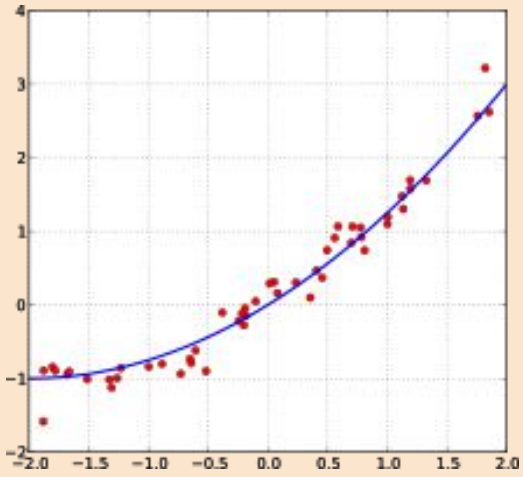
Линейная аппроксимация



Линейная аппроксимация

При обработке экспериментальных данных часто возникает необходимость аппроксимировать их линейной функцией.

Аппроксимацией (приближением) функции f(x) называется нахождение такой функции (***аппроксимирующей функции***) g(x), которая была бы близка заданной. Критерии близости функций могут быть различные.

В случае если приближение строится на дискретном наборе точек, аппроксимацию называют ***точечной*** или ***дискретной***.

В случае если аппроксимация проводится на непрерывном множестве точек (отрезке), аппроксимация называется ***непрерывной*** или ***интегральной***. Примером такой аппроксимации может служить разложение функции в ряд Тейлора, то есть замена некоторой функции степенным многочленом.

Наиболее часто встречающим видом точечной аппроксимации является ***интерполяция*** – нахождение промежуточных значений величины по имеющемуся дискретному набору известных значений.

Пусть задан дискретный набор точек, называемых ***узлами интерполяции***, а также значения функции в этих точках. Требуется построить функцию g(x), проходящую наиболее близко ко всем заданным узлам. Таким образом, критерием близости функции является g(xi)=yi.

Найдя интерполяционный полином, мы можем вычислить значения функции между узлами, а также определить значение функции даже за пределами заданного интервала (провести ***экстраполяцию***).

Аппроксимация линейной функцией

Любая линейная функция может быть записана уравнением $y = a \cdot x + b$

Аппроксимация заключается в отыскании коэффициентов a и b уравнения таких, чтобы все экспериментальные точки лежали наиболее близко к аппроксимирующей прямой.

С этой целью чаще всего используется метод наименьших квадратов (МНК), суть которого заключается в следующем: сумма квадратов отклонений значения точки от аппроксимирующей точки принимает минимальное значение:

$$F(a,b) = \sum_{i=1}^{n} (y_i - (a \cdot x_i + b))^2 \rightarrow min$$

Решение поставленной задачи сводится к нахождению экстремума указанной функции двух переменных. С этой целью находим частные производные функции функции по коэффициентам a и b и приравниваем их к нулю.

$$\begin{cases} \dfrac{\partial F(a,b)}{\partial a} = -2\sum_{i=1}^{n}\left(y_i - (a\cdot x_i + b)\right)\cdot x_i = 0 \\ \dfrac{\partial F(a,b)}{\partial b} = -2\sum_{i=1}^{n}\left(y_i - (a\cdot x_i + b)\right) = 0 \end{cases}$$

Решаем полученную систему уравнений

$$\begin{cases} a \cdot \sum_{i=1}^{n} x_i^2 + b \cdot \sum_{i=1}^{n} x_i = \sum_{i=1}^{n} x_i \cdot y_i \\ a \cdot \sum_{i=1}^{n} x_i + \sum_{i=1}^{n} b = \sum_{i=1}^{n} y_i \end{cases} \Rightarrow \begin{cases} a \cdot \sum_{i=1}^{n} x_i^2 + b \cdot \sum_{i=1}^{n} x_i = \sum_{i=1}^{n} x_i \cdot y_i \\ a \cdot \sum_{i=1}^{n} x_i + n \cdot b = \sum_{i=1}^{n} y_i \end{cases}$$

