ЛЕКЦИЯ 13
Тема СПЕЦИАЛИЗИРОВАННЫЕ СИСТЕМЫ КОМПЬЮТЕРНОЙ МАТЕМАТИКИ

В наши дни компьютерная математика получила должную известность и интенсивно развивается как передовое научное направление на стыке математики и информатики. Для подготовки программ таких расчетов еще совсем недавно использовались различные универсальные языки программирования. Но уже в начале 90-х годов на смену им пришли специализированные системы компьютерной математики (СКМ). Наибольшую известность получили системы Eureka, Mercury, Mathcad, Mathematica, Maple и др. Ежегодно появляются их новые версии.

Среди ряда современных СКМ особое место занимает математическая система MATLAB. Около двадцати лет работа сотен ученых и программистов была направлена на постоянное расширение его возможностей и совершенствование заложенных алгоритмов. В настоящее время с его библиотекой численных методов ни по объему, ни по качеству не может сравниться ни одна из систем.

Система MATLAB вобрала в себя не только передовой опыт развития и компьютерной реализации численных методов, накопленный за последние три десятилетия, но и весь опыт становления математики за всю историю человечества. На базе ядра MATLAB созданы многочисленные расширения, обеспечивающие моделирование и анализ систем в разнообразных сферах человеческой деятельности. Спектр проблем, исследование которых может быть осуществлено при помощи MATLAB и его расширений, охватывает: матричный анализ, обработку сигналов и изображений, проектирование и анализ радиочастотных цепей, задачи математической физики, оптимизационные задачи, финансовые задачи, обработку и визуализацию данных, работу с картографическими изображениями, нейронные сети, нечеткую логику и многое другое.

Ее охотно используют в своих научных проектах ведущие университеты и научные центры мира. Для современного инженера и научно-технического работника MATLAB является незаменимым инструментом моделирования и исследования различных прикладных систем, прежде всего, за счет использования готовых решений. В пакете MATLAB тщательно отработаны средства визуализации результатов вычислений и отображения различных графических объектов.

Система MATLAB

Система MATLAB создана таким образом, что любые (подчас весьма сложные) вычисления можно выполнять в режиме *прямых вычислений,* то есть без подготовки программы. Это превращает MATLAB в необычайно мощный калькулятор, который способен производить не только обычные для калькуляторов вычисления (например, выполнять арифметические операции и вычислять элементарные функции), но и операции с векторами и матрицами, комплексными числами, рядами и полиномами. Можно почти мгновенно задать и вывести графики различных функций – от простой синусоиды до сложной трехмерной фигуры.

Работа с системой в режиме прямых вычислений (или в командном режиме) носит диалоговый характер и происходит по правилу «задал вопрос, получил ответ». Пользователь набирает на клавиатуре вычисляемое выражение, редактирует его (если нужно) в командной строке и завершает ввод нажатием клавиши ENTER. В качестве примера на рис. 1.1 показаны простейшие вычисления.

