Математическая обработка результатов эксперимента

Изучая теорию интерполяции, вы познакомились с интерполяционными формулами, которые в точности воспроизводят значения данной функции в узлах интерполяции. Однако в ряде случаев выполнение этого условия затруднительно или даже нецелесообразно:

1. Если заданные величины x и y являются экспериментальными данными, то могут содержать в себе существенные ошибки, т. к. получены в результате измерений или наблюдений. Поэтому построение аппроксимирующего многочлена, воспроизводящего в точности заданное значение функции, означало бы тщательное копирование допущенных при измерениях ошибок.
2. Если имеются точные значения функции в некоторых точках, но число таких точек n весьма велико, то интерполяционный многочлен будет очень высокой степени[5].
3. Между узлами интерполяции значения интерполяционного многочлена могут сильно отклоняться от значений интерполируемой функции.

***Пример* 2.** Известный из физики закон охлаждения состоит в том, что *скорость охлаждения тела пропорциональна избытку температуры тела над температурой среды*, т. е. выражается формулой

$$\upsilon = a\,\vartheta,$$

где υ – скорость охлаждения, ϑ - избыток температуры и a – коэффициент пропорциональности (закон Ньютона).

В Таблице 9 приведены опытные данные, выражающие зависимость между υ и ϑ для некоторого определенного тела, по которым можно определить значения коэффициента пропорциональности a.

Таблица 9

$\vartheta°$	220	200	180	160	140	120	100
υ	8,81	7,40	6,10	4,89	3,88	3,02	2,30

По существу, при отсутствии ошибок измерения для определения значения a достаточно иметь лишь одно наблюдение. Однако на самом деле это не так, и, как показывает Таблица 2, каждое из наблюдений дает новое значение

---

[5] Если только разности не будут становиться постоянными.

a. Например, первое из наблюдений дает $\vartheta = 220$, $\upsilon = 8{,}81$, откуда, следует $a = 0{,}040$. Аналогично следующие наблюдения дают

$a = 0{,}037;\ 0{,}034;\ 0{,}031;\ 0{,}028;\ 0{,}025;\ 0{,}023.$

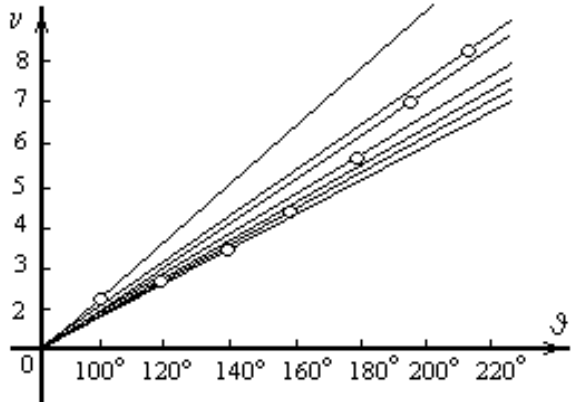

Рисунок 17. Опытные данные Примера 2

Смотря по тому, каким из наблюдений мы пользуемся, зависимость между скоростью охлаждения и избытком температуры будет иметь тот или иной коэффициент пропорциональности и будет изображаться соответствующей прямой (Рисунок 17).

Конечно, есть возможность по имеющимся семи наблюдениям, построить интерполяционный многочлен шестой степени, который будет в точности совпадать с наблюденными значениями. Но это совершенно не нужно, так как все эти значения содержат ошибки. Требуется по имеющимся наблюденным данным построить линейную функцию, которая для различных (в том числе и имеющихся в таблице) значений ϑ давала бы наиболее близкие значения υ.

Таким образом, возникает задача построения многочлена некоторой вполне определенной степени, но меньшей чем n - 1, который хотя и не дает точных значений функции в узлах интерполяции, но достаточно близко к ним подходит.

Метод наименьших квадратов

В Примере 2 рассматривалась зависимость вида $\upsilon = a\,\vartheta$, причем значения υ и ϑ получаются из наблюдений. Если бы измерения величины υ и ϑ производились без ошибок, то для определения коэффициента a было бы достаточно одного измерения. Если рассматривать более общую зависимость, например, $\upsilon = a\vartheta + b$, то здесь имеется два неизвестных

коэффициента, для нахождения которых достаточно было бы двух абсолютно точных измерений.

На самом деле, абсолютно точные измерения чаще всего невозможны. Для того, чтобы исключить влияние ошибок, производится большое число измерений. Каждое измерение дает нам уравнение, связывающее неизвестные коэффициенты. При большом числе измерений мы приходим, следовательно, к системе, число уравнений в которой значительно больше, нежели число неизвестных. Здесь ставится задача отыскания наиболее вероятных значений коэффициентов, которые, вообще говоря, не будут точно удовлетворять ни одному из уравнений системы. Эту задачу можно сформулировать в более общем виде. Пусть дана функция

$$y = f(x; a_0, a_1, ..., a_m) \tag{10}$$

независимой переменной x и $m + 1$ параметра $a_0, a_1, ..., a_m$. Эти параметры постоянны, но заранее неизвестны и подлежат определению. Для их отыскания производится ряд измерений величин x и y[6]. Подставляя их в равенство (10), мы получаем уравнения между параметрами $a_0, a_1, ..., a_m$, вида

$$y_i = f(x_i, a_0, a_1, ..., a_m) \ (i = 1, 2, \ldots, n), \tag{11}$$

где x_i и y_i — соответствующие друг другу измеренные значения, а n — число измерений. Если бы значения x и y находились точно, то для отыскания $m + 1$ параметра достаточно было бы произвести $m + 1$ измерение.

На самом деле, значения x и y содержат ошибки, и никакие $m + 1$ измерений не позволят определить истинные значения параметров. Поэтому обычно производится большее число измерений ($n > m + 1$), в результате чего число уравнений (11) будет больше, чем число неизвестных параметров. В этом случае система (11) будет несовместной, т. е. точные решения каких-либо ($m + 1$) из уравнений системы могут не удовлетворять остальным уравнениям[7]. Требуется найти *наиболее вероятные значения неизвестных параметров.* Эти вероятные значения будут тем более близки к истинным, чем больше число наблюдений.

Так как уравнения (11) удовлетворяются неточно, то будем иметь

$$y_i - f(x_i, a_0, a_1, ..., a_m) = \varepsilon_i \ (i = 1, 2, \ldots, n), \tag{12}$$

---

[6] Разумеется, так поступают тогда, когда константы $a_0, a_1, ..., a_m$ не поддаются непосредственным измерениям, а величины x и y измерению доступны.