Определяем значения коэффициентов

$$\begin{cases} a = \dfrac{n \cdot \sum_{i=1}^{n} x_i \cdot y_i - \sum_{i=1}^{n} x_i \cdot \sum_{i=1}^{n} y_i}{n \cdot \sum_{i=1}^{n} x_i^2 - \left(\sum_{i=1}^{n} x_i\right)^2} \\ b = \dfrac{\sum_{i=1}^{n} y_i - a \cdot \sum_{i=1}^{n} x_i}{n} \end{cases}$$

Для вычисления коэффициентов необходимо найти следующие составляющие:

$$sumx = \sum_{i=1}^{n} x_i$$

$$sumx = \sum_{i=1}^{n} x_i$$

$$sumy = \sum_{i=1}^{n} y_i$$

$$sumx2 = \sum_{i=1}^{n} x_i^2$$

$$sumxy = \sum_{i=1}^{n} x_i \cdot y_i$$

Тогда значения коэффициентов будут определены как

$$\begin{cases} a = \dfrac{n \cdot sumxy - sumx \cdot sumy}{n \cdot sumx2 - sumx^2} \\ b = \dfrac{sumy - a \cdot sumx}{n} \end{cases}$$

Реализация

Для примера реализации воспользуемся набором значений, полученных в соответствии с уравнением прямой

y = 8 · x - 3

Рассчитаем указанные коэффициенты по методу наименьших квадратов.

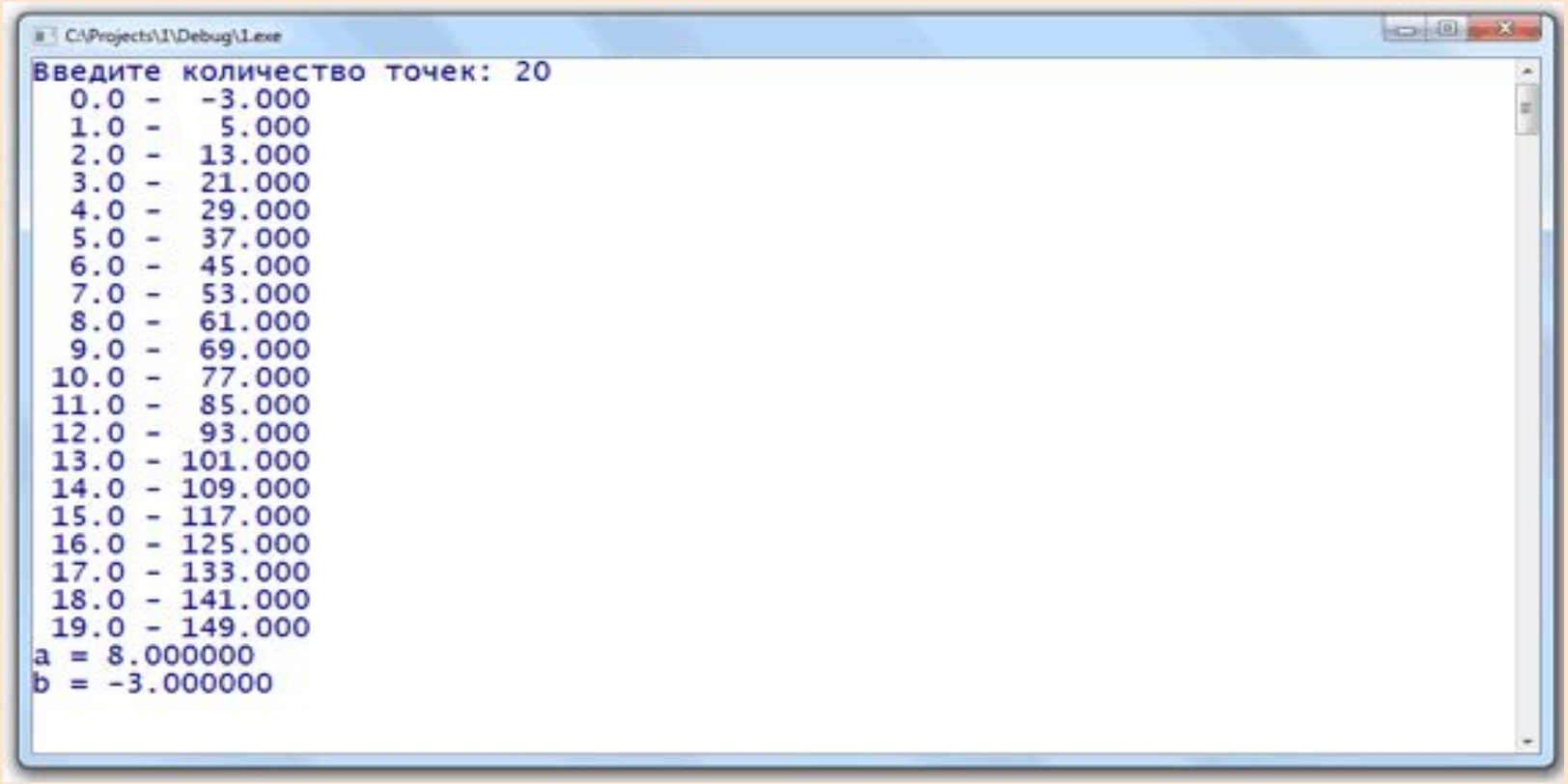
Результат сохраняем в форме двумерного массива, состоящего из 2 столбцов.