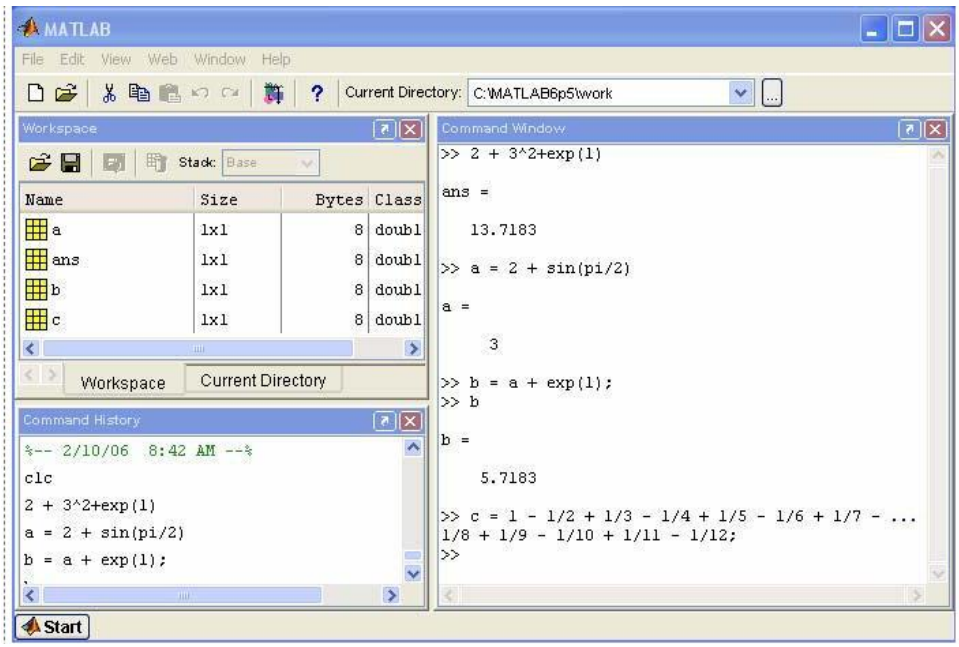

Рис. 2.1. Среда разработки на MATLAB.

Даже из таких простых примеров можно сделать некоторые выводы:

•для указания ввода исходных данных используется символ >>;

•данные вводятся с помощью простейшего строчного редактора;

•для блокировки вывода результата вычислений некоторого выражения после него надо установить знак «;» (точка с запятой);

•если не указана переменная для значения результата вычислений, то MATLAB назначает такую переменную с именем ans;

•знаком присваивания является привычный математикам знак равенства =, а не комбинированный знак :=, как во многих других языках программирования и математических системах;

•встроенные функции (например sin) записываются строчными буквами, и их аргументы указываются в *круглых скобках;*

Центральным понятием всех математических систем является *математическое выражение.* Оно задает то, что должно быть вычислено в численном (реже символьном) виде. Вот примеры простых математических выражений:

```
2+32.301*sin(x)  4+exp(3)/5  sqrt(y)/2  sin(pi/2)
```

Математические выражения строятся на основе чисел, констант, переменных, операторов, функций и разных спецзнаков.

Операторы и функции

Оператор – это специальное обозначение для определенной операции над данными – *операндами.* Например, простейшими арифметическими операторами являются знаки суммы +, вычитания –, умножения * и деления /. Операторы используются совместно с операндами. Например, в выражении 2+3 знак + является оператором сложения, а числа 2 и 3 – операндами.

Следует отметить, что большинство операторов относится к матричным операциям, что может служить причиной серьезных недоразумений. Например, операторы умножения * и деления / вычисляют произведение и частное от деления двух массивов, векторов или матриц. Есть ряд специальных операторов, например, оператор \ означает деление *справа налево,* а операторы* и

./ означают, соответственно, *поэлементное* умножение и *поэлементное* деление массивов. Полный список операторов можно получить, используя команду

```
>> help ops
```

Постепенно мы рассмотрим все операторы системы MATLAB и обсудим особенности их применения. А пока приведем только часть полного списка операторов, содержащую арифметические операторы:

`+` сложение, `-` вычитание,

- матричное умножение, `/` матричное деление, `\` обратное деление, `^` возведение в степень, `'` транспонирование

`.*` поэлементное умножение,
`./` поэлементное деление,
`.\` обратное поэлементное деление,
`.^` поэлементное возведение в степень
`.'` поэлементное транспонирование.

Приоритет в выполнении арифметических операций обычный: сначала (т.е. самый высший приоритет) – возведение в степень, затем – умножение и деление, и потом – сложение и вычитание. Операции одинакового приоритета выполняются слева направо, но круглые скобки могут изменить этот порядок.

Функции – это имеющие уникальные имена объекты, выполняющие определенные преобразования своих аргументов и при этом возвращающие результаты этих преобразований. *Возврат результата* — отличительная черта функций. При этом результат вычисления функции с одним выходным параметром подставляется на место ее вызова, что позволяет использовать функции в математических выражениях, например функцию `sin` в `2*sin(pi/2)`.