[7] В случае $n < m + 1$ система (2) была бы совместной и всегда имела бы бесчисленное множество решений.

где ε_i — отклонения измеренных значений y_i от вычисленных по формуле (10). Принцип наименьших квадратов утверждает, что *наивероятнейшими значениями параметров будут такие, при которых сумма квадратов отклонений* ε_i, *будет наименьшей*[8], т. е.

$$S = \sum_{i=0}^{n} \varepsilon_i^2 = \sum_{i=0}^{n} [y_i - f(x_i, a_0, a_1, ..., a_m)]^2 = \min. \tag{13}$$

Функцию S называют *функцией невязки*.

Аппроксимация в виде линейной комбинации функций

Часто функцию $f(x)$ выбирают в виде линейной комбинации подходящих функций

$$f(x) = a_0 f_0(x) + a_1 f_1(x) + \ldots + a_m f_m(x). \tag{14}$$

Условие минимума S определяется уравнениями

$$\frac{\partial S}{\partial a_0} = 0, \ \frac{\partial S}{\partial a_1} = 0, \ ..., \ \frac{\partial S}{\partial a_m} = 0. \tag{15}$$

Поскольку

$$S = \sum_{i=0}^{n} [y_i - (a_0 f_0(x) + a_1 f_1(x) + \ldots + a_m f_m(x))]^2, \tag{16}$$

это условие эквивалентно системе уравнений:

$$\begin{cases} \dfrac{\partial S}{\partial a_0} = 2 \sum\limits_{i=0}^{n} [y_i - (a_0 f_0(x) + a_1 f_1(x) + \ldots + a_m f_m(x))] f_0(x) = 0, \\ \dfrac{\partial S}{\partial a_1} = 2 \sum\limits_{i=0}^{n} [y_i - (a_0 f_0(x) + a_1 f_1(x) + \ldots + a_m f_m(x))] f_1(x) = 0, \\ \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \\ \dfrac{\partial S}{\partial a_m} = 2 \sum\limits_{i=0}^{n} [y_i - (a_0 f_0(x) + a_1 f_1(x) + \ldots + a_m f_m(x))] f_m(x) = 0. \end{cases}$$

Эти $m + 1$ уравнений, очевидно, можно представить в виде

---

[8] При этом мы считаем, что отклонения ε_i подчиняются нормальному закону распределения.

$$\begin{bmatrix} \sum_{i=0}^{n} f_0^2(x_i) & \sum_{i=0}^{n} f_0(x_i)f_1(x_i) & \dots & \sum_{i=0}^{n} f_0(x_i)f_m(x_i) \\ \dots & \dots & \dots & \dots \\ \sum_{i=0}^{n} f_0(x_i)f_m(x_i) & \dots & \dots & \sum_{i=0}^{n} f_m^2(x_i) \end{bmatrix} \begin{bmatrix} a_0 \\ \vdots \\ a_m \end{bmatrix} =$$

$$= \begin{bmatrix} \sum_{i=0}^{n} f_0(x_i)y_i \\ \vdots \\ \sum_{i=0}^{n} f_m(x_i)y_i \end{bmatrix} \qquad (17)$$

Так как элементы матрицы в левой части и вектора-столбца в правой определяются табличными данными, то выписанная система m + 1 линейных уравнений с m +1 неизвестными может быть решена. Можно выбрать любую функцию $f(x)$, лишь бы она была линейной относительно своих коэффициентов.

Если предполагается, что данные могли бы быть смоделированы в виде линейной комбинации произвольных функций (14), следует использовать функцию *linfit*, чтобы вычислить коэффициенты a_i. Это так называемая аппроксимация *линейной комбинацией функций* (Рисунок 18).

linfit(*vx, vy, F***)**

Возвращает вектор, содержащий коэффициенты, чтобы создать линейную комбинацию функций из *F*, дающую наилучшую аппроксимацию данных из векторов *vx* и *vy*. *F* – функция, которая возвращает вектор, состоящий из функций, которые нужно объединить в виде линейной комбинации.

Примеры применения способа наименьших квадратов

Занимаясь интерполированием функций, мы рассматривали, как говорят, *точечную интерполяцию,* т. е. строили интерполяционные многочлены, значения которых точно совпадали со значениями самой функции в узлах интерполяции. Однако было указано, что возможна и другая постановка задачи, когда от интерполяционного многочлена требуется лишь, чтобы он как-то приближал заданную функцию и не обязательно совпадал с ней в отдельных заданных точках. Говоря о

заданной функции, мы будем подразумевать при этом, что нам известны лишь отдельные ее значения в некоторых точках[9].

Мы предполагаем, следовательно, что зависимость y от x имеет вид многочлена

$$y = a_0 + a_1x + \ldots + a_mx^m. \qquad (18)$$

с неизвестными коэффициентами.

Нашей задачей является нахождение по результатам наблюдений наиболее вероятных значений коэффициентов $a_0, a_1, \ldots, a_m$.

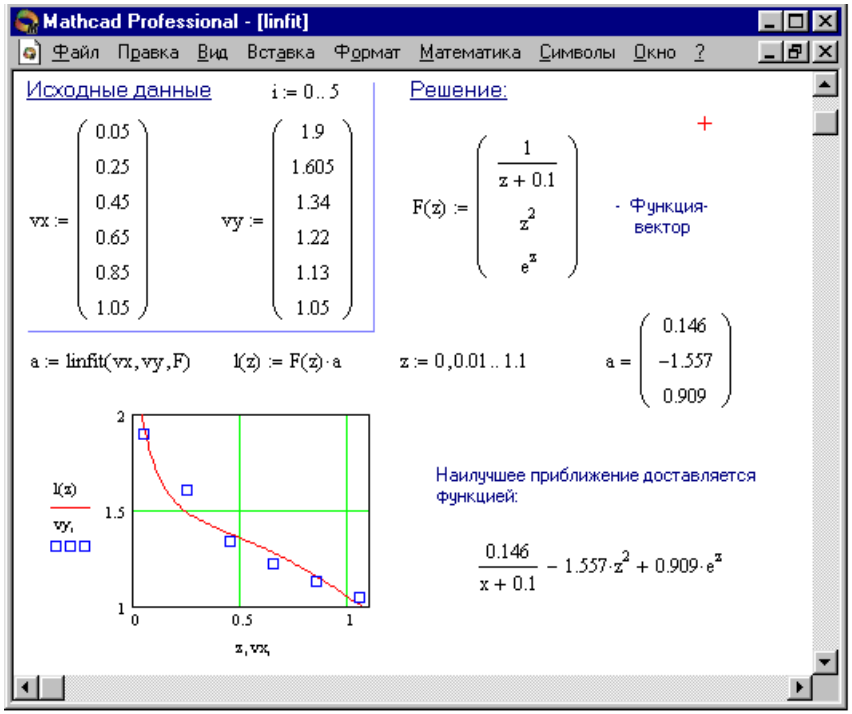

Рисунок 18. Использование функции *linfit* для нахождения коэффициентов в линейной комбинации функций

[9] Можно говорить и о приближении функций, заданных аналитически.

