ЛАБОРАТОРНАЯ РАБОТА №4

ТЕМА: ИНТЕРПОЛЯЦИЯ И АППРОКСИМАЦИЯ ФУНКЦИЙ

При исследовании различных явлений (физических, экономических, социальных и др.) получают экспериментальные данные, отражающие закономерности, лежащие в основе этих явлений. Во многих случаях эту зависимость можно выразить функцией, но какой именно сразу не ясно. Требуется приближенно определить эту зависимость по опытным данным.

Пусть в результате n измерений получен ряд экспериментальных точек (x_i, y_i). Известно, что через n точек можно всегда провести кривую, аналитически выражаемую многочленом (n-1)-й степени. Этот многочлен называют *интерполяционным*. И вообще, замену функции $\varphi(x)$ на функцию $\psi(x)$ так, что их значения совпадают в заданных точках, называют *интерполяцией*.

ЗАДАНИЕ

Данные, получаемые в реальном эксперименте, немного отличаются от реальных значений в силу погрешностей, накладываемых неисправностями и несовершенством измерительных инструментов, ошибками исследователя и т.д. Начнем с того, что для готовой функциональной зависимости сформируем данные, немного отличающиеся от точных.

1. Сформируем исходную зависимость и выведем её график на экран.

```
> restart:
> n:=10:
> # Формируем список с точками прообраза экспериментальной зависимости
> X:=evalf([i$i=1..n]);
```

$$X := [1., 2., 3., 4., 5., 6., 7., 8., 9., 10.]$$

```
> # Вид исхоой звисимости
> y:=x->a[0]+a[1]/x;
```

$$y := x \to a_0 + \frac{a_1}{x}$$

```
> # Параметры зависимости y(x) и шума
> a[0]:=2:a[1]:=-1:mu:=0:sigma:=0.1:
> # Список с точками образа исходной зависимости
> Z:=map(y,X);
```

$$Z := [1.000000000, 1.500000000, 1.666666667, 1.750000000, 1.800000000, 1.833333333, 1.857142857, 1.875000000, 1.888888889, 1.900000000]$$

```
> # Построим график для исходной зависимости
> with(plots):
> grOriginal:=plot(y(x),x=0..10,color=red):
> display(grOriginal);
```

2. Сформируем список точек экспериментальной зависимости так, будто они получены действительно в проведенном опыте.

> *# Список с точками нормального шума*

```
> with(stats):
> Noise:=[random[normald[mu,sigma]](n)];
```

Noise := [0.1275676034, 0.04761860895, −0.2620054215, 0.08547207946, −0.1089001803, 0.07165427609, −0.002996436194, 0.1330855643, −0.1296345120, −0.04703304005]