Функции в общем случае имеют список аргументов (параметров), заключенный в круглые скобки. Например, функция Бесселя записывается как `bessel(NU,X)`. В данном случае список параметров содержит два аргумента – `NU` в виде скаляра и `X` в виде вектора. Многие функции допускают ряд форм записи, отличающихся списком параметров. Если функция возвращает несколько значений, то она записывается в виде

```
[Y1,Y2,…]=func(X1,X2,…)
```

где `Y1,Y2,…` – список *выходных* параметров и `X1,X2,…` – список *входных* аргументов (параметров). Со списком элементарных функций можно ознакомиться, выполнив команду `help elfun`, а со списком специальных функций – с помощью команды `help specfun`. Функции могут быть *встроенными* (внутренними) и *внешними,* или *m-функциями.* Так, встроенными являются наиболее распространенные элементарные функции, например, `sin(x)` и `exp(y)`, тогда как функция `sinh(x)` является внешней функцией. Внешние функции содержат свои определения в m-файлах. Задание таких функций с помощью специального редактора m-файлов мы рассмотрим в главе 2. Встроенные функции хранятся в откомпилированном ядре системы MATLAB, в силу чего они
выполняются предельно быстро.

Функции пользователя.

Хотя в ядро MATLAB последних версий встроено уже около тысячи операторов и функций, пользователю всегда может понадобиться та или иная функций, простая или сложная, отсутствующая в ядре. Язык программирования систем MATLAB предоставляет ряд эффективных возможностей для задания функций пользователя.

Одна из таких возможностей заключается в применении функции `inline`, аргументом которой надо в апострофах задать выражение, задающее функцию одной или нескольких переменных. В приведенном ниже примере задана функция двух переменных – суммы квадратов sin(x) и cos(y):

```
>> sc2 = inline('sin(x).^2+cos(y).^2') sc2 =
Inline function:
sc2(x,y) = sin(x).^2+cos(y).^2
```

Можно также задавать свои функции в виде m-файлов. Подробно это описано в главе 10. Например, можно в окне редактора m-файлов (открывается командой New в меню File) создать m- файл с именем sc2 и листингом:

```
function y=sc2(x,y) y=sin(x).^2+cos(y).^2;
```

Записав его на диск, можно командой `type sc2` вывести листинг созданной функции:

```
>> type sc2 function y=sc2(x,y)
y=sin(x).^2+cos(y).^2;
```

Обращение к функции, созданной описанными методами, задается как sc2(x,y), где на место x и y подставляются значения переменных – аргументов функции пользователя. Например:

```
>>sc2(1,2) ans =
0.8313
>>sc2(2,1) ans =
1.1187
```

Способы формирование и операции на MATLAB.

Одним из самых заметных и важных преимуществ системы MATLAB над другими программными пакетами, ориентированными на вычисления и программирование это то, что она специально предназначенная для проведения сложных вычислений с векторами, матрицами и массивами – в том числе многомерными. При этом она по умолчанию предполагает, что каждая заданная переменная – это вектор, матрица или массив. Это значит, что в системе MATLAB даже переменные, принимающие единственное числовое значение, т.е. являющиеся по своей сути скалярами, в своем внутреннем представлении являются массивами, состоящими из единственного элемента. Например, если задано `X=1`, то это значит, что `X`– это массив с единственным элементом, имеющим значение `1`.

Двумерные массивы в математике принято называть *матрицами*. Любая строка и любой столбец матрицы являются одномерными массивами, которые также принято называть *вектором-строкой* и *вектором-столбцом* соответственно.

Рассмотрим способы формирования массивов.

Использование операции конкатенации.

Операция *конкатенации* является одним из самых важных способов создания массивов. Эта операция обозначается с помощью квадратных скобок `[9]`. Для формирования вектора-строки необходимо, чтобы элементы этого одномерного массива располагались между открывающей и закрывающей квадратными скобками и отделялись друг от друга либо пробелом или запятой. Так, например, присваивание

```
>> V=[1 2 3]
```

Или

```
>> V=[1,2,3] V =
1 2 3
```

задает вектор V, имеющий три элемента со значениями 1, 2 и 3. Право выбора в качестве *разделителя элементов* пробел или запятую остается за пользователем. Однако если элементами массива являются комплексные числа, то с точки зрения наглядности лучше использовать запятую.