Если бы число наблюдений в точности равнялось числу неизвестных коэффициентов, то мы имели бы дело с задачей интерполяции. Значительно более важным является тот случай, когда число наблюдений n много больше степени многочлена. В этом случае получается обычная задача способа наименьших квадратов[10]. Требуется найти коэффициенты $a_0, a_1, ..., a_m,$ дающие минимум функции невязки

$$S = \sum_{i=0}^{n} [y_i - (a_0 + a_1 x + ... + a_m x^m)]^2. \qquad (19)$$

Условие минимума S определяется уравнениями

$$\frac{\partial S}{\partial a_0} = 0, \frac{\partial S}{\partial a_1} = 0, ..., \frac{\partial S}{\partial a_m} = 0. \qquad (20)$$

Условие (19) с учетом (20) эквивалентно системе уравнений

$$\begin{cases} \dfrac{\partial S}{\partial a_0} = 2 \sum\limits_{i=0}^{n} [y_i - (a_0 + a_1 x_i + ... + a_m x_i^m)], \\ \dfrac{\partial S}{\partial a_1} = 2 \sum\limits_{i=0}^{n} [y_i - (a_0 + a_1 x_i + ... + a_m x_i^m)] x_i, \\ . \ . \ . \ . \ . \ . \ . \ . \ . \ . \ . \ . \ . \ . \ . \\ \dfrac{\partial S}{\partial a_m} = 2 \sum\limits_{i=0}^{n} [y_i - (a_0 + a_1 x_i + ... + a_m x_i^m)] x_i^m. \end{cases}$$

Эти $m + 1$ уравнений можно представить в виде:

[10] Нет смысла применять метод наименьших квадратов при $n \le m + 1$. В этом случае система уравнений для определения коэффициентов a_i совместна и сумма квадратов будет равна нулю при любых решениях системы.

$$\begin{cases} (n+1)a_0 + a_1\sum_{i=0}^{n} x_i + a_2\sum_{i=0}^{n} x_i^2 + \ldots + a_m\sum_{i=0}^{n} x_i^m = \sum_{i=0}^{n} y_i \ , \\ a_0\sum_{i=0}^{n} x_i + a_1\sum_{i=0}^{n} x_i^2 + a_2\sum_{i=0}^{n} x_i^3 + \ldots + a_m\sum_{i=0}^{n} x_i^{m+1} = \sum_{i=0}^{n} x_i y_i \ , \\ \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \\ a_0\sum_{i=0}^{n} x_i^m + a_1\sum_{i=0}^{n} x_i^{m+1} + a_2\sum_{i=0}^{n} x_i^{m+2} + \ldots + a_m\sum_{i=0}^{n} x_i^{2m} = \sum_{i=0}^{n} x_i^m y_i \ . \end{cases} \quad (21)$$

Решая систему уравнений (21), находим коэффициенты $a_0, a_1, \ldots, a_m$, которые являются искомыми параметрами многочлена (18).

Систему (21) можно записать компактно:

$$\begin{cases} b_{00}a_0 + b_{01}a_1 + \ldots + b_{0m}a_m = c_0, \\ b_{10}a_0 + b_{11}a_1 + \ldots + b_{1m}a_m = c_1, \\ \ldots\ldots \\ b_{m0}a_0 + b_{m1}a_1 + \ldots + b_{mm}a_m = c_m. \end{cases} \quad (22)$$

где $b_{k,\,l} = \sum_{i=0}^{n} x_i^{k+l}, \quad c_k = \sum_{i=0}^{n} x_i^k y_i, \quad k, l = 0, 1, \ldots, m$. (23)

Используйте *regress*, когда нужно использовать единственный полином, чтобы приблизить все данные. Функция *regress* допускает использование полинома любого порядка. Однако на практике не следует использовать степень полинома выше $n = 4$ (Рисунок 19).

Так как *regress* пытается приблизить все точки данных, используя один полином, это не даст хороший результат, когда данные не связаны единой полиномиальной зависимостью. Например, предположим, ожидается, что y_i зависят линейно от x в диапазоне от x_1 до x_{10} и ведут себя подобно кубическому полиному в диапазоне от x_{11} до x_{20}. Если используется *regress* с $n = 3$, можно получить хорошее приближение для второй половины, но плохое — для первой. Функция *loess* облегчает эти проблемы, выполняя локальное приближение. Вместо создания одного полинома, как это делает *regress*, *loess* создаёт различные полиномы второго порядка в зависимости от расположения на кривой.

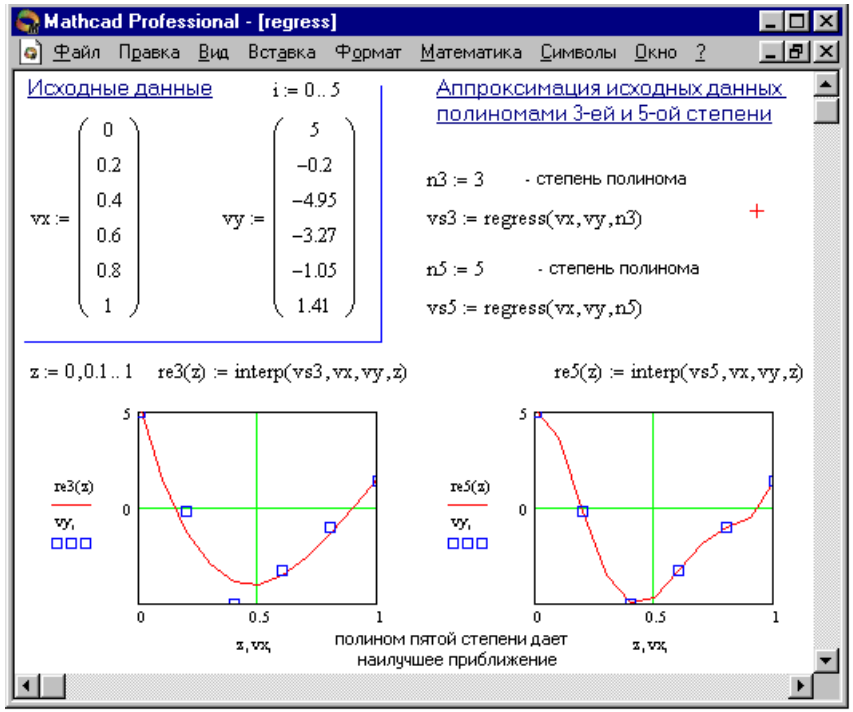

Рисунок 19. Аппроксимация полиномами различной степени с помощью функции *regress*

