot x[i-k] .$$

Здесь суммирование можно проводить в конечных пределах, т.е. только для тех i, для которых искомый сигнал g[i] отличен от нуля. Бесконечные пределы суммирования, как и в уравнении свертки, записываются для общности (чтобы можно было проводить корреляцию с сигналом любой длины).

Смысл сигнала y[n] в том, что его величины для каждого n показывают, насколько входной сигнал в позиции n похож на искомый сигнал. Если во входном сигнале присутствует только шум, то и значения корреляции будут шумом небольшой амплитуды. Но как только в шуме входного сигнала появится форма, похожая на искомый сигнал, так значение корреляции в этой точке станет высоким.

Задание на выполнение работы.

Для заданной числовой последовательности большой длительности $\vec{g}$ и короткого фрагмента $\vec{x}$, взятого в произвольном месте $\vec{g}$. Причем длительность $\vec{x}$ много меньше, чем $\vec{g}$. Необходимо с помощью операции корреляции обнаружить место в $\vec{g}$, из которой была взята последовательность $\vec{x}$. График корреляции $\vec{y}$ вывести на экран.

Необходимые функций:

load имяфайля.mat - возвращает все переменные из MAT-файла в рабочее пространство Octave.
p=zeros(1,N) – создает вектор заданной длины N.
[y, I] = max(x) - возвращает наибольший элемент из массива X и сохраняет в переменной Y. Возвращает вектор-строку индексов I этих элементов в данном столбце.
n = norm(v, p) - вычисляет p-норму вектора v.
' – операция транспонирования.
x*y' - операция скалярного произведения векторов x и y.
Цикл **for … end** – необходим для прохождения всего заданного вектора $\vec{g}$ и вычисления корреляции фрагмента $\vec{g}$ с вектором $\vec{x}$. Результат вычисления сохраняется в векторе $\vec{p}$, созданном функцией **zeros**.
Приблизительный ход работы следующий:

```
octave:52> load D:  \jingle.mat
octave:53> load D:\  snippet.mat
octave:54> p=zeros(  );
octave:55> for i=0:
>
>     n=norm(jingle(1+i:  +i))*norm(snippet);
>     if n!=0
>         p(i+1)=(jingle(1+i:  +i)*snippet')/n;
>     end
> end
octave:56> [maxv,maxi]=max(p)
```

Лабораторная работа № 4
Дискретное преобразование Фурье

Цель работы: Изучение спектров различных дискретных сигналов с помощью функции fft. Определить влияние длительности сигналов и типа оконной функции на свойства дискретных спектров.

Краткие теоретические сведения:

Многие сигналы удобно анализировать, раскладывая их на синусоиды (гармоники). Тому есть несколько причин. Например, подобным образом работает человеческое ухо. Оно раскладывает звук на отдельные колебания различных частот. Кроме того, синусоиды являются«собственными функциями» линейных систем(т.к. они проходят через линейные системы, не изменяя формы, а изменяют лишь фазу и амплитуду). Еще одна причина в том, что теорема Котельникова формулируется в терминах спектра сигнала.

Преобразование Фурье – это разложение функций на синусоиды(далее косинусные функции мы тоже называем синусоидами, т.к. они отличаются от«настоящих» синусоид только фазой). Существует несколько видов преобразования Фурье.

1. Непериодический непрерывный сигнал можно разложить в интеграл Фурье.
2. Периодический непрерывный сигнал можно разложить в бесконечный ряд Фурье.
3. Непериодический дискретный сигнал можно разложить в интеграл Фурье.
4. Периодический дискретный сигнал можно разложить в конечный ряд Фурье.

Компьютер способен работать только с ограниченным объемом данных, следовательно, реально он способен вычислять только последний вид преобразования Фурье. Рассмотрим его подробнее.

Пусть дискретный сигнал x[n] имеет период N точек. В этом случае его можно представить в виде конечного ряда (т.е. линейной комбинации) дискретных синусоид. Разложение сигнала на синусоиды(т.е. получение коэффициентов) называется прямым преобразованием Фурье. Обратный процесс– синтез сигнала по синусоидам– называется обратным преобразованием Фурье.