Для формирования вектора-столбца необходимо, чтобы элементы этого одномерного массива отделялись друг от друга точкой с запятой. Пример:

```
>> U=[1;2;3] U =
1
2
3
```

Задание матрицы требует указания нескольких строк. Для разграничения строк используется знак ; (точка с запятой). Этот же знак в конце ввода предотвращает вывод матрицы или вектора (и вообще любой операции) на экран дисплея. Так, ввод

```
>> M=[1 2 3; 4 5 6; 7 8 9];
```

задает квадратную матрицу, которую можно вывести:

```
>> M

M =
                    1     2     3
                    4     5     6

                    7     8     9
```

Использование операции индексации.

Для доступа к индивидуальному элементу одномерного массива нужно применить операцию *индексации*, для чего после его имени указать в круглых скобках индекс (т.е. номер) элемента. Например, третий элемент вектора V и вектора U, обозначаются как V(3) и U(3) соответственно.

Для указания отдельного элемента матрицы M используется выражение вида M(k,m), где k,m обозначают номер строки и столбца соответственно. Например, если задать M(2,2), то результат будет равен 5. Поскольку вектор-строка является частным случаем матрицы с количеством строк, равным единице, а вектор-столбец является частным случаем матрицы с количеством столбцов, равным единицы, то обозначения V(3) и U(3) эквиваленты обозначениям V(1,3) и U(3,1)

Операция индексации позволяет непосредственно задавать элементы векторов и матриц. Если нужно присвоить элементу M(k,m) новое значение x, следует использовать выражение

```
M(k,m)=x
```

Например, если элементу M(2,2) надо присвоить значение 10, следует записать

```
>> M(2,2)=10;
```

Выражение M(k) с одним индексом дает доступ к элементам матрицы, развернутым в один столбец. Такая матрица образуется из исходной, если подряд выписать ее столбцы. Следующий пример поясняет такой доступ к элементам матрицы M:

```
>> M=[1 2 3; 4 5 6; 7 8 9] M =
                    1     2     3
                    4     5     6
                    7     8     9
```

```
>>M(2) ans =
4
>>M(9) ans =
9
>>M(5)=100;
>>M
M =
                    1     2     3
                    4   100     6
                    7     8     9
```

Использование оператора : (двоеточие).

Очень часто необходимо произвести формирование упорядоченных числовых последовательностей или задание диапазона значений. Для этого используется оператор: (двоеточие) – один из наиболее важных операторов в MATLAB. Общий синтаксис использования этого оператора имеет вид:

```
Начальное_значение:Шаг:Конечное_значение
```

Данная конструкция порождает возрастающую последовательность чисел, которая начинается с начального значения, идет с заданным шагом и завершается конечным значением. Если `Шаг` не задан, то он принимает значение 1. Если конечное значение указано меньшим, чем начальное значение – выдается сообщение об ошибке. Примеры применения оператора : для формирования вектор строк:

```
>>1:5 ans =
12 3 4 5
>>i=0:2:10
i=
O 2 4 6 8 10
>>i=10:-2:2
i=
108 6 4 2
>>V=0:pi/2:2*pi;
>>V
    V
    =
    0    1.5708    3.1416    4.7124    6.2832
>> X=1:-.2:0
X =
1.0000            0.8000 0.6000 0.4000 0.2000
0
```

Генерирование матриц

MATLAB имеет функции, которые позволяют создавать матрицы с заданными свойствами. Ниже приводятся примеры несколько таких функций:

`a1=ones(5,5)`	– матрица 5x5 с единичными матричными элементами,
`a2=zeros(4,6)`	– матрица 4x6 с нулевыми матричными элементами,
`a3=eye(4,4)`	– единичная матрица 4x4,
`a4=rand(7,8)`	– 7x8 матрица со случайными матричными элементами из интервала [0-1].
`a5=magic(5)`	– задает магическую матрицу размера 5x5, у которой сумма всех столбцов,

всех строк и даже диагоналей равна одному и тому же числу.