Она делает это, исследуя данные в малой окрестности точки, представляющей интерес. Аргумент *span* управляет размером этой окрестности. По мере того, как диапазон становится большим, *loess* становится эквивалентным *regress* с $n = 2$. Значение по умолчанию — *span* = 0,75.

Рисунок 20 показывает, как *span* влияет на приближение, выполненное функцией *loess*. Заметьте, что меньшее значение *span* лучше приближает флуктуации данных. Большее значение *span* сглаживает колебания данных и создает более гладкую приближающую функцию.

regress(*vx*, *vy*, *n***)**

Возвращает вектор, требуемый *interp*, чтобы найти полином порядка *n*, который наилучшим образом приближает данные из *vx* и *vy*. *vx* есть *m*-мерный вектор, содержащий координаты *x*. *vy* есть *m*-мерный вектор,

содержащий координаты *y,* соответствующие m точкам, определенным в *vx*.

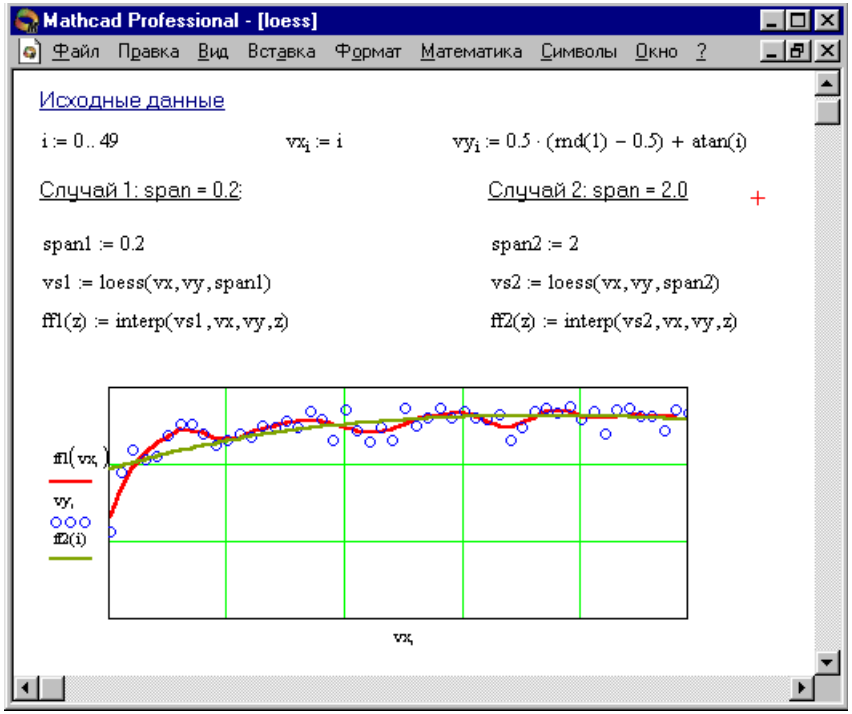

Рисунок 20. Влияние различных значений *span* на функцию *loess*

loess(*vx, vy, span*)

Возвращает вектор, требуемый *interp,* чтобы найти набор полиномов второго порядка, которые наилучшим образом приближают определённые окрестности выборочных точек, определенных в векторах *vx* и *vy*. *vx* есть *m*-мерный вектор, содержащий координаты *x*. *vy* есть *m*-мерный вектор, содержащий координаты *y,* соответствующие *m* точкам, определенным в *vx***.** Аргумент *span (span* > 0) определяет, насколько большие окрестности будет использовать *loess* при выполнении локального приближения.

interp(*vs, vx, vy, x*)

Возвращает интерполируемое значение *y,* соответствующее *x.* Вектор vs вычисляется *loess* или *regress* на основе данных из *vx* и *vy*.

Аппроксимирующие функции в Mathcad

В пакете Mathcad представлен широкий набор функций для представления наблюденных данных уравнениями, некоторые из которых уже были рассмотрены в предыдущих разделах. В таблице 10 приведен список функций.

Таблица 10

Аппроксимирующие функции в Mathcad

Функция	**Вид аппроксимируемой зависимости**	**Уравнение**
linfit(*vx,vy, F***)**	линейная комбинация функций	$a_1 f_1(x) + \ldots + a_2 f_2(x) + a_m f_m(x)$
regress(*vx, vy, n***)**	Полином степени *n*	$a_0 + a_1 x + \ldots + a_n x^n$
loess(*vx, vy, span***)**	Набор полиномов второй степени	
slope(*vx, vy***)** **intercept(***vx, vy***)**	линейная	$a + b\,x$
line(*vx, vy***)**	линейная	$a + b\,x$
medfit(*vx, vy***)**	линейная	$a + b\,x$
pwrfit(*vx, vy, vg***)**	степенная	$a\,x^{\,b} + c$
expfit(*vx, vy, vg***)**	показательная	$a\,\mathrm{e}^{\,bx} + c$
logfit(*vx, vy, vg***)**	логарифмическая	$a\,\ln(x + b) + c$
sinfit(*vx, vy, vg***)**	синусоидальная	$a\,\sin(x - b) + c$
lgfit(*vx, vy, vg***)**	логистическая	$\frac{a}{1 + b\,e^{-cx}}$
genfit(*vx, vy, vg, F***)**	нелинейная	

Особый интерес представляет собой функция *genfit*, использование которой возможно для аппроксимации в общем случае любой нелинейной зависимости. Все, что можно делать с помощью любой из

функций из Таблицы 3, можно также делать, хотя в ряде случаев и менее удобно, с помощью *genfit.*

Однако имеются случаи, когда гибкость функций, представляющих ту или иную математическую зависимость, недостаточна. Например, если данные могут быть смоделированы в виде суммы

$$f(x) = a_1 \sin(2x) + a_2 \operatorname{tg}(3x)$$

и все, что нужно сделать – решить уравнение относительно неизвестных коэффициентов a_1 и a_2, значит эта проблема решается с помощью *linfit*.