> *# Список с точками образа экспериментальной зависимости (прибавляем шум).*

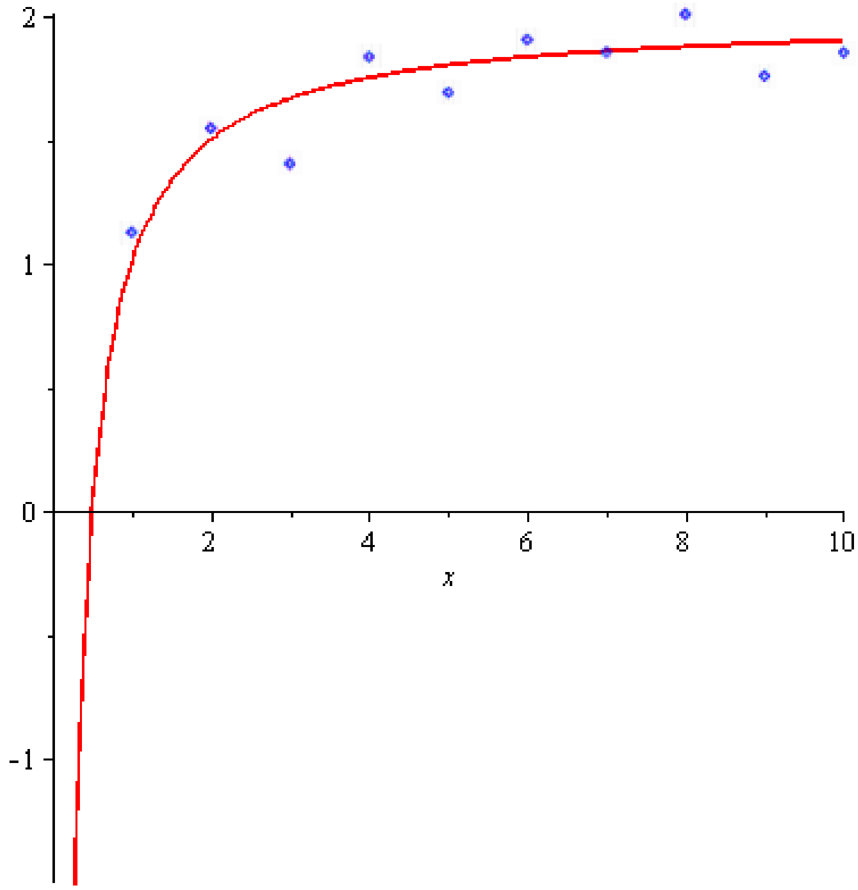
```
> Y:=Z+Noise:
```

> *# Нанесем на координатную плоскость график исходной зависимости и точечный график полученной экспериментальной*

```
> Expr:=zip((x,y)->[x,y],(X,Y));
```

Expr := [[1., 1.127567603], [2., 1.547618609], [3., 1.404661246], [4., 1.835472079], [5., 1.691099820], [6., 1.904987609], [7., 1.854146421], [8., 2.008085564], [9., 1.759254377], [10., 1.852966960]]

```
> grExpr:=pointplot(Expr,color=blue):
> display(grOriginal,grExpr);
```



Посмотрите внимательно на функции, которые мы использовали и определите, что именно они делают.

Запомните один из способов вывода нескольких графиков с разными стилями на одну координатную плоскость.

3. Подберем интерполирующий полином для точек, которые получены экспериментально с помощью функции **interp()**.

```
> yInterp:=interp(X,Y,x);
```

$$yInterp := 0.0001265574603\,x^9 - 0.006325069653\,x^8 + 0.1356206635\,x^7 - 1.630475028\,x^6 + 12.04848726\,x^5 - 56.37403723\,x^4 + 165.5832232\,x^3 - 291.0777890\,x^2 + 273.7226335\,x - 101.2738972$$

```
> # Нанесем график полученной функции на координатную плоскость
> grInterp:=plot(yInterp,x=1..10,color=green):
> display(grExpr,grInterp);
```

График полученной функции должен проходить в точности через каждую заданную точку. Однако он совершенно не похож на график исходной зависимости. И если использовать этот полином для нахождения значения функции в точках, лежащих между заданными, то можно получить значение, сильно отличающееся от действительного.

4. С помощью функции **eval** найдите значение исходной функции в точке x=1.5, и значение полинома в этой же точке. Сравните результаты.

ЗАДАНИЕ 2

Так как экспериментальные результаты имеют погрешности, нет смысла добиваться прохождения кривой точно через каждую точку. Кривую нужно провести так, чтобы она в наименьшей степени зависела от случайных ошибок. Эта задача называется *сглаживанием* (*аппроксимацией*) экспериментальной зависимости и часто решается методом *наименьших квадратов*. Сглаживающую кривую называют *аппроксимирующей кривой* или *линией регрессии*.

Для единственности решения нужно, как и при интерполяции, оговорить класс аппроксимирующей кривой и, кроме того, критерии её близости к точкам исходных данных. В методе наименьших квадратов таким критерием является минимум суммы квадратов отклонений ординат $y(x_i)$ линии регрессии от ординат y_i экспериментальных точек. Эта сума имеет вид: $S = \sum_{i=1}^{n}(y_i - y(x_i))^2$.

В качестве аппроксимирующей функции часто выбираются степенные полиномы.

Попробуем использовать в качестве аппроксимирующей функции полином первой степени (линейную функцию): $y(x) = ax + b$. Уравнение для суммы квадратов отклонений примет в этом случае вид:
$S = \sum_{i=1}^{n}(y_i - ax_i - b)^2$.