При следующем запуске программы добавим случайную составляющую к указанному набору значений и снова рассчитаем коэффициенты.

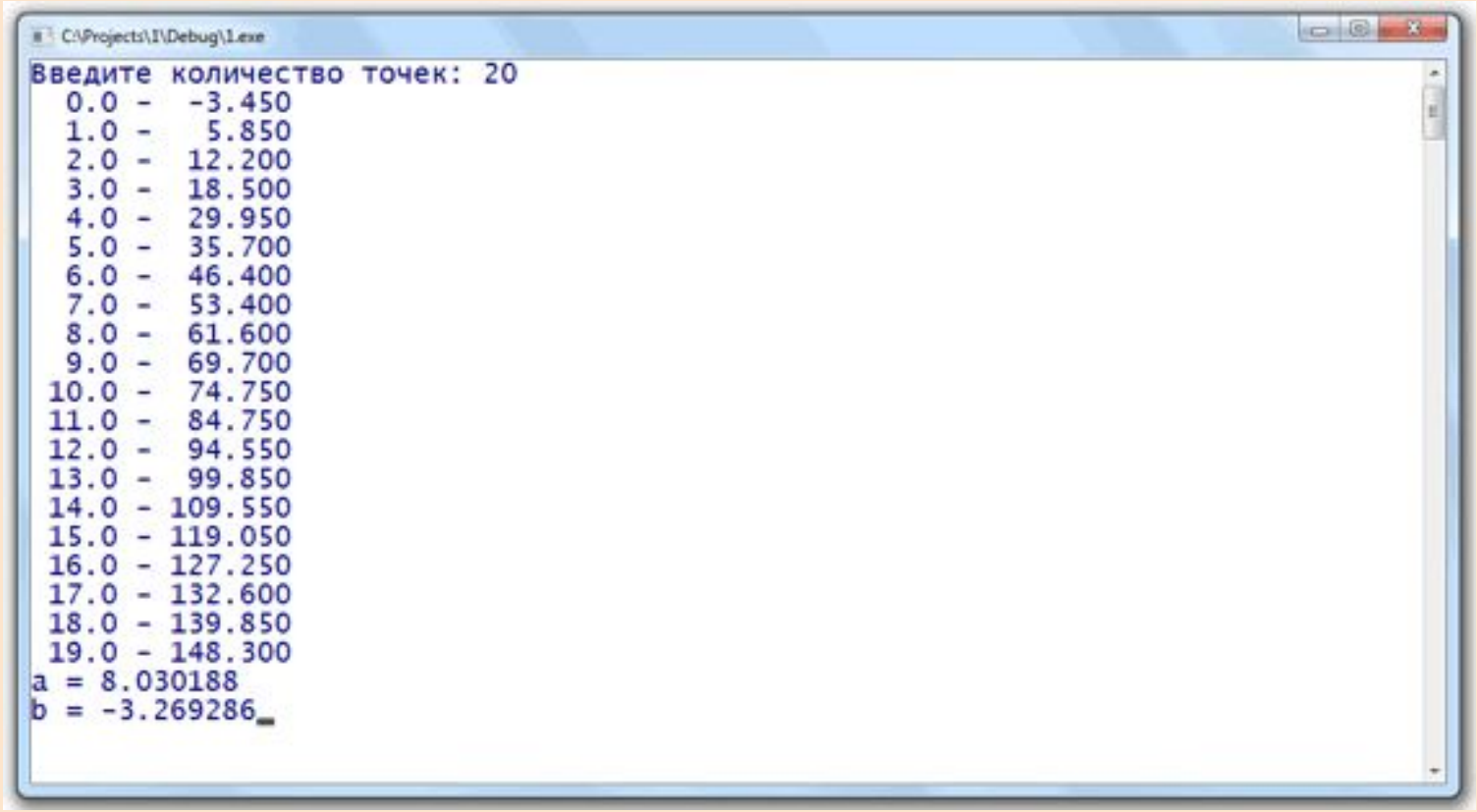
```
#include <stdio.h>
#include <stdlib.h>
// Задание начального набора значений
double ** getData(int n) {
  double **f;
  f = new double* [2];
  f[0] = new double[n];
  f[1] = new double[n];
  for(int i=0; i<n; i++) {
        f[0][i] = (double)i;
        f[1][i] = 8*(double)i - 3;
  // Добавление случайной составляющей
  //f[1][i] = 8*(double)i - 3 + ((rand()%100)-50)*0.05;
  }
  return f;
}
```

```
// Вычисление коэффициентов аппроксимирующей прямой
void getApprox(double **x, double *a, double *b, int n) {
  double sumx = 0;
  double sumy = 0;
  double sumx2 = 0;
  double sumxy = 0;
  for(int i=0; i<n; i++) {
            sumx += x[0][i];
            sumy += x[1][i];
            sumx2 += x[0][i]*x[0][i];
            sumxy += x[0][i]*x[1][i];
  }
  *a = (n*sumxy - (sumx*sumy))/(n*sumx2-sumx*sumx);
  *b = (sumy - *a*sumx)/n;
  return;
}
int main() {
  double **x, a, b;
  int n;
  system("chcp 1251");
  system("cls");
  printf("Введите количество точек: ");
  scanf("%d",&n);
  x = getData(n);
  for(int i=0; i<n; i++)
            printf("%5.1lf - %7.3lf\n",x[0][i], x[1][i]);
  getApprox(x, &a, &b, n);
  printf("a = %lf\nb = %lf", a, b);
  getchar(); getchar();
  return 0;
}
```