В противоположность этому, если данные должны быть смоделированы в виде суммы

$$f(x) = 2 \sin(a_1 x) + 3 \operatorname{tg}(a_2 x)$$

и требуется найти неизвестные параметры a_1 и a_2, то это задача для функции *genfit*.

Таким образом, если данные должны быть смоделированы в виде

$$f(x) = f(x, u_0, u_1, \ldots, u_n),$$

нужно использовать функцию *genfit*, чтобы найти неизвестные параметры u_i.

genfit(*vx, vy, vg, F***)**

Возвращает вектор, содержащий n параметров $u_0, u_1, \ldots, u_{n-1}$, которые обеспечивают наилучшее приближение данных из *vx* и *vy* функцией *f*, зависящей от x и параметров $u_0, u_1, \ldots, u_{n-1}$.

Аргументы:

vx - вектор значений вещественных данных x;

vy - вектор значений вещественных данных y;

F – функция, которая возвращает $n + 1$-мерный вектор, содержащий *f* и ее частные производные относительно параметров. Для нахождения частных производных необходимо использовать команду символьных преобразований **Символы ⇒ Переменные ⇒ Дифференциалы**;

vg - n-мерный вектор начальных значений для n параметров.

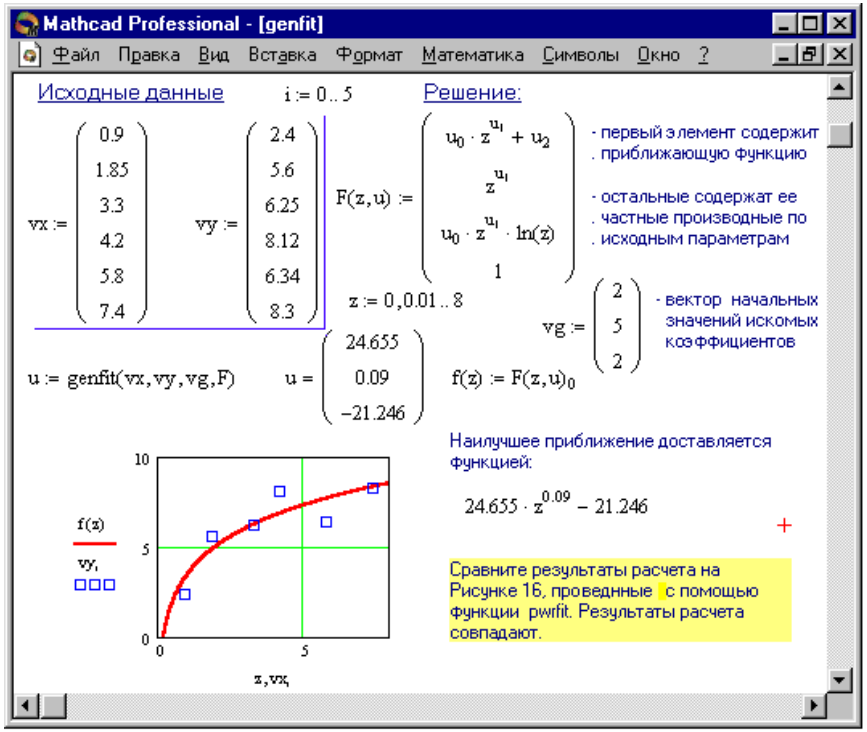

Рисунок 21. Использование функции *genfit*

На Рисунке 21 приведен пример использования функции *genfit* для определения коэффициентов степенной функции. Причем результаты расчета, полученные с использованием функции *genfit,* полностью совпадают с результатами использования функции *pwrfit*.

Функции Mathcad для линейной аппроксимации

Встроенные функции *intercept* (to intercept - отложить отрезок на линии) и *slope* (наклон) решают самую простую и распространенную задачу *линейной аппроксимации* экспериментальных данных:

$$f(x) = slope(vx, vy)\, x + intercept(vx, vy)$$

slope (vx, vy)

Возвращает скаляр: наклон линии для данных из векторов *vx* и *vy*.

intercept(*vx, vy*)

Возвращает скаляр: смещение по оси ординат линии для данных из векторов *vx* и *vy*.

Рисунок 22 показывает, как можно использовать эти функции, чтобы провести линию через набор данных.

В последних версиях Mathcad появились две новые функция для определения коэффициентов прямой линии, аппроксимирующей исходный набор точек.

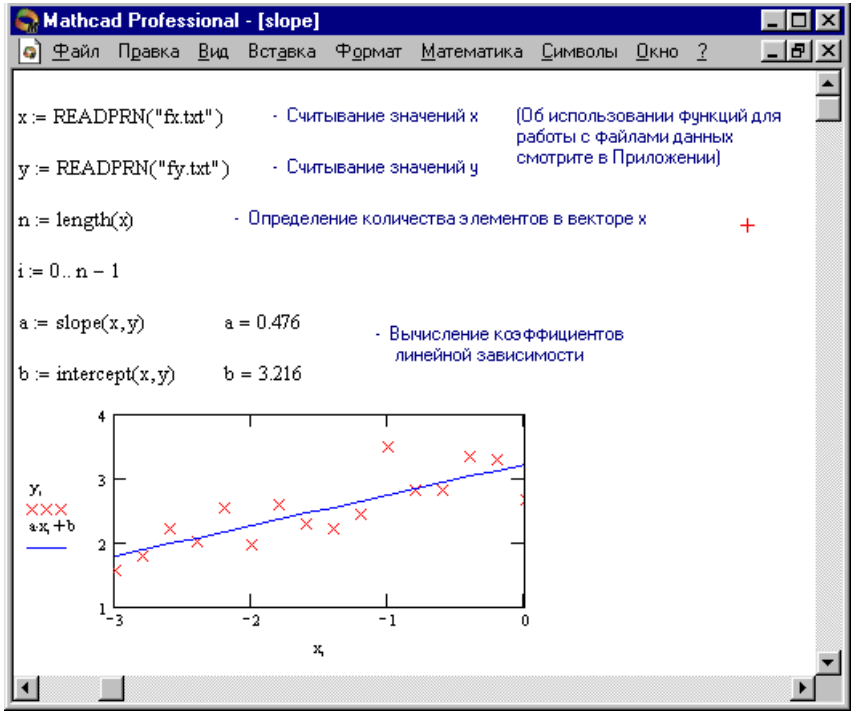

Рисунок 22. Использование функций slope и intercept

line(*vx, vy*), medfit(*vx, vy*)

Определяют вектор, содержащий коэффициенты линейного уравнения $a + b\,x$.

