**Интерполяция и аппроксимация данных**

**Цель работы: изучение методов аппроксимации данных, получаемых в результате проведения экспериментов и исследования процессов, протекающих в различных физических подсистемах.**

Под аппроксимацией обычно подразумевается описание некоторой, порой не заданной явно, зависимости или совокупности представляющих ее данных с помощью другой, обычно более простой или более единообразной зависимости. Часто данные находятся в виде отдельных узловых точек, координаты которых задаются таблицей данных.

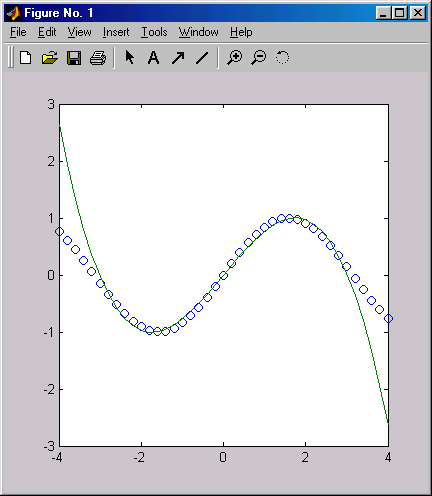
Результат аппроксимации может не проходить через узловые точки. Напротив, задача интерполяции — найти данные в окрестности узловых точек. Для этого используются подходящие функции, значения которых в узловых точках совпадают с координатами этих точек. Например, при *линейной интерполяции*зависимости *у(х)*узловые точки соединяются друг с другом отрезками прямых и считается, что искомые промежуточные точки расположены на этих отрезках.

Для повышения точности интерполяции применяют параболы (квадратичная интерполяция) или полиномы более высокой степени (полиномиальная интерполяция). Для обработки данных MATLAB использует различные функции интерполяции и аппроксимации данных. Набор таких функций вместе с несколькими вспомогательными функциями описан в этом разделе.

Полиномиальная регрессия

Одна из наиболее известных аппроксимаций — полиномиальная. В системе MATLAB определены функции аппроксимации данных полиномами по методу наименьших квадратов — полиномиальной регрессии. Это выполняет функция, приведенная ниже:

* polyfit(x.y.n) — возвращает вектор коэффициентов полинома р(х) степени п, который с наименьшей среднеквадратичной погрешностью аппроксимирует функцию у(х). Результатом является вектор-строка длиной n+1, содержащий коэффициенты полинома в порядке уменьшения степеней х и у равно n+1, то реализуется обычная полиномиальная аппроксимация, при которой график полинома точно проходит через узловые точки с координатами (х.у), хранящиеся в векторах х и у. В противном случае точного совпадения графика с узловыми точками не наблюдается;
* [p.S] = polyflt(x.y.n) — возвращает коэффициенты полинома р и структуру S для использования вместе с функцией polyval с целью оценивания или предсказания погрешности;
* [p.S] = polyf1t(x,y,n,mu) возвращает коэффициенты полинома р и структуру S для использования вместе с функцией polyval с целью оценивания или предска-зния погрешности, но так, что присходит центрирование (нормирование) и масштабирование х, xnorm = (х - mu(l))/mu(2), где mu(l) = mean(x) и mu(2) = std(x). Центрирование и масштабирование не только улучшают свойства степенного многочлена, получаемого при помощи polyval, но и значительно повышают качественные характеристики самого алгоритма аппроксимации.



***Рис. 17.10.****Пример использования функции polyfit*

Пример (полиномиальная регрессия для функции s

» х=(-3:0.2:3)':

y=sin(x);

p=polyflt(x.y,3)

р =

-0.0953 0.0000 0.8651 -0.0000

»x=(-4:0.2:4)';y=sin(x);

» f=polyval(p,x);plot(x.y.'o',x,f)