Объединение матриц

Объединение – это процесс соединения маленьких матриц в большую. Для этого также используется операция *конкатенации*. Например, создадим вначале магическую матрицу размера

```
3x3 :
>> A=magic(3) A =
        8     1     6
        3     5     7
        4     9     2
```

Теперь можно построить матрицу, содержащую четыре матрицы:

```
>> B=[A A+16; A+32 A+16]
  B =
    8     1     6      24  17    22
    3     5     7      19  21    23
    4     9     2      20  25    18
   40    33    38      24  17    22
   35    37    39      19  21    23
   36    41    34    20    25    18
```

Полученная матрица имеет уже размер `6x6`.

Операции над массивами на MATLAB

В традиционных языках программирования вычисления с массивами осуществляются поэлементно в том смысле, что нужно запрограммировать каждую отдельную операцию над отдельным элементом массива. В системе MATLAB допускаются мощные групповые операции над всем массивом сразу, что позволяет с одной стороны компактно задавать выражения, а с другой выполнять гигантский объем работы с высоким быстродействием.

Арифметические операции.

Стандартные арифметические операции (+,-,*) над массивами используются в системе MATLAB в смысле линейной алгебры. Поэтому сложение, вычитание или умножение двух массивов, накладывают определенные требования к ним. Так, например сложение + или вычитание – двух массивов, требует, чтобы они были одинакового размера или один из них был скаляром (размером 1x1). Перемножение * дух матриц выполнимо, когда число столбцов первого сомножителя равно числу строк второго. Кроме того, возможно перемножение матрицы любого размера на скаляр. Аналогичные ограничения накладываются и на другие операции. Дадим описание знаков операций:

+Сложение. При сложении матрицы должны иметь одинаковые размеры. К матрице любого размера можно прибавлять скаляр.

−Вычитание. При вычитании матрицы должны иметь одинаковые размеры. Из матрицы любого размера можно вычитать скаляр.

* Матричное умножение. Число столбцов первого сомножителя равно числу строк второго. / Деление матриц. A/B эквивалентно A*inv(B), inv - функция для вычисления обратной
матрицы.

\Левое деление. A\B эквивалентно inv(A)*B. Результатом операции A\B является решение системы линейных уравнений A*X=B.

^Возведение в степень. Вычисление целой степени квадратной матрицы.

' Транспонирование матриц. Если матрица комплексная, то выполняется транспонирование и комплексное сопряжение.

Наряду с вышеупомянутыми операциями над массивами целиком, в системе MATLAB существуют так называемые поэлементные операции, обозначаемые комбинацией двух символов: точка и соответствующий знак операции. Дадим их описание:

`.*` Поэлементное умножение. Матрицы должны иметь одинаковые размеры.

`./` Поэлементное деление. Матрицы должны быть одинаковых размеров.

`.\` Поэлементное левое деление. Матрицы должны быть одинаковых размеров.

`.^` Поэлементное возведение в степень. Матрицы должны быть одинаковых размеров.

`.'` Транспонирование матрицы. (Без комплексного сопряжения).

Вычисление функций от массивов.

Система MATLAB позволяет производить групповые вычисления над массивами, используя обычные математические функции, которые в традиционных языках программирования работают только со скалярными аргументами. В результате с помощью крайне компактных записей, удается произвести большой объем вычислений.

Как отмечалось, принадлежность MATLAB к матричным системам вносит коррективы в назначение операторов и приводит, при неумелом их использовании, к казусам. Рассмотрим следующий характерный пример:

```
>> x=0:5
x =
0     1 2 3   4                                    5
>> cos(x)
ans =
1.0000   0.5403     -0.4161 -0.9900 -0.6536 0.2837
>> sin(x)/x ans =
-0.0862
```

Вычисление массива косинусов здесь прошло корректно. А вот вычисление массива значений функции `sin(x)/x` дает неожиданный, на первый взгляд, эффект — вместо массива с шестью элементами вычислено единственное значение!