Для минимизации S вычислим частные производные от S по a и b и приравняем их нулю. Получим следующую систему линейных уравнений:

$$\begin{cases} \sum_{i=1}^{n} 2(y_i - ax_i - b)(-x_i) = 0; \\ \sum_{i=1}^{n} 2(y_i - ax_i - b)(-1) = 0. \end{cases}$$

Решая эту систему относительно неизвестных коэффициентов, найдем:

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

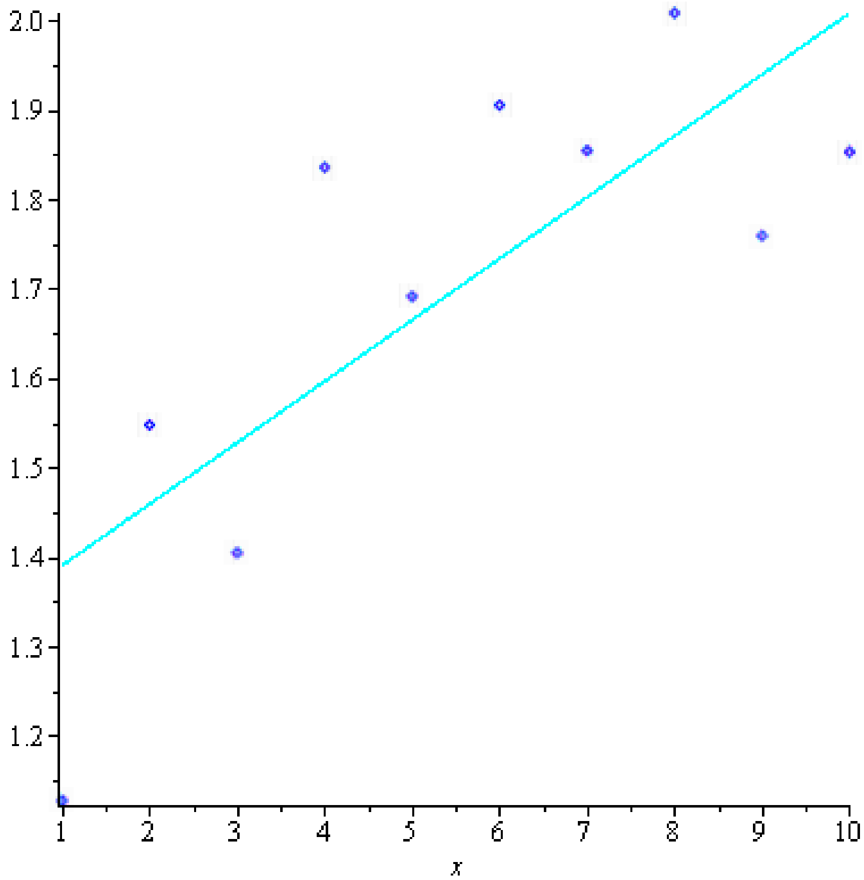
1. Используя полученные формулы для коэффициентов, рассчитайте их самостоятельно в программе. Для вычисления сумм используйте функцию **sum()**, пример использования которой приведен ниже:

```
> SX:=sum(X[i],i=1..n);
```

$$SX := 55.$$

2. После подсчета коэффициентов создайте с их использованием саму функцию и выведите её на экран вместе с точечным графиком экспериментальных данных.

```
> regr1:=a1*x+b1;
> gr1:=plot(regr1,x=1..10,color=cyan):
> display(grExpr,gr1);
```



3. Используем теперь команду **Fit()** из пакета **Statistics**, предназначенную для подгонки функции к конкретной её модели и данным.
   Fit(f, X, Y, x)
   f – модель функции;
   X – вектор или матрица значений независимой переменной;
   Y – вектор значений зависимой переменной;
   v – имя независимой переменной, которое будет использоваться в результате.

```
> with(Statistics):
> regr11:=Fit(a*x+b,X,Y,x);
```

Сравните полученные функции **regr1** и **regr11**. Если их коэффициенты сильно различаются, где-то была допущена ошибка.

4. Как видно из графика, линейная регрессия (прямая линия) плохо аппроксимирует экспериментальные точки. Подберите кубическую регрессию (полином третьей степени) для тех же данных с помощью команды **Fit()**. Затем аналогично с линейным графиком, выведите этот на экран вместе с экспериментальными точками и графиком оригинальной функции.