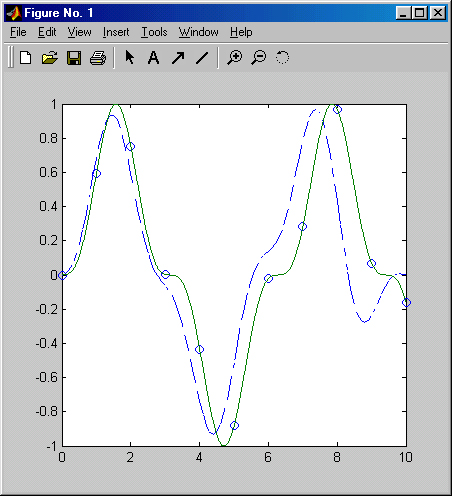
Рис. 17.14, построенный в этом примере, дает наглядное представление о точности полиномиальной аппроксимации. Следует помнить, что она достаточно точна в небольших окрестностях от точки *х =*0, но может иметь большие погрешности за их пределами или в промежутках между узловыми точками.

График аппроксимирующего полинома третьей степени на рис. 17.10 показан сплошной линией, а точки исходной зависимости обозначены кружками. К сожалению, при степени полинома свыше 5 погрешность полиномиальной регрессии (и аппроксимации) сильно возрастает и ее применение без центрирования и масштабирования становится рискованным. Обратите внимание на то, что при полиномиальной регрессии узловые точки не ложатся точно на график полинома, поскольку их приближение к нему является наилучшим в смысле минимального среднеквадратического отклонения. Об этом уже говорилось.

Интерполяция периодических функций рядом Фурье

Под интерполяцией обычно подразумевают вычисление значений функции *f(x)*в промежутках между узловыми точками. Линейная, квадратичная и полиномиальная интерполяция реализуются при полиномиальной аппроксимации. А вот для периодических (и особенно для гладких периодических) функций хорошие результаты может дать их интерполяция тригонометрическим рядом Фурье. Для этого используется следующая функция:

* interpft(x.n) — возвращает вектор у, содержащий значения периодической функции, определенные в п равномерно расположенных точках. Если length(x)=rr; и х имеет интервал дискретизации dx, то интервал дискретизации для у составляет dy=dx\*m/n, причем п не может быть меньше, чем т. Если X — матрица, interpft оперирует столбцами X, возвращая матрицу Y с таким же числом столбцов, как и у X, но с п строками. Функция y=interpft(x.n.dim) работает либо со строками, либо со столбцами в зависимости от значения параметра dim.



***Рис. 17.11.****Пример использования функции interpft*

Пример:

» x=0:10:y-sin(x).^3:

» xl-0:0.1:10;yl-interpft(y,101);

» x2=0:0.01:10:y2-sin(x2).^3;

» plot(xl,yl, '--').hold on.plotCx.y. 'o' .x2.y2)

Рис. 17.11 иллюстрирует эффективность данного вида интерполяции на примере функции sin(x).^3, которая представляет собой сильно искаженную синусоиду.

Исходная функция на рис. 17.12 представлена сплошной линией с кружками, а интерполирующая функция — штрих-пунктирной линией.

Интерполяция на неравномерной сетке

Для интерполяции на неравномерной сетке используется функция griddata:

* ZI = griddata(x.y.z.XI.YI) — преобразует поверхность вида z = f(x. у), которая определяется векторами (x.y.z) с (обычно) неравномерно распределенными элементами. Функция griddata аппроксимирует эту поверхность в точках, определенных векторами (XI.YI) в виде значений ZI. Поверхность всегда проходит через заданные точки. XI и YI обычно формируют однородную сетку (созданную с помощью функции meshgrid).

XI может быть вектором-строкой, в этом случае он определяет матрицу с постоянными столбцами. Точно так же YI может быть вектором-столбцом, тогда он определяет матрицу с постоянными строками.

* [XI.YI.ZI] = griddata(x,y,z,xi ,yi ) — возвращает аппроксимирующую матрицу ZI, как описано выше, а также возвращает матрицы XI и YI, сформированные из вектора-столбца xi и вектора-строки yi . Последние аналогичны матрицам, возвращаемым функцией meshgrid;
* [...] = griddata (....method) — использует определенный метод интерполяции:
  + 'nearest' — ступенчатая интерполяция;
  + 'linear' — линейная интерполяция (принята по умолчанию);
  + 'cubic' — кубическая интерполяция;
  + ' v4 ' — метод, используемый в МATLAB 4.