Причина «парадокса» здесь в том, что оператор `/` вычисляет отношение двух *матриц, векторов* или *массивов.* Если они одной размерности, то результат будет одним числом, что в данном случае и выдала система. Чтобы действительно получить вектор значений `sin(x)/x`, надо использовать специальный оператор *поэлементного* деления массивов – `./`. Тогда будет получен массив чисел:

```
>> sin(x)./x
Warning: Divide by zero. ans =
NaN 0.8415 0.4546 0.0470 -0.1892 -0.1918
```

Впрочем, и тут без особенностей не обошлось. Так, при x=`0` значение `sin(x)/x` дает устранимую неопределенность вида `0/0=1`. Однако, как и всякая численная система, MATLAB классифицирует попытку деления на 0 как ошибку и выводит соответствующее предупреждение. А вместо ожидаемого численного значения выводится символьная константа NaN, означающая, что неопределенность `0/0` – это все же не обычное число.

Множественная индексация массивов.

Как мы уже убедились, одна из уникальных возможностей системы MATLAB – это возможность осуществлять множественные вычисления обычными математическими функциями. В то же время, как мы уже говорили, для доступа к отдельному элементу массива используется операция *индексации*. Встает закономерный вопрос – а можно ли

получить доступ не к отдельному элементу массива, а к его подмножеству. Ответ – да. Это другая уникальная возможность системы MATLAB – *множественная индексация массивов*. Рассмотрим этот вопрос на конкретном примере.
Сформируем произвольный одномерный массив `V`:

```
V=10:10:100 V =
        10    20    30    40    50    60    70    80    90   100
```

Чтобы получить доступ к подмножеству элементов `U`, состоящего из каждого второго числа массива `V`, сформируем одномерный массив индексов:

```
>> Ind=2:2:10
Ind =
       2      4  6     8    10
```

Тогда решение поставленной задачи будет иметь вид:

```
>> U=V(Ind) U =
20 40 60 80 100
```

Более того, эту задачу можно было бы решить без промежуточной переменной `Ind`, а записать непосредственно:

```
>> U=V(2:2:10) U =
20 40 60 80 100
```

Приведем еще примеры использования множественной индексации массивов:

`V(1:5)` – первые пять элементов вектора `V`; `V(3:end)` – элементы вектора `V` с 3 до последнего; `V(:)` – все элементы вектора `V`;
`A(2,:)` – вторая строка матрицы `A`;
`A(:,3)` – третий столбец матрицы `A`;
`A(1,1:5)` – первые пять элементов первой строки матрицы `A`; `A(:,:)` – полностью дублирует исходную матрицу `A`;
Использование множественной индексации массивов позволяет удалять отдельные столбцы и строки матрицы. Проделаем это с матрицей `A`:

```
>> A=[1 2 3; 4 5 6; 7 8 9] A =
                            1     2     3
                            4     5     6
                            7     8     9
```

Удалим второй столбец, используя пустое множество – оператор []:

```
>> A(:,2)=[] A =
                              1     3
                              4     6
                              7     9
```

Аналогичную операцию можно использовать для удаления строк матрицы.

Функции для работы с матрицами.

В системе MATLAB предусмотрено большое количество функций предназначенных непосредственно для работы с матрицами. Полный список таких функций можно непосредственно получить, используя команды `help datafun` и `help elmat`. Здесь же мы ограничимся некоторыми из них:

`length(v)` – длина вектора `v`;
`sum(v)` – сумма всех элементов вектора `v`;
`prod(v)` – произведение всех элементов вектора `v`;
`max(v)` – нахождение максимального значения среди элементов вектора `v`;
`min(v)` – нахождение минимального значения среди элементов вектора `v`;
`mean(v)` – вычисление среднего арифметического элементов вектора `v`;
`sort(v)` – упорядочение элементов вектора по возрастанию;
`size(A)` – размер матрицы `A`;
`diag(A)` – выделение главной диагонали матрицы `A`;

Выводы

В среде MATLAB заложены достаточно мощные и удобные средства программирования. Программирование в среде MATLAB основано на использовании М-файлов-сценариев и М- файлов-функций.

В среде MATLAB заложена возможность поддержки функций с переменным числом аргументов.

Использование управляющих структур (условные операторы, циклы) и функции диалога позволяет организовывать нелинейные структуры программ.