Результат выполнения
Запуск без случайной составляющей



Запуск со случайной составляющей



Построение графика функции

Для наглядности построим график функции, полученный аппроксимацией по методу наименьших квадратов. Подробнее о построении графика функции описано здесь.

```
#include <windows.h>
const int NUM=70; // количество точек
LONG WINAPI WndProc(HWND, UINT, WPARAM,LPARAM);
double **x; // массив данных
// Определение коэффициентов линейной аппроксимации по МНК
void getApprox(double **m, double *a, double *b, int n) {
  double sumx = 0;
  double sumy = 0;
  double sumx2 = 0;
  double sumxy = 0;
  for(int i=0; i<n; i++) {
        sumx += m[0][i];
        sumy += m[1][i];
        sumx2 += m[0][i]*m[0][i];
        sumxy += m[0][i]*m[1][i];
  }
  *a = (n*sumxy - (sumx*sumy))/(n*sumx2-sumx*sumx);
  *b = (sumy - *a*sumx)/n;
  return;
}
```

```
// Задание исходных данных для графика
// (двумерный массив, может содержать несколько рядов данных)
double ** getData(int n) {
  double **f;
  double a, b;
  f = new double* [3];
  f[0] = new double[n];
  f[1] = new double[n];
  f[2] = new double[n];
  for(int i=0; i<n; i++) {
        double x = (double)i * 0.1;
        f[0][i] = x;
        f[1][i] = 8*x - 3 + ((rand()%100)-50)*0.05;
  }
```

```
getApprox(f, &a, &b, n); // аппроксимация
  for(int i=0; i<n; i++) {
        double x = (double)i * 0.1;
        f[2][i] = a*x + b;
  }
  return f;
}
// Функция рисования графика
void DrawGraph(HDC hdc, RECT rectClient, double **x, int n, int numrow=1) {
  double OffsetY, OffsetX;
  double MAX_X = 0;
  double MAX_Y = 0;
  double ScaleX, ScaleY;
  double min, max;
  int height, width;
  int X,Y;    // координаты в окне (в px)
  HPEN hpen;
  height = rectClient.bottom - rectClient.top;
  width = rectClient.right - rectClient.left;
```