Метод определяет тип аппроксимирующей поверхности. Метод 'cubic1 формирует гладкие поверхности, в то время как 'linear' и 'nearest' имеют разрывы первых и нулевых производных соответственно. Все методы, за исключением v4, основаны на триангуляции Делоне. Метод ' v4 ' включен для обеспечения совместимости с версией 4 системы MATLAB. Пример:

» x=rand(120.1)\*4-2;y=rand(120.1)\*4-2;

z=x.\*y.\*exp(-x.^2-y.^2);

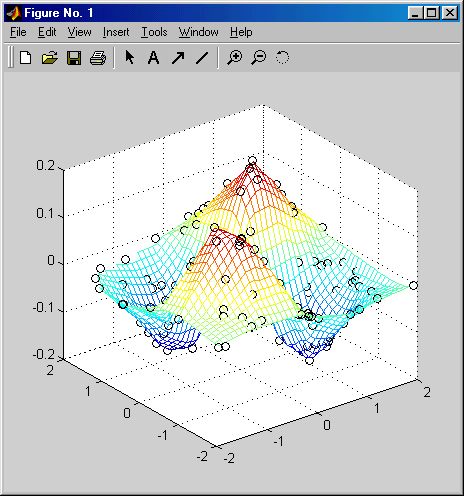
» t=-2:0.1:2:[X,Y]=meshgrid(t,t);

Z=griddata(x.y.z.X.Y):

» mesh(X.Y.Z).hold on.plot3(x.y,z, 'ok')

Функции griddataS и griddatan работают аналогично griddata, но для для трехмерного и n-мерного случая — с использованием алгоритма qhul 1 . Используются, в частности, при трехмерной и n-мерной триангуляции.

Рис. 17.13 иллюстрирует применение функции griddata.



***Рис 17.12.****Пример использования функции griddata*

Одномерная табличная интерполяция

В ряде случаев очень удобна сплайновая интерполяция и аппроксимация таблично заданных функций. При ней промежуточные точки ищутся по отрезкам полиномов третьей степени — это *кубическая сплайновая интерполяция.*При этом обычно такие полиномы вычисляются так, чтобы не только их значения совпадали с координатами узловых точек, но также чтобы в узловых точках были непрерывны производные первого и второго порядков. Такое поведение характерно для гибкой линейки, закрепленной в узловых точках, откуда и происходит название spline (сплайн) для этого вида интерполяции (аппроксимации). Для одномерной табличной интерполяции используется функция interpl:

* yi = Interpl(x.Y.xi) — возвращает вектор yi, содержащий элементы, соответствующие элементам xi и полученные интерполяцией векторов х и Y. Вектор х определяет точки, в которых задано значение Y. Если Y — матрица, то интерполяция выполняется для каждого столбца Y и у1 имеет длину length (xi) - by- size (Y. 2);
* yi = interpl (x.Y.xi .method) — позволяет с помощью параметра method задать метод интерполяции:
  + 'nearest' — ступенчатая интерполяция;
  + 'linear' — линейная интерполяция (принята по умолчанию);
  + 'spline' — кубическая сплайн-интерполяция;
  + 'cubic' или 'pchip' — интерполяция многочленами Эрмита;
  + 'v5cubic' — кубическая интерполяция MATLAB 5.
* yi = interpl (x.Y.xi .method, значение величин вне пределов изменения х) **—**позволяет отобразить особенные точки на графике;
* yi = i nterpl(х, Y, xi.method.' сообщение') — позволяет изменить сообщение об особенных точках на графике.

Все методы интерполяции требуют, чтобы значения х изменялись монотонно. Когда х — вектор равномерно распределенных точек, для более быстрой интерполяции лучше использовать методы '\*1inear', '\*cubic', '\*nearest' или '\*spline'. Обратите внимание, что в данном случае наименованию метода предшествует знак звездочки.