Пусть x[n], n=0,…,N-1 – исходный комплексный сигнал, состоящий из N комплексных чисел. Обозначим X[k], k=0,…N-1 – его комплексный спектр, также состоящий из N комплексных чисел. Тогда справедливы следующие формулы прямого и обратного преобразований Фурье:

$$X[k] = \sum_{n=0}^{N-1} x[n] \cdot e^{-jnk(2\pi/N)}$$

$$x[n] = \frac{1}{N} \sum_{k=0}^{N-1} X[k] \cdot e^{jnk(2\pi/N)}$$

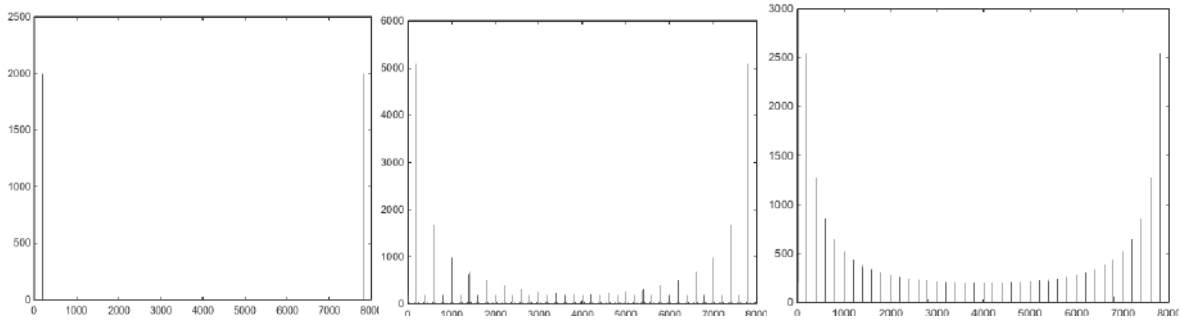
Если по этим формулам разложить в спектр действительный сигнал, то первые N/2+1 комплексных коэффициентов спектра будут совпадать (с точностью до

нормирующих множителей) со спектром«обычного» действительного ДПФ, представленным в«комплексном» виде, а остальные коэффициенты будут их симметричным отражением относительно половины частоты дискретизации. Для косинусных коэффициентов отражение четное, а для синусных – нечетное.

Задание на выполнение работы.

Для сигналов, сформированных в лабораторной работе 2, необходимо с помощью функции **fft** вычислить их комплексный спектр. Привести графики для амплитудного (с помощью функции abs) и фазового спектров (с помощью функции angle).

Результатом работы должны быть следующего вида графики



Необходимые функций:

Y = fft(X) - вычисляет для массива данных X дискретное преобразование Фурье. Y – комплексный спектр сигнала X.

fftshift(Y) - перегруппировывает выходные массивы функций fft Y, размещая нулевую частоту в центре спектра.

abs(Y) - возвращает массив модулей комплексных элементов Y. Таким образом, формируется АЧХ сигнала.

angle(Y) - возвращает массив значений аргументов для элементов Y. Таким образом, формируется ФЧХ сигнала.

Лабораторная работа № 5
Проектирование дискретных во времени фильтров

Цель работы: Изучение различных типов фильтров. Изучение средств получения коэффициентов дискретных фильтров для заданных параметров сигнала.

Краткие теоретические сведения:

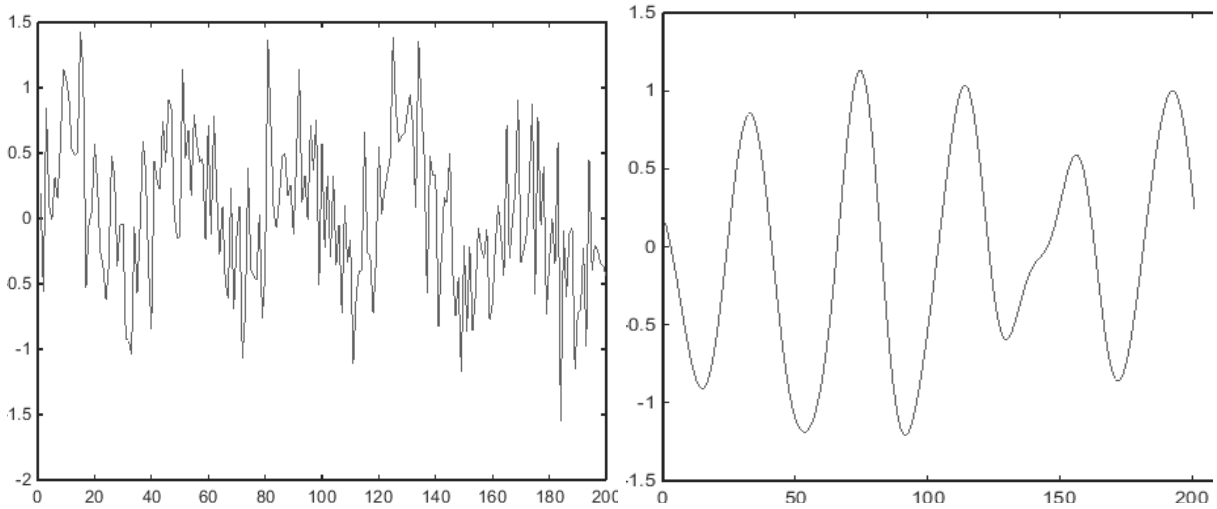
Цифровой фильтр — в электронике любой фильтр, обрабатывающий цифровой сигнал с целью выделения и/или подавления определённых частот этого сигнала. В отличие от цифрового, аналоговый фильтр имеет дело с аналоговым сигналом, его свойства недискретны, соответственно передаточная функция зависит от внутренних свойств составляющих его элементов.