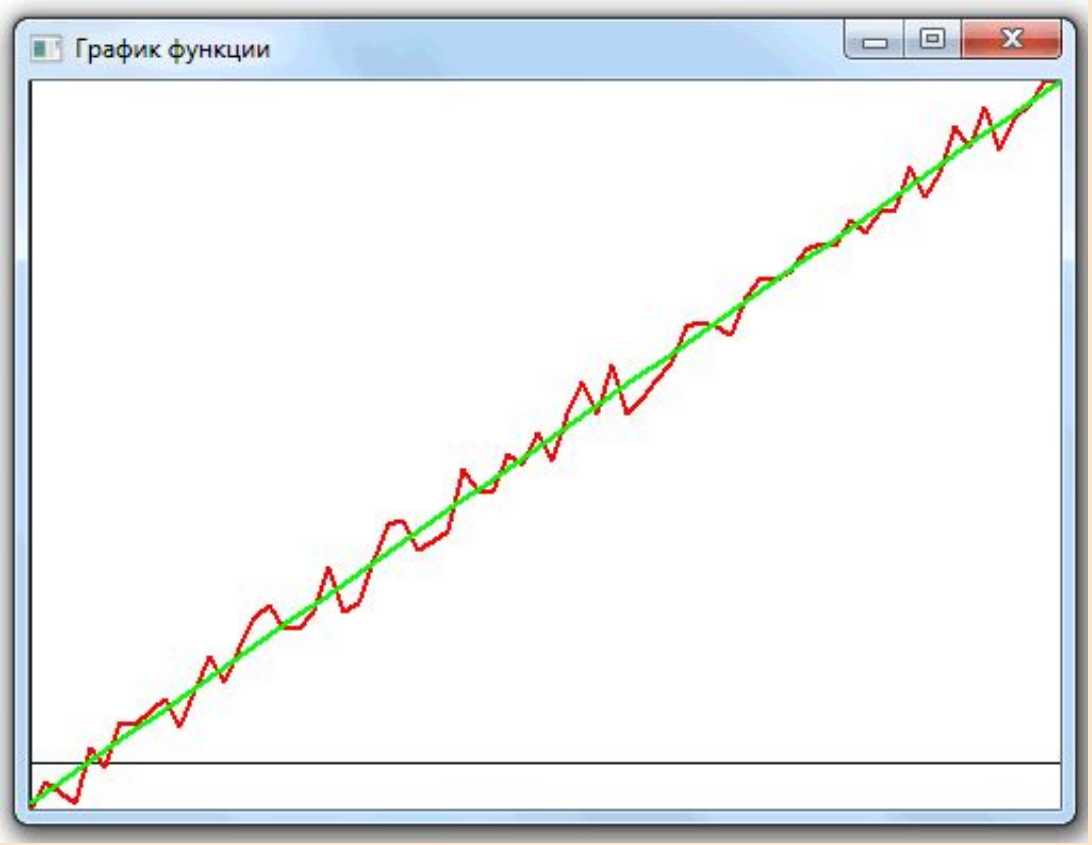
```
// Область допустимых значений X
  min = x[0][0];
  max = x[0][0];
  for(int i=0; i<n; i++) {
        if(x[0][i] < min)
        min = x[0][i];
        if(x[0][i] > max)
        max = x[0][i];
  }
  double temp = max - min;
  MAX_X = max - min;
  OffsetX = min*width/MAX_X;  // смещение X
  ScaleX = (double)width/MAX_X; // масштабный коэффициент X
```

```
// Область допустимых значений Y
 min = x[1][0];
 max = x[1][0];
 for(int i=0; i<n; i++) {
       for(int j=1; j<=numrow; j++) {
       if(x[j][i] < min)
       min = x[j][i];
       if(x[j][i] > max)
       max = x[j][i];
       }
 }
```

```
 MAX_Y = max - min;
  OffsetY = max*height/(MAX_Y);  // смещение Y
  ScaleY = (double)height/MAX_Y; // масштабный коэффициент Y
// Отрисовка осей координат
  hpen = CreatePen(PS_SOLID, 0, 0); // черное перо 1px
  SelectObject(hdc, hpen);
  MoveToEx(hdc,0,OffsetY,0);  // перемещение в точку (0;OffsetY)
  LineTo(hdc, width, OffsetY);   // рисование горизонтальной оси
  MoveToEx(hdc,OffsetX,0,0);  // перемещение в точку (OffsetX;0)
  LineTo(hdc, OffsetX, height);  // рисование вертикальной оси
  DeleteObject(hpen);  // удаление черного пера
  // Отрисовка графика функции
  int color = 0xFF; // красное перо для первого ряда данных
  for(int j=1; j<=numrow; j++) {
  hpen = CreatePen(PS_SOLID, 2, color); // формирование пера 2px
  SelectObject(hdc, hpen);
  X = (int)(OffsetX + x[0][0]*ScaleX);    // координаты начальной точки графика
  Y = (int)(OffsetY - x[j][0]*ScaleY);
  MoveToEx(hdc,X,Y,0);  // перемещение в начальную точку
  for(int i=0; i<n; i++) {
            X = OffsetX + x[0][i]*ScaleX;
            Y = OffsetY - x[j][i]*ScaleY;
            LineTo(hdc, X, Y);
  }
  color = color << 8; // изменение цвета пера для следующего ряда
  DeleteObject(hpen); // удаление текущего пера
  }
```

```
// Главная функция