Рисунок 10 показывает применение функций *line* и *medfit*. А также приведено сравнение результатов вычислений по точности. Как показано

на Рисунке 10, функция *line* дает более точное приближение, нежели функция *medfit.*

Нахождение коэффициентов для степенных функций

Формулу вида $y = a + b\,x^{m}$ (24)

с заданным показателем степени *m* можно привести к линейному виду, построив на оси *Ox* функциональную шкалу для функции $X = x^{m}$.

Функция Mathcad *pwrfit* служит для нахождения коэффициентов для степенных функций.

pwrfit(*vx, vy, vg*)

Определяет вектор, содержащий коэффициенты для кривой показателя степени вида $a\,x^{b} + c$.

Аргументы:

vx - вектор значений вещественных данных *x*;
vy - вектор значений вещественных данных *y*;
vg - вектор параметров *a, b* и *c* в показательном уравнении.

Векторы *vx* и *vy* должны иметь одно и то же число элементов. Вектор оценок *vg* необходим для инициализации. Уменьшение значения встроенной системной переменной TOL может увеличивать точность аппроксимациис использованием функции *pwrfit*.

На Рисунке 23 поясняется использование функции *pwrfit***.**

Подбор коэффициентов для показательных функций

Из наиболее часто встречающихся на практике зависимостей осталось рассмотреть зависимость вида

$$y = a\,e^{bx} \qquad (25)$$

Функция Mathcad *expfit* служит для нахождения коэффициентов для степенных функций.

expfit (*vx, vy, vg*)

Определяет вектор, содержащий коэффициенты для кривой показателя степени вида $a\,e^{bx} + c$.

Аргументы:

vx - вектор значений вещественных данных *x*;
vy - вектор значений вещественных данных *y*;
vg - вектор параметров *a, b* и *c* в показательном уравнении.

Векторы *vx* и *vy* должны иметь одно и то же число элементов. Вектор оценок *vg* необходим для инициализации. Уменьшение значения

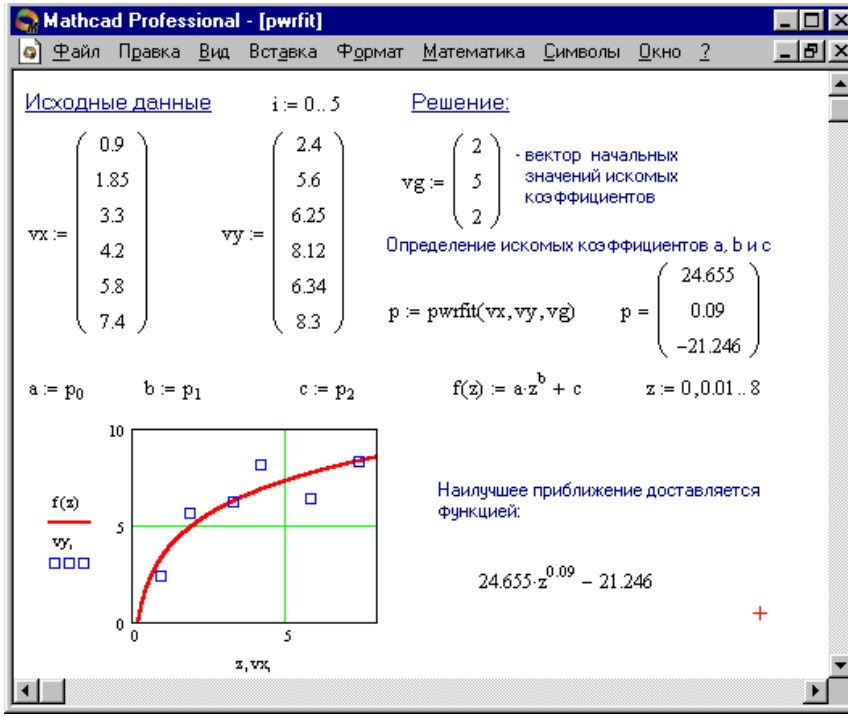

Рисунок 23. Аппроксимация данных с использованием функции prwfit

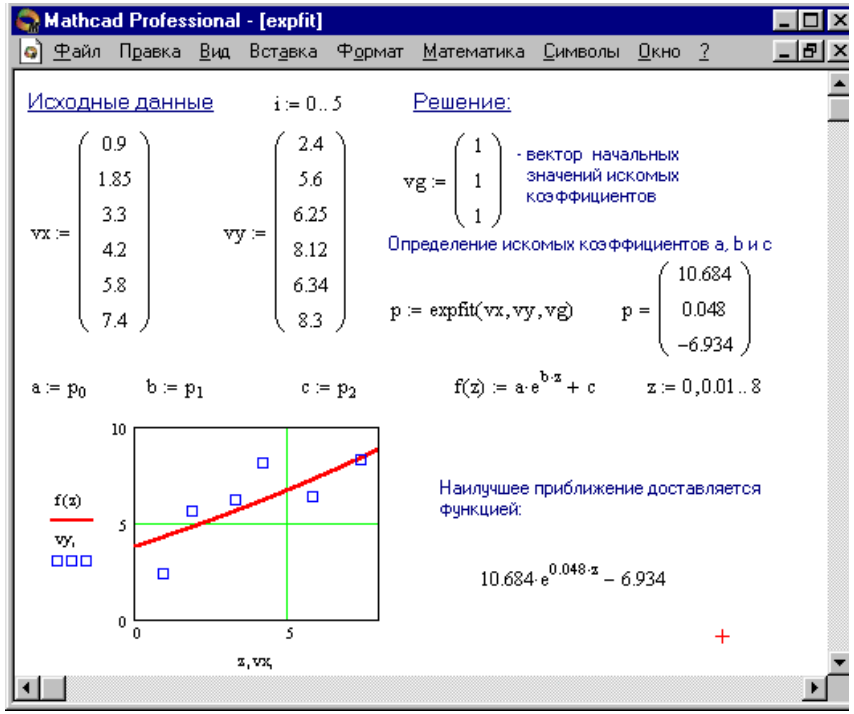

Рисунок 24. Аппроксимация данных с использованием функции expfit

встроенной системной переменной TOL может увеличивать точность аппроксимации с использованием функции *expfit*.

