Графика и визуализация данных в Matlab

MatLab обладает широким набором средств для построения графиков функций одной и двух переменных и отображения различных типов данных. Все графики выводятся в графические окна со своими меню и панелями инструментов. Вид графиков определяется аргументами графических команд и затем может быть изменен при помощи инструментов графического окна. Важно понимать, что для построения графиков функций на некоторой области изменения аргументов следует вычислить значения функции в точках области, часто для получения хороших графиков следует использовать достаточно много точек.

Разберем сначала, как получить график функции одной переменной, к примеру: *f*(x)=exsin(πx)+x2 на отрезке [-2, 2].

Первый шаг состоит в задании координат точек по оси абсцсс. Заполнение вектора x элементами с постоянным шагом при помощи двоеточия позволяет просто решить эту задачу. Далее необходимо поэлементно вычислить значения *f*(x) для каждого элемента вектора x и записать результат в вектор f. Для построения графика функции осталось использовать какую-либо из графических функций MatLab. Достаточно универсальной графической функцией является plot. В самом простом случае она вызывается с двумя входными аргументами - парой x и f (т. е. plot выводит зависимость элементов одного вектора от элементов другого). Последовательность команд, записанная ниже, приводит к появлению графического окна Figure No.1 с графиком функции (рис. ).

>> x=[-2:0.05:2];

>> f=exp(x).\*sin(pi\*x)+x.^2;

>>plot(x,f)

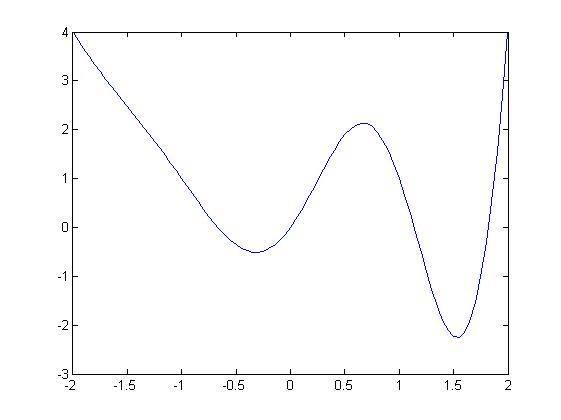


Рис. 2.1 График функции *f*(x)=exsin(πx)+x2

Тип линии, цвет и маркеры определяются значением третьего дополнительного аргумента функции plot. Этот аргумент указывается в апострофах, например, вызов plot(x,f,'ro:') приводит к построению графика красной пунктирной линией, размеченной круглыми маркерами. Обратите внимание, что абсциссы маркеров определяются значениями элементов вектора x. Всего в дополнительном аргументе может быть заполнено три позиции, соответствующие цвету, типу маркеров и стилю линии. Обозначения для них приведены в табл.. Порядок позиций может быть произвольный, допустимо указывать только один или два параметра, например, цвет и тип маркеров. Посмотрите на результат выполнения следующих команд: plot(x,f,'g'), plot(x,f,'ko'), plot(x,f,':').

Функция plot имеет достаточно универсальный интерфейс, она, в частности, позволяет отображать графики нескольких функций на одних осях.

Пусть требуется вывести график не только *f*(x), но и http://www.studfiles.ru/html/2706/468/html_tJY9QryHDy.kG10/img-IWbjAJ.pngна отрезке [-2, 2]. Сначала необходимо вычислить значения g(x):

>> g=exp(-x.^2).\*sin(5\*pi\*x);

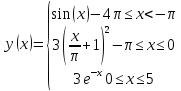
а затем вызвать plot, указав через запятую пары x, f и x, g и, при желании, свойства каждой из линий:

>>plot(x,f,'ko-', x,g,'k:')

Таблица 4.1 Сокращения для цвета, типа маркеров и стиля линий

|  |  |  |  |
| --- | --- | --- | --- |
| Цвет | | Тип маркера | |
| Y | Желтый | • | Точка |
| m | Розовый | ° | Кружок |
| c | Голубой | x | Крестик |
| r | Красный | + | Знак плюс |
| g | Зеленый | \* | Звездочка |
| b | Синий | s | Квадрат |
| w | Белый | d | Ромб |
| k | Черный | v | Треугольник вершиной вниз |
| Тип линии | | ^ | Треугольник вершиной вверх |
| — | Сплошная | < | Треугольник вершиной влево |
| : | Пунктирная | > | Треугольник вершиной вправо |
| —. | Штрих-пунктирная | p | Пятиконечная звезда |
| -- | Штриховая | h | Шестиконечная звезда |

Допускается построение произвольного числа графиков функций, свойства всех линий могут быть различными. Кроме того, области построения каждой из функций не обязательно должны совпадать, но тогда следует использовать разные вектора для значений аргументов и вычислять значения функций от соответствующих векторов. Для получения графика кусочно-заданной функции:



достаточно выполнить последовательность команд:

>> x1=[-4\*pi:pi/10:-pi];

>> y1=sin(x1);

>>x2=[-pi:pi/30:0];

>> y2=3\*(x2/pi+1).^2;

>>x3=[0:0.02:5];

>> y3=3\*exp(-x3)

>>plot(x1,y1,x2,y2,x3,y3)

Графики ветвей функции отображаются различными цветами. Можно было поступить и по-другому, а именно: после заполнения x1, y1, x2, y2, x3 и y3 собрать вектор x для значений аргумента и вектор y для значений y(x) и построить зависимость y от x:

>> x=[x1 x2 x3];

>> y=[y1 y2 y3];

>>plot(x,y)

Для сравнения поведения двух функций со значениями разных порядков удобно применять plotyy. Функция plotyy вызывается от двух пар входных аргументов (векторов) и приводит к появлению двух линий графиков, каждой из которых отвечает своя ось ординат.