В интервале [1,10] аппроксимирующая кривая должна быть очень близка к оригинальной кривой.

5. Теперь проделайте то же с командой **Fit()**, но для модели *a+b/x*, которая и была использована при создании исходной функции.

Когда вид зависимости ещё не известен, исследователь должен сделать предположение, основываясь на расположении точек на плоскости. Это может быть одна из функций, приведенных в таблице ниже, либо их комбинация, либо что-то ещё более сложное.

Гиперболическая	$y = \frac{a}{x} + b$
Логарифмическая	$y = a \cdot \ln(x) + b$
Показательная	$y = b \cdot e^{a \cdot x}$
Степенная	$y = b \cdot x^{a}$
Комбинированная	$y = \frac{1}{a \cdot e^{-x} + b}$

ЗАДАНИЕ 3

1. Постройте точечный график по данным для своего варианта.
2. Выберите три вида зависимости (модели функции), наиболее подходящих для данных из вашего варианта. Сформируйте с помощью команды Fit() функции для выбранных моделей со своими данными.
3. Выведите графики созданных функций на одну систему координат разными цветами. Диапазон изменения независимой переменной указывать в соответствии с крайними точками.
4. Изучите контекстное меню графика.

Варианты.

1. X = [2.2, 2.5, 4, 5.3, 6.1, 8, 10.1, 11, 12.9, 14]
   Y = [5.0, 4.8, 4.2, 4.5, 4.2, 4.3, 4.2, 4.3, 4.0, 4.1]
2. X = [0.6, 1.2, 1.8, 2.6, 5.5, 6.5, 9.9, 10.8, 11.5, 12.7]
   Y = [-6.4, -3.0, -1.3, 0.86, 4.4, 5.4, 7.5, 8.0, 8.1, 8.7]
3. X = [1.7, 2.2, 3.8, 4.4, 5.3, 5.9, 8.4, 10.1, 12.7, 14.8]
   Y = [-0.11, -0.058, -0.095, 0.33, 0.22, 0.46, 0.56, 0.79, 0.64, 0.80]
4. X = [0.5, 1.2, 1.9, 3.1, 5.5, 6.1, 7.9, 9.1, 10.5, 11]
   Y = [0.21, -2.5, -4.2, -5.3, -7.2, -7.4, -8.2, -8.5, -9.2, -9.2]
5. X = [-6.1, -3.2, -2.1, 1.6, 2.6, 6.1, 7.3]
   Y = [-1.4, 4.0, 5.1, 5.5, 4.6, -1.4, -4.7]
6. X = [1.4, 2.2, 3.4, 4.5, 4.9, 5.2, 6.1]
   Y = [0.72, 1.5, 3.2, 6.2, 7.1, 8.2, 11]
7. X = [1.2, 2.5, 3.7, 4.5, 5.3, 5.7, 6.1, 7.2, 7.6, 8]
   Y = [0.76, 0.87, 0.79, 1.3, 1.3, 1.6, 1.7, 2.2, 2.2, 2.4]
8. X = [-2.4, -1.2, -0.4, 1.5, 3.9, 5.2, 7.1]
   Y = [-28, -3.4, -0.39, 6.8, 120, 280, 720]
9. X = [-3.1, -1.2, -0.1, 1.6, 2.6, 3.1, 3.9]
   Y = [-0.83, 2.4, 3.0, 2.0, 0.29, -0.84, -3.1]
10. X = [1.2, 2.2, 3.4, 4.3, 5.1, 7, 8.1, 9.2, 10.5, 13]
    Y = [-3.4, -2.3, -2.1, -1.6, -1.7, -1.4, -1.4, -1.2, -1.4, -1.3]
11. X = [1.3, 2.1, 3.2, 4, 5.1, 5.4, 6.2, 7, 7.4, 7.8]
    Y = [-0.52, -0.71, -1.2, -1.0, -1.5, -1.4, -1.7, -1.9, -2.3, -2.4]
12. X = [2.2, 2.5, 3.1, 4.2, 6.1, 7.3, 8.5, 9.8, 10.9, 12]
    Y = [4.9, 4.6, 4.0, 4.0, 3.5, 3.6, 3.5, 3.5, 3.2, 3.3]
13. X = [1.4, 2.2, 3.3, 4, 4.1, 5.6, 6.5, 7.1, 7.5, 7.9]
    Y = [2.5, 2.3, 1.7, 1.8, 1.5, 0.93, 0.19, -0.23, -0.92, -1.3]
14. X = [2.2, 2.5, 4, 5.3, 6.1, 8, 10.1, 11, 12.9, 14]
    Y = [-1.4, -1.6, -2.0, -1.7, -1.9, -1.8, -1.9, -1.8, -2.1, -2.0]
15. X = [-6.1, -4.2, -1.6, 1.6, 3.8, 6.9, 7.2]
    Y = [46, 15, 0.56, -0.73, -11, -66, -75]