На Рисунке 24 поясняется использование функции *expfit***.**

Сглаживание данных эксперимента

Сглаживание предполагает использование набора значений *y* (и, возможно, *x*) и возвращение нового набора значений *y*, который является более гладким, чем исходный набор. В отличие от аппроксимации, сглаживание приводит к новому набору значений *y*, а не к функции, которая может оценивать значения между заданными точками данных.

Функция *supsmooth* использует симметричную линейную процедуру сглаживания Методом наименьших квадратов по правилу *k*-ближайших точек, чтобы выполнить локальную линейную аппроксимацию исходных данных. *supsmooth* адаптивно выбирает различную ширину полосы сглаживания для различных частей данных.

supsmooth(*vx, vy*)

Возвращает *n*-мерный вектор, созданный локальным использованием симметричной линейной процедуры сглаживания методом наименьших квадратов по правилу *k*-ближайших точек, в которой *k* выбирается адаптивно.

Аргументы:

vx,vy – *n*-мерные векторы вещественных чисел. Элементы *vx* должны быть расположены в порядке возрастания.

Рисунок 25 показывает пример использования функции *supsmooth* для сглаживания экспериментальных данных.

В Mathcad имеется еще 2 функции для сглаживания экспериментальных данных. Сглаживание с помощью функции *ksmooth* полезно, когда данные взяты в точках, отделяемых друг от друга интервалами примерно равной ширины.

ksmooth(*vx, vy, b*)

Возвращает *n*-мерный вектор, созданный сглаживанием при помощи гауссова ядра данных из вектора *vy*.

Аргументы:

vx,vy – *n*-мерные векторы вещественных чисел;

b - параметр, управляет окном сглаживания и должен быть в несколько раз больше величины интервала между точками *x*.

Функция *medsmooth* наиболее устойчивая из трех, так как в меньшей степени подвержена влиянию зашумленных данных.

medsmooth(*vy, m*)

Возвращает *n*-мерный вектор, созданный сглаживанием вектора *vy* с помощью скользящей медианы.

Аргументы:

vx,vy – *n*-мерные векторы вещественных чисел;

m - ширина окна, по которому происходит сглаживание, причем *m* должно быть нечетным числом и $m < n$.

Порядок выполнения работы

Упражнение 1. Из файлов данных **C:\Variant\vx*.txt** и **C:\Variant\vy*.txt** (где * - номер варианта), используя функцию *READPRN*, сформировать векторы x_i и y_i соответственно. С помощью функции *length* определить число элементов *n* в векторах *x* и *y*.

Упражнение 2. Аппроксимировать многочленами 1-ой, 2-ой и 5-ой степеней по *методу наименьших квадратов* функцию, заданную таблицей значений x_i и y_i. Сравнить качество приближений, вычислив сумму квадратов отклонений для каждого случая. Построить графики многочленов и отметить узловые точки (x_i, y_i).

Указание: составить нормальную систему метода наименьших квадратов и решить ее с помощью встроенной функции *lsolve*.

Упражнение 3. Для сформированных данных (x_i, y_i) определить параметры *линейной зависимости* с использованием встроенных функций Mathcad:

- *slope* и *intercept*;
- *line;*
- *medfit.*

Отобразить графически совокупность точек векторов x_i и y_i и результаты проведенной аппроксимации.

Упражнение 4. Аппроксимировать данные из векторов x_i и y_i

- полиномами 2 и 5-ой степени при помощи функций *regress* и *interp*;
- наборами полиномов второго порядка с помощью функций *loess* и *interp*, (при *span* равном 0,6 и 2,0).

Отобразить графически результаты аппроксимации.

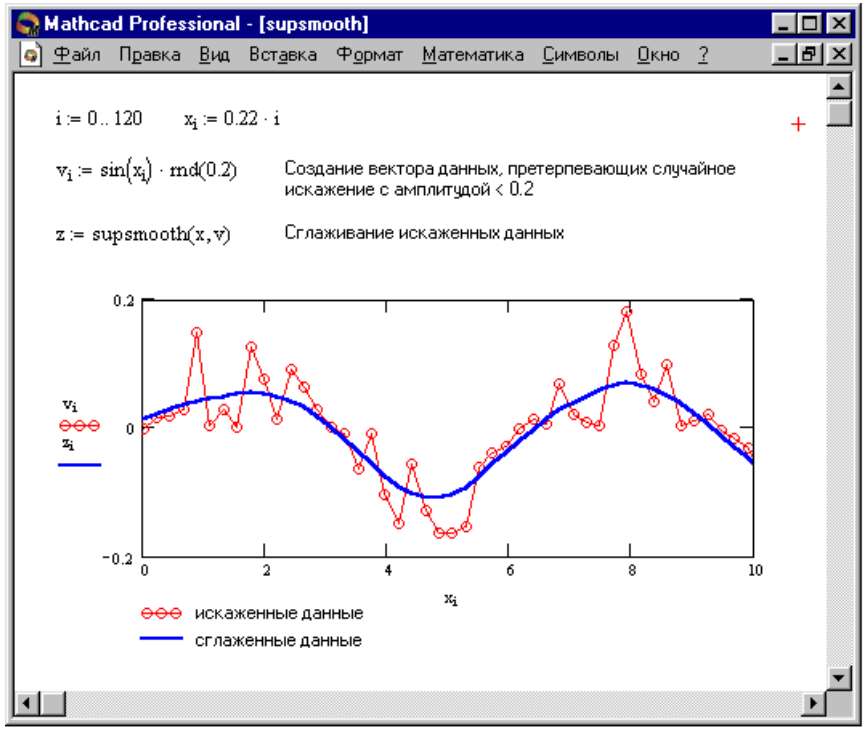

Рисунок 25. Сглаживание зашумленных данных с помощью функции supsmooth

Упражнение 5. Аппроксимировать экспериментальные данные из таблиц значений x_i и y_i линейной комбинацией функций:

$$f(x) = a_1 f_1(x) + a_2 f_2(x) + a_3 f_3(x).$$

Коэффициенты вектора a найти с помощью функции *linfit*. Отобразить графически совокупность точек векторов x_i и y_i и результаты проведенной *аппроксимации.*