Пример (интерполяция функции косинуса):

» x=0:10:y=cos(x);

» xi=0:0.1:10:

» yi=interpl(x.y.xi);

» plot(x,y.'x',xi,yi,'g'),hold on

» yi=interpl(x,y.xi.'spline'):

» plot(x.y,'o',xi,yi,'m').grid,hold off

Узловые точки на рис. 17.17 обозначены кружками с наклонными крестиками. Одна из кривых соответствует линейной интерполяции, другая — сплайн-интерполяции. Нетрудно заметить, что сплайн-интерполяция в данном случае дает гораздо лучшие результаты, чем линейная интерполяция. При последней точки просто соединяются друг с другом отрезками прямых, так что график интерполирующей кривой при линейной интерполяции получается негладким.

Двумерная табличная интерполяция

Двумерная интерполяция существенно сложнее, чем одномерная, рассмотренная выше, хотя смысл ее тот же — найти промежуточные точки некоторой зависимости *z(x, у)*вблизи расположенных в пространстве узловых точек. Для двумерной табличной интерполяции используется функция interp2:

* ZI = interp2(X,Y.Z,XI.YI) — возвращает матрицу *ZI,*содержащую значения функции в точках, заданных аргументами XI и YI, полученные путем интерполяции двумерной зависимости, заданной матрицами X, Y и Z. При этом X и Y должны быть монотонными и иметь тот же формат, как если бы они были получены с помощью функции meshgrid (строки матрицы X являются идентичными; то же можно сказать о столбцах массива Y). Матрицы X и Y определяют точки, в которых задано значение Z. Параметры XI и YI могут быть матрицами, в этом случае interp2 возвращает значения Z, соответствующие точкам (XI(i,j),YI(i.j)). В качестве альтернативы можно передать в качестве параметров вектор-строку xi и вектор-столбец yi. В этом случае interp2 представляет эти векторы так, как если бы использовалась команда mesh-grid(xi .yi);
* ZI = interp2(Z,XJ.YI) — подразумевает, что Х=1:n и Y=l:m, где [m.n]=size(Z);
* ZI = interp2(Z,ntimes) — осуществляет интерполяцию рекурсивным методом с числом шагов ntimes;
* ZI = interp2(X,Y,Z.XI,YI.method) — позволяет с помощью опции method задать метод интерполяции:
  + 'nearest' — интерполяция по соседним точкам;
  + 'linear' — линейная интерполяция;
  + 'cubic' — кубическая интерполяция (полиномами Эрмита);
  + 'spline' — интерполяция сплайнами.

Все методы интерполяции требуют, чтобы X и Y изменялись монотонно и имели такой же формат, как если бы они были получены с помощью функции meshgrid. Когда X и Y — векторы равномерно распределенных точек, для более быстрой интерполяции лучше использовать методы '\*1inear', '\*cubic', или '\*nearest'.

Пример:

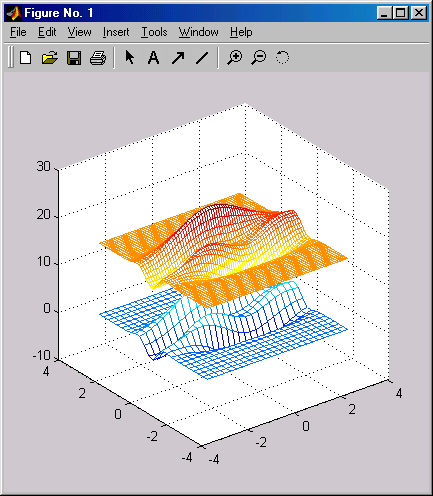
» [X.Y]=meshgrid(-3:0.25:3);

Z=peaks(X/2.Y\*2):

» [Xl,Yl]=meshgrid(-3:0.1:3);

Zl=interp2(X,Y.Z.Xl.Yl):

» mesh(X.Y,Z).hold on.mesh(Xl.Yl,Zl+15).hold off



***Рис. 17.13.****Применение функции interpZ*

Рис. 17.13 иллюстрирует применение функции interp2 для двумерной интерполяции (на примере функции peaks).