int  WINAPI  WinMain (HINSTANCE  hInstance,
HINSTANCE hPrevInstance, LPSTR lpCmdLine, int nCmdShow) {
  HWND hwnd;
  MSG msg;
  WNDCLASS w;
  x = getData(NUM); // задание исходных данных
  memset(&w,0,sizeof(WNDCLASS));
  w.style = CS_HREDRAW | CS_VREDRAW;
  w.lpfnWndProc = WndProc;
  w.hInstance = hInstance;
  w.hbrBackground=CreateSolidBrush(0x00FFFFFF);
  w.lpszClassName = "My Class";
  RegisterClass(&w);
  hwnd = CreateWindow("My Class", "График функции",
  WS_OVERLAPPEDWINDOW, 500, 300, 500, 380, NULL, NULL,
                hInstance, NULL);
  ShowWindow(hwnd,nCmdShow);
  UpdateWindow(hwnd);
  while(GetMessage(&msg, NULL, 0, 0)) {
                TranslateMessage(&msg);
                DispatchMessage(&msg);
  }
  return msg.wParam;
}
// Оконная функция
LONG WINAPI WndProc(HWND hwnd, UINT Message,
WPARAM wparam, LPARAM lparam) {
  HDC hdc;
  PAINTSTRUCT ps;
  switch (Message) {
                case WM_PAINT:
                hdc = BeginPaint(hwnd, &ps);
                DrawGraph(hdc,ps.rcPaint, x, NUM, 2); // построение графика
                EndPaint(hwnd,&ps);
                break;
                case WM_DESTROY:
                PostQuitMessage(0);
     break;
                default:
                return DefWindowProc(hwnd, Message, wparam, lparam);
  }
  return 0;
}
```

Результат выполнения



https://www.youtube.com/watch?v=_-XAg8gQ2ak

Метод наименьших квадратов, урок 1/2. Линейная функция

3,6	**26,851**	**2,04**	**12,3116**	**1,296**	**0,8772**	**7,02454**			0,30570					
0,8772	1,296	2,04		7,02454										
1,296	2,04	3,6		12,3116										
2,04	3,6	8		26,851										
59,52381	-53,5714	8,928571	a=	-1,68155										
-53,5714	50,59524	-9,10714	b=	2,057798										
8,928571	-9,10714	1,946429	c=	2,859161										

Аппроксимировали табличные данные квадратичной зависимостью $y = -1,68155x^2 + 2,057798x + 2,859161$ с невязкой $S = 0,30570$.