Графики оформляются в MatLab специальными командами и функциями. Сетка наносится на оси командой gridon, а убирается при помощи gridoff.

Заголовок размещается в графическом окне посредством функции title, входным аргументом которой является строка, заключенная в апострофы:

>>title('Результаты эксперимента')

При наличии нескольких графиков требуется расположить легенду обратившись к legend. Надписи легенды, заключенные в апострофы, указываются во входных аргументах функции legend, их число должно совпадать с числом линий графиков. Кроме того, последний дополнительный входной аргумент определяет положение легенды:

-1 - вне графика в правом верхнем углу графического окна;

0 - выбирается лучшее положение в пределах графика так, чтобы как можно меньше перекрывать сами графики;

1 - в верхнем правом углу графика (это положение используется по умолчанию);

2 - в верхнем левом углу графика;

3 - в нижнем левом углу графика;

4 - в нижнем правом углу графика.

Функции xlabel и ylabel предназначены для подписей к осям, их входные аргументы так же заключаются в апострофы.

Обратимся теперь к визуализации векторных и матричных данных. Самый простой способ отображения векторных данных состоит в использовании функции plot с вектором в качестве входного аргумента. При этом получающийся в виде ломаной линии график символизирует зависимость значений элементов вектора от их индексов. Второй дополнительный аргумент может определять цвет, стиль линии и тип маркеров, например: plot(x,'ko'). Вызов функции plot от матрицы приводит к нескольким графикам, их число совпадает с числом столбцов матрицы, а каждый из них является зависимостью элементов столбца от их строчных индексов. Цвет и стиль линий и тип маркеров сразу для всех линий так же определяется вторым дополнительным аргументом.

Наглядным способом представления матричных и векторных данных являются разнообразные диаграммы. Простейшая столбцевая диаграмма строится при помощи функции bar:

>> x=[0.7 2.1 2.5 1.9 0.8 1.3];

>>bar(x)

Дополнительный числовой аргумент bar указывает на ширину столбцов (по умолчанию он равен 0.8), а значения большие единицы, например bar(x,1.2), приводят к частичному перекрытию столбцов. Указание матрицы во входном аргументе bar приводит к построению групповой диаграммы, число групп совпадает с числом строк матрицы, а внутри каждой группы столбиками отображаются значения элементов строк.

При обработке больших массивов векторных данных часто требуется получить информацию о том, какая часть данных находится в том или ином интервале. Функция hist предназначена для отображания гистограммы данных и нахождения числа данных в интервалах. Входным аргументом hist является вектор с данными, а выходным - вектор, содержащий количество элементов, попавших в каждый из интервалов. По умолчанию берется десять равных интервалов. Например, вызов hist(randn(1,5000)) приводит к появлению на экране гистограммы данных, распределенных по нормальному закону, а n=hist(randn(1,5000)) к заполнению вектора n длины десять (при этом гистограмма не строится). Число интервалов указывается во втором дополнительном аргументе hist. Можно задать интервалы, использовав в качестве второго аргумента не число, а вектор, содержащий центры интервалов. Более удобно задавать интервалы не центрами, а границами. В этом случае требуется сначала определить количество элементов в интервалах при помощи функции histc, а затем применить bar со специальным аргументом 'histc', например:

>> x=randn(1,10000);

>>int=[-2:0.5:2];

>> n=histc(x,int);

>>bar(int,n,'histc')

Каждое окно имеет свои оси, при наличии нескольких пар осей (в одном окне или в разных) вывод графиков производится в текущие оси. Последняя созданная пара осей является текущей. Для того, чтобы выбрать текущие оси из нескольких имеющихся, достаточно щелкнуть по ним левой кнопкой мыши перед вызовом графической функции. Возможна и обратная ситуация, когда в процессе работы требуется добавлять графики к уже имеющимся на некоторых осях. В этой ситуации перед добавлением графика следует выполнить команду holdon. Для завершения такого режима достаточно воспользоваться holdoff.

В одном графическом окне можно расположить несколько осей со своими графиками. Функция subplot предназначена для разбиения окна на части и определения текущей из них. Предположим, что требуется вывести графики на шесть пар осей в одно графическое окно (две по вертикали и три по горизонтали). Создайте графическое окно при помощи figure и выполните команду:

>>subplot(2,3,1)

В левом верхнем углу окна появились оси. Первые два аргумента в subplot указывают на общее число пар осей по вертикали и горизонтали, а последний аргумент означает номер данной пары осей. Нумерация идет слева направо, сверху вниз. Используйте subplot(2,3,2), : , subplot(2,3,6) для создания остальных пар осей. Вывод любой из графических функций можно направить в нужные оси, указав их при помощи subplot(2,3,k), например:

>>subplot(2,3,3)

>>bar([1.2 0.3 2.8 0.9])

>>subplot(2,3,6)

>>surf(X,Y,Z)

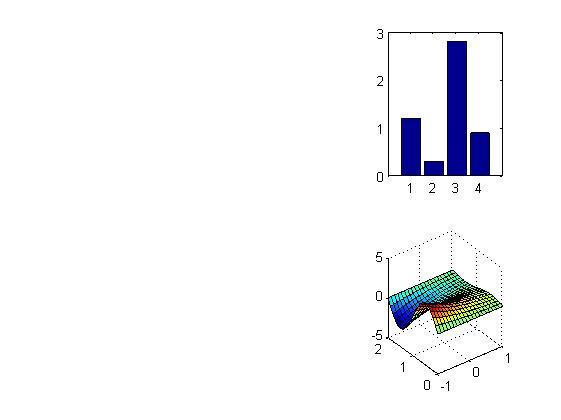


Рис. 2.6. Расположение нескольких графиков в одном окне