Таблица 11

Варианты упражнения 5

№ вари-анта	$f_1(x)$	$f_2(x)$	$f_3(x)$
1	e^x	$1/\sqrt{1+2\cos^2 x}$	$\sin x$
2	$1/(1+x^2)$	e^x	$\sin(3x)$
3	$1/(1+x^2)$	$e^{\sin x}$	x
4	$\operatorname{arctg} x$	$\ln\ln x$	$\sin x$
5	$e^{-x^2/2}$	$1/x$	e^{-x}
6	$(1+x)/(2+x)$	$\cos(x/10)$	$\cos x$
7	$1/(1+e^{x^2})$	$\sqrt{1+x^2}$	$\cos x$
8	$\cos(x/2)$	$2-\cos x$	$\sin(x/2)$
9	$1/(1+e^x)$	$\operatorname{arctg}\sqrt{x}$	$\sin(3x)$
10	$\ln(x+5)$	$\sqrt{1+x}$	$\sin x$
11	$1/x$	$\sqrt{1+x}$	$1/x^2$
12	$\cos x$	$1/(1+x+x^2)$	$1/(1+x)$
13	e^x	$\cos 4x$	$-e^{x/2}$
14	$\sqrt{1+e^{-x}}$	$e^{x/3}$	$\sin^2(3x)$
15	$1/(1+x+x^2)$	$\cos(x/10)$	$\cos(x/10)$

Упражнение 6. Аппроксимировать экспериментальные данные из таблиц значений x_i и y_i функцией вида:

$$f(x) = e^{u_0 + u_1 \cdot x + u_2 \cdot x^2}.$$

Параметры вектора *u* найти с помощью функции *genfit*. Отобразить графически совокупность точек векторов x_i и y_i и результаты проведенной *аппроксимации.*

Упражнение 7. Зависимость между величинами *x* и *y* (Таблица 12) описывается функцией *y* = *f*(*x, a, b*), где *a* и *b* – неизвестные параметры. Найти эти параметры с помощью функций *pwrfit* или *expfit* в зависимости от варианта.

Таблица 12

Варианты упражнения 7

№ вари-анта	*y* = *f*(*x, a, b*)	Величины *x* и *y*	
1	$a\,x^{b}$	*x*	0.1; 0.3; 0.4; 0.7; 0.9; 0.95
		y	0.01; 0.13; 0.32; 1.71; 3.65; 4.29
2	$a\,e^{bx}$	*x*	0.1; 0.4; 0.9; 1.2; 1.6; 2
		y	2.54; 5.22; 17.34; 35.63; 93.05; 243.02
3	$a\,x^{b}$	*x*	0.1; 0.2; 0.4; 0.5; 0.8; 1
		y	1; 1.52; 2.31; 2.64; 3.5; 4
4	$a\,e^{bx}$	*x*	1; 1.6; 1.9; 2.2; 2.6; 3
		y	17.93; 44.09; 69.15; 108.45; 197.61; 360.07
5	$a\,x^{b}$	*x*	0.1; 0.5; 0.9; 1.5; 1.6; 2
		y	0.0005; 0.06; 0.36; 1.69; 2.05; 4
6	$a\,e^{bx}$	*x*	1; 1.2; 1.4; 1.5; 1.8; 2
		y	0.9; 0.77; 0.65; 0.6; 0.47; 0.4
7	$a\,x^{b}$	*x*	0.1; 0.2; 0.25; 0.5; 0.9; 1
		y	6.32; 4.47; 4; 2.83; 2.11; 2
8	$a\,e^{bx}$	*x*	0.1; 0.6; 0.9; 1.2; 1.5; 2
		y	3.12; 3.81; 4.3; 4.85; 5.47; 6.68
9	$a\,x^{b}$	*x*	0.1; 0.4; 0.9; 1.2; 1.6; 2
		y	0.01; 0.22; 1.55; 3.1; 6.18; 10.56
10	$a\,e^{bx}$	*x*	0.1; 0.6; 0.9; 1.2; 1.5; 2
		y	2.71; 1.65; 1.22; 0.9; 0.67; 0.41

11	$a\,x^{b}$	x	1; 1.6; 1.9; 2.2; 2.6; 3
		y	4; 8.1; 10.48; 13.05; 16.77; 20.78
12	$a\,e^{bx}$	x	0.1; 0.2; 0.25; 0.5; 0.9; 1
		y	1.9; 1.81; 1.76; 1.56; 1.28; 1.21
13	$a\,x^{b}$	x	0.1; 0.6; 0.9; 1.2; 1.5; 2
		y	1.19; 2.45; 2.88; 3.23; 3.53; 3.96
14	$a\,e^{bx}$	x	0.1; 0.2; 0.4; 0.5; 0.8; 1
		y	4.25; 4.51; 5.08; 5.4; 6.46; 7.29
15	$a\,x^{b}$	x	1; 1.2; 1.4; 1.5; 1.8; 2
		y	2; 1.73; 1.53; 1.45; 1.25; 1.15

Упражнение 8. Используя функцию равномерного распределения *rnd,* создать вектор данных y, претерпевающих случайное искажение с амплитудой $0.1v$:

$$y_i := \cos(x_i)\ \mathrm{rnd}\ (0.1\ v)$$

где v - номер варианта, $i := 0..50$, $x_i := 0.1i$. Выполнить сглаживание экспериментальной функции, заданной таблицей значений x_i и y_i, с помощью встроенной функции Mathcad *supsmooth.* Результаты сглаживания отобразить графически.

Упражнение 9. Выполнить сглаживание экспериментальной функции, заданной таблицей значений x_i и y_i (см. Упражнение 8), используя встроенные функции Mathcad: *medsmooth* и *ksmooth***.** Результаты сглаживания отобразить графически.

Контрольные вопросы

1. В каких случаях применение интерполяции нецелесообразно?
2. В чем состоит принцип наименьших квадратов?
3. Влияет ли число наблюденных данных на число параметров в аппроксимируемой функции?
4. С помощью какой функции в Mathcad возможна аппроксимация в виде линейной комбинации функций?
5. Каким образом можно определить вид аппроксимирующей функции?
6. Назовите функции полиномиальной аппроксимации в Mathcad.
7. Назовите функции Mathcad для линейной аппроксимации.
8. Назовите особенности определения коэффициентов для степенных функций.
9. Назовите особенности определения коэффициентов для показательных функций.