16. X = [-1.1, -0.2, 1.1, 1.6, 4.6, 5.7, 5.9]
    Y = [-3.4, -4.0, -3.4, -2.7, 6.6, 12, 13]
17. X = [0.1, 1.6, 2.4, 4.1, 6.5, 7.8, 8.9, 10.2, 11.4, 12]
    Y = [-1.5, 4.0, 4.5, 5.9, 6.6, 7.2, 7.4, 7.8, 7.7, 7.9]
18. X = [-6.1, -4.2, -3.1, 1.6, 4.2, 5.7, 7.7]
    Y = [300, 110, 47, -0.89, -41, -120, -340]
19. X = [-6.3, -4.1, -2.7, 2.6, 3.1, 4.6, 5.7]
    Y = [-330, -100, -35, 4.1, 10, 55, 120]
20. X = [-6.1, -4.2, -3.1, 1.6, 4.2, 5.7, 7.7]
    Y = [0.015, 0.019, 0.015, 0.46, 0.49, 0.51, 0.50]
21. X = [1.5, 2.3, 3.2, 4.1, 4.5, 5.1, 6.2, 6.6, 7.1, 7.3]
    Y = [-1.4, -1.4, -1.5, -0.89, -0.95, -0.54, -0.076, 0.31, 0.39, 0.63]
22. X = [1.7, 2.2, 3.8, 4.4, 5.3, 5.9, 8.4, 10.1, 12.7, 14.8]
    Y = [-0.11, -0.058, -0.095, 0.33, 0.22, 0.46, 0.56, 0.79, 0.64, 0.80]
23. X = [-6.1, -3.2, -2.1, 1.6, 2.6, 6.1, 7.3]
    Y = [-1.4, 4.0, 5.1, 5.5, 4.6, -1.4, -4.7]
24. X = [1.4, 2.2, 3.4, 4.5, 4.9, 5.2, 6.1]
    Y = [0.72, 1.5, 3.2, 6.2, 7.1, 8.2, 11]
25. X = [-2.4, -1.2, -0.4, 1.5, 3.9, 5.2, 7.1]
    Y = [-28, -3.4, -0.39, 6.8, 120, 280, 720]
26. X = [-3.1, -1.2, -0.1, 1.6, 2.6, 3.1, 3.9]
    Y = [-0.83, 2.4, 3.0, 2.0, 0.29, -0.84, -3.1]
27. X = [2.2, 2.5, 3.1, 4.2, 6.1, 7.3, 8.5, 9.8, 10.9, 12]
    Y = [4.9, 4.6, 4.0, 4.0, 3.5, 3.6, 3.5, 3.5, 3.2, 3.3]
28. X = [2.2, 2.5, 4, 5.3, 6.1, 8, 10.1, 11, 12.9, 14]
    Y = [-1.4, -1.6, -2.0, -1.7, -1.9, -1.8, -1.9, -1.8, -2.1, -2.0]
29. X = [-6.1, -4.2, -1.6, 1.6, 3.8, 6.9, 7.2]
    Y = [46, 15, 0.56, -0.73, -11, -66, -75]
30. X = [-1.1, -0.2, 1.1, 1.6, 4.6, 5.7, 5.9]
    Y = [-3.4, -4.0, -3.4, -2.7, 6.6, 12, 13]