В данном случае поверхность снизу — двумерная линейная интерполяция, которая реализуется по умолчанию, когда не указан параметр method.

Трехмерная табличная интерполяция

Для трехмерной табличной интерполяции используется функция interp3:

* VI = interp3(X.Y.Z.V.XI,YI.ZI) — интерполирует, чтобы найти VI, значение основной трехмерной функции V в точках матриц XI, YI и ZI. Матрицы X, Y и Z определяют точки, в которых задано значение V. XI, YI и ZI могут быть матрицами, в этом случае InterpS возвращает значения Z, соответствующие точкам (XI (i ,j) ,YI(i. j), ZI (i. j)). В качестве альтернативы можно передать векторы xi, yl и zi. Векторы аргументы, имеющие неодинаковый размер, представляются, как если бы использовалась команда meshgrid;
* VI = interp3(V.XI.YI.ZI) - подразумевает X=1:N, Y=1:M, Z=1:P, где [M,N.P]=size(V);
* VI = interpS(V.ntimes) — осуществляет интерполяцию рекурсивным методом с числом шагов ntimes;
* VI = interp3(... .method) — позволяет задать метод интерполяции:
  + 'nearest' — ступенчатая интерполяция;
  + 'linear' — линейная интерполяция;
  + 'cubic' — кубическая интерполяция (полиномами Эрмита);
  + 'spline' — интерполяция сплайнами.

Все методы интерполяции требуют, чтобы X, Y и Z изменялись монотонно и имели такой же формат, как если бы они были получены с помощью функции meshgrid. Когда X и Y и Z — векторы равномерно распределенных в пространстве узловых точек, для более быстрой интерполяции лучше использовать методы '\*li'near', '\*cubic' или '\*nearest'.

N-мерная табличная интерполяция

MATLAB позволяет выполнить даже n-мерную табличную интерполяцию. Для этого используется функция interpn:

* VI = interpn(Xl.X2,X3,... ,V,Y1.Y2.Y3....)- интерполирует, чтобы найти VI, значение основной многомерной функции V в точках массивов Yl, Y2, Y3,.... Функции interpn должно передаваться 2ЛГ+1 аргументов, где *N —*размерность интерполируемой функции. Массивы XI, Х2, ХЗ,... определяют точки, в которых задано значение V. Параметры Yl, Y2, Y3,... могут быть матрицами, в этом случае interpn возвращает значения VI, соответствующие точкам (YKi, j) ,Y2(i, j), Y3(i, j),...). В качестве альтернативы можно передать векторы yl, y2, уЗ,... В этом случае interpn интерпретирует их, как если бы использовалась команда ndgrid(yl. У2.у3....);
* VI = interpn(V.Yl,Y2,Y3,...) - подразумевает Xl=1size(V.l), X2=l:size(V,2), X3=l:size(V,3) и т. д.;
* VI = Interpn(V.ntimes) — осуществляет интерполяцию рекурсивным методом с числом шагов ntimes;
* VI = interpn(...method) — позволяет указать метод интерполяции:
* 'nearest' — ступенчатая интерполяция;
* 'linear' — линейная интерполяция;
* 'cubic' — кубическая интерполяция.

В связи с редким применением такого вида интерполяции, наглядная трактовка которой отсутствует, примеры ее использования не приводятся.

Интерполяция кубическим сплайном

Сплайн-интерполяция используется для представления данных отрезками полиномов невысокой степени — чаще всего третьей. При этом кубическая интерполяция обеспечивает непрерывность первой и второй производных результата интерполяции в узловых точках. Из этого вытекают следующие свойства кубической сплайн-интерполяции:

* график кусочно-полиномиальной аппроксимирующей функции проходит точно через узловые точки;
* в узловых точках нет разрывов и резких перегибов функции;
* благодаря низкой степени полиномов погрешность между узловыми точками обычно достаточно мала;
* связь между числом узловых точек и степенью полинома отсутствует;
* поскольку используется множество полиномов, появляется возможность аппроксимации функций с множеством пиков и впадин.