Фильтр с конечной импульсной характеристикой (нерекурсивный фильтр, КИХ-фильтр) — один из видов электронных фильтров, характерной особенностью которого является ограниченность по времени его импульсной характеристики (с какого-то момента времени она становится точно равной нулю). Знаменатель передаточной функции такого фильтра — некая константа.

Фильтр с бесконечной импульсной характеристикой (рекурсивный фильтр, БИХ-фильтр) — электронный фильтр, использующий один или более своих выходов в качестве входа, то есть образует обратную связь.Основным свойством таких фильтров является то, что их импульсная переходная характеристика имеет бесконечную длину во временной области, а передаточная функция имеет дробно-рациональный вид. Такие фильтры могут быть как аналоговыми, так и **цифровыми**.

Задание на выполнение работы. С помощью функции **filter** произвести фильтрацию сигналов. К сигналам, сформированных в лабораторной работе 2, добавить гауссовский шум с помощью функции **randn**. Мощность шума должна быть равна 0.25, 0.125, 0.05 и 0.025. Затем получившийся зашумленный сигнал пропустить через фильтр нижних частот. Для этого определить наибольшую верхнюю частоту полезного сигнала и получить коэффициенты фильтра для данной верхней граничной частоты.

Результатом работы должны быть следующего вида графики



с шумом фильтрованный

Необходимые функций:

X = randn(m, n) - формирует массив размера m x n, элементами которого являются случайные величины, распределенные по нормальному закону с математическим ожиданием 0 и среднеквадратическим отклонением 1.

[b,a] = butter(n, Wn) - производит синтез дискретного фильтра Баттерворта n-го порядка, имеющего АЧХ фильтра нижних частот и нормированную частоту среза Wn.

freqz(b, a,…) - выводит в текущем графическом окне графики амплитудно-частотной и фазочастотной характеристик.

Y = filter(b,a,X) - фильтрует сигнал, заданный в виде одномерного массива X, используя дискретный фильтр с коэффициентами b,a.

Список литературы

1. Оппенгейм, А. Цифровая обработка сигналов / А. Оппенгейм, Р. Шафер. - 3-е изд., испр. – М.: Техносфера, 2012. – 1048 с.

2. Щетинин, Ю.И. Анализ и обработка сигналов в среде MATLAB: учебное пособие / Ю.И. Щетинин. – Новосибирск: НГТУ, 2011. – 115 с.

Учебно-методическое издание

Илья Владимирович Пешков

МЕТОДИЧЕСКИЕ УКАЗАНИЯ К ВЫПОЛНЕНИЮ ЛАБОРАТОРНЫХ РАБОТ ПО КУРСУ «ЦИФРОВАЯ ОБРАБОТКА СИГНАЛОВ»

Техническое исполнение – В.М. Гришин
Книга печатается в авторской редакции

Лицензия на издательскую деятельность
ИД № 06146. Дата выдачи 26.10.01.
Формат 60 x 84 /16. Гарнитура Times. Печать трафаретная
Печ.л. 2,3 Уч.-изд.л. 2,2
Тираж 300 экз. (1-й завод 1-20 экз.). Заказ 22

Отпечатано с готового оригинал-макета на участке оперативной полиграфии
Елецкого государственного университета им. И.А. Бунина

Федеральное государственное бюджетное образовательное учреждение
высшего образования
«Елецкий государственный университет им. И.А. Бунина»
399770, г. Елец, ул. Коммунаров, 28,1