Как отмечалось, в переводе spline означает «гибкая линейка». График интерполирующей функции при этом виде интерполяции можно уподобить кривой, по которой изгибается гибкая линейка, закрепленная в узловых точках. Реализуется сплайн-интерполяция следующей функцией:

* yi = spline(x,y,xi) — использует векторы х и у, содержащие аргументы функции и ее значения, и вектор xi, задающий новые точки; для нахождения элементов вектора yi используется кубическая сплайн-интерполяция;
* рр = spline(x.y) — возвращает рр-форму сплайна, используемую в функции ppval и других сплайн-функциях.

Пример:

» х=0:10: y=3\*cos(x);

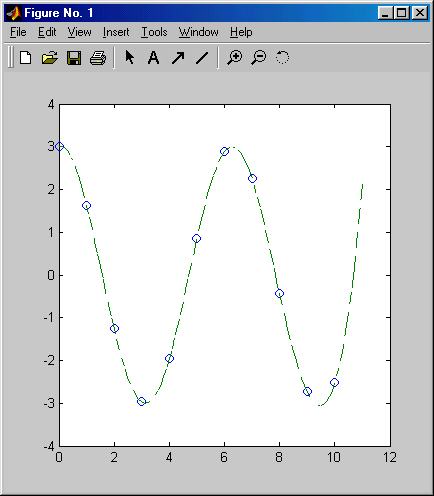
» xl=0:0.1:11;

» y1=spline(x,y.xl);

» plot(x.y,'о'.xl.yl.'--')

Сплайн-интерполяция дает неплохие результаты для функций, не имеющих разрывов и резких перегибов. Особенно хорошие результаты получаются для монотонных функций.

Результат интерполяции показан на рис. 17.14.



***Рис. 17.14.****Пример применения функции spline*

Ввиду важности сплайн-интерполяции и аппроксимации в обработке и представлении сложных данных в состав системы MATLAB входит пакет расширения Spline Toolbox, содержащий около 70 дополнительных функций, относящихся к реализации сплайн-интерполяции и аппроксимации, а также графического представления сплайнами их результатов. Для вызова данных об этом пакете (если он установлен) используйте команду help splines.

**Обработка данных в графическом окне**

Средства обработки данных в графическом окне

Решение большинства задач интерполяции и аппроксимации функций и табличных данных обычно сопровождается их визуализацией. Она, как правило, заключается в построении узловых точек функции (или табличных данных) и в построении функции аппроксимации или интерполяции. Для простых видов аппроксимации, например полиномиальной, желательно нанесение на график формулы, полученной для аппроксимации.

В MATLAB 6,0 совмещение функций аппроксимации с графической визуализацией доведено до логического конца — предусмотрена аппроксимация рядом методов точек функции, график которой построен. И все это выполняется прямо в окне редактора графики Property Editor. Для этого в позиции Tools графического окна имеются две новые команды:

* Basic Fitting - основные виды аппроксимации (регрессии);
* Data Statistics - статистические параметры данных.

Команда Basic Fitting открывает окно, дающее доступ к ряду видов аппроксима-- ции и регрессии: сплайновой, эрмитовой и полиномиальной со степенями от 1 (линейная аппроксимация) до 10. В том числе со степенью 2 (квадратичная аппроксимация) и 3 (кубическая аппроксимация). Команда Data Statistics открывает окно с результатами простейшей статистической обработки данных.

Полиномиальная регрессия для табличных данных

Рассмотрим самый характерный пример обработки данных, примерно представляющих некоторую (например, экспериментальную) зависимость вида у(х). Пусть она задана в табличной форме, причем колонки таблицы соответствуют элементам векторов X и Y одинакового размера в следующем примере:

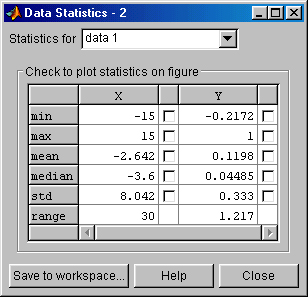
» Х=[2.4,6.8.10.12.14];

» Y=[3.76.4.4.5.1.5.56.6.6.3.6.7];

» plot(X.Y.'o');

Напомним, что последняя команда строит график узловых точек кружками (без соединения их отрезками прямых).

Рис. 17.15 показывает пример выполнения полиномиальной регрессии (аппроксимации) для степеней полинома 1, 2 и 3 (дальнейшее повышение степени полинома в данном случае уже лишено смысла, поскольку графики полиномиальной регрессии со степенью выше 3 почти не различаются).



***Рис. 17.15.****Пример обработки табличных данных в графическом окне*

Поясним, что же показано на рис. 17.20. В левом верхнем углу сессии MATLAB видна запись приведенных выше исходных векторов и команды построения заданных ими точек кружками. Справа показано большое окно графики с построенными в нем кружками узловыми точками.

http://www.phys.nsu.ru/cherk/matlab6/Chapter%2017/Prim.GIF*При проведении полиномиальной аппроксимации надо помнить, что максимальная степень полинома на 1 меньше числа точек, т. е. числа элементов в векторах X и Y.*

Исполнив команду Tools *>*Basic Fitting, можно получить окно регрессии. Оно показано на рис. 17.20 слева прямо под записью исходных команд в командной строке. В этом окне птичкой отмечены три вида полиномиальной регрессии — порядка 1 (linear — линейная), 2 (quadratic — квадратичная) и 3 (cubic — кубическая). Стоит отметить какой-либо вид регрессии, как соответствующая кривая функции регрессии (аппроксимации) появится в графическом окне.

Установив птичку у параметра Show equations (Показать уравнения), можно получить в графическом окне запись уравнений регрессии (аппроксимации). Наконец, можно сместить выводимую по умолчанию легенду в место, где она не закрывала бы другие детали графика.

Наконец, исполнив команду Tools *>*Data Statistics, можно получить окно *с*рядом статистических параметров данных, представленных векторами X и Y. Отметив птичкой тот или иной параметр в этом окне (оно показано на рис. 17.20 под окном графики), можно наблюдать соответствующие построения на графике, например вертикалей с минимальным, средним и максимальных значением у и горизонталей с минимальным, средним и максимальным значением х.

http://www.phys.nsu.ru/cherk/matlab6/Chapter%2017/Prim.GIF*Безусловно, эта новинка понравится большинству пользователей системы MATLAB 6.0. Однако нельзя не отметить, что статистические данные более чем скупы.*

Оценка погрешности аппроксимации

Средства обработки данных из графического окна позволяют строить столбцовый или линейчатый график погрешностей в узловых точках и наносить на эти графики норму погрешности. Норма дает статистическую оценку среднеквадрати-ческой погрешности. Чем она меньше, тем точнее аппроксимация. Для вывода графика погрешности надо установить птичку у параметра Plot residuals (График погрешностей) и в меню ниже этого параметра выбрать тип графика.

Таким образом, интерфейс графического окна позволяет выполнять эффективную обработку данных наиболее распространенными способами.

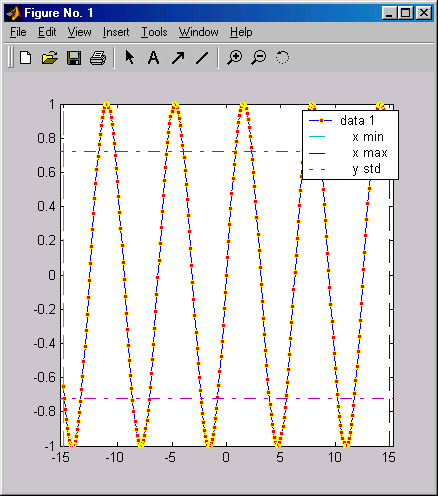
Сплайновая интерполяция в графическом окне

Попытка аппроксимации полиномом 8-й степени не дает положительного результата — кривая проходит внутри облака точек, совершенно не интерполируя это облако.

Однако если применить сплайновую интерполяцию, то картина кардинально меняется. На этот раз кусочная линия интерполяции прекрасно проходит через все точки и поразительно напоминает синусоиду. Даже ее пики со значениями 1 и -1 воспроизводятся удивительно точно, причем и в случаях, когда на них не попадают узловые точки.

Причина столь великолепного результата кроется в уже отмеченных ранее особенностях сплайновой интерполяции - она выполняется по трем ближайшим

точкам, причем эти тройки точек постепенно перемещаются от начала точечного графика функции к ее концу. Кроме того, непрерывность первой и второй производных при сплайновой интерполяции делает кривую очень плавной, что характерно и для первичной функции — синусоиды. Так что данный пример просто является удачным случаем применения сплайновой интерполяции.



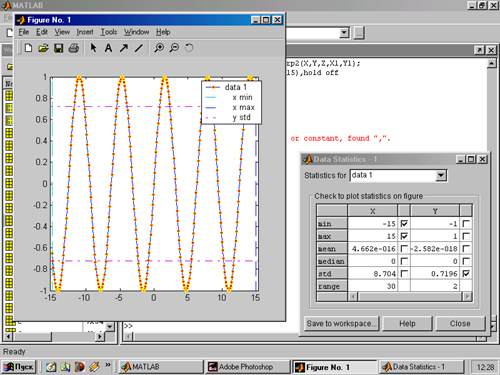
***Рис. 17.16.****Пример сплайновой интерполяции в графическом окне*

Мы не можем практически называть этот подход полноценной аппроксимацией, поскольку в данном случае нет единого выражения для аппроксимирующей функции. На каждом отрезке приближения используется кубический полином с новыми коэффициентами. Поэтому и вывода аппроксимирующей функции в поле графика не предусмотрено.

Эрмитовая многоинтервальная интерполяция

MATLAB 6.0 дает возможность в графическом окне использовать еще один вид многоинтервальной интерполяции на основе полиномов третьей степени Эрмита. Техника интерполяции здесь таже, что и в случае сплайновой интерполяции, (рис. 17.17).

Полиномы Эрмита имеют более гибкие линии, чем сплайны. Они точнее следуют за отдельными изгибами исходной зависимости. Это хорошо показывает рис. 17.17.



***Рис. 17.17.****Пример эрмитовой интерполяции синусойды в графическом окне*

Сравнение сплайновой и эрмитовой интерполяции

Оба вида интерполяции в данном случае дают превосходные результаты, поскольку представляемая ими кусочная функция практически почти точно проходит через все заданные точки. Однако если учесть, что эти точки принадлежат синусоиде, то в данном случае результаты сплайновой интерполяции оказываются явно лучшими. Особенно это характерно для экстремальных точек.

Поскольку в этих двух методах интерполяции кривая интерполяции проходит точно через узловые точки, в этих точках погрешности интерполяции равны нулю. Вы можете проверить это задав вывод графика погрешности. В целом, можно заключить, что сплайновая интерполяция лучше, когда нужно эффективное сглаживание быстро меняющихся от точки к точке данных и когда исходная зависимость описывается линиями, которые мы наблюдаем при построении их с помощью гибкой линейки. Эрмитова интерполяция лучше отслеживает быстрые изменения исходных данных, но имеет худшие сглаживающие свойства.

Все это говорит о том, что надо внимательно подходить к оценке приемлемости того или иного вида интерполяции (или аппроксимации) для конкретных типов исходных данных.

**Ход работы:**

* 1. Получить у преподавателя функцию, моделирующую процесс.
  2. Выбрать один период функции, моделирующей процесс.
  3. Разбить выделенный фрагмент (период исследуемой функции) на 9 интервалов и получить координаты точек, имитирующие результаты эксперимента.
  4. Исследовать возможности получения моделей процесса различными методами аппроксимации на массивах координат точек эксперимента.
  5. Провести сравнительный анализ методов аппроксимации.
  6. Расширить исследуемый диапазон и проверить точность аппроксимации моделями, полученными на шаге 4.
  7. Отчет должен содержать: графики аппроксимаций экспериментальных данных, оценки погрешностей внутри и вне диапазона, оценку сложности полученных моделей.