# Построение наилучшего приближения функции методом наилучшего среднеквадратичного приближения.(1.3.2(1.3.3Ф))

## Описание метода

Для данной функции необходимо построить наилучшее приближение, используя набор базисных линейно независимых функций, которые ко всему прочему еще и ортогональны. В данной задаче этими функциями являются функции ряда Фурье.

Функция, подаваемая на вход, должна быть определена на симметричном интервале, для того, что аппроксимация работала, т.е. в промежутке от -*l* до *l*.

Тогда можно построить аппроксимацию по Фурье для этой функции, которая будет выглядеть следующим образом:

$$g(x) = \sum_{k=0}^{n} \left( a_k \sin\left(\frac{k\pi x}{l}\right) + b_k \cos\left(\frac{k\pi x}{l}\right) \right)$$

Коэффициенты $a_k$ будут находиться из следующих соображений.

Из идеи метода наилучшего среднеквадратического приближения будет следовать следующая формула:

$$\sum_{k=0}^{n} a_k \int_a^b \rho(x)\varphi_k(x)\varphi_i(x)dx = \int_a^b \rho(x) f(x)\varphi_i(x)dx$$

Здесь весовая функция $\rho(x) = \frac{1}{sqrt(l)}$. Она сразу объединена с нормирующим множителем для менее громоздкой записи.

Здесь в общем виде нужно было бы решать систему линейных уравнений, но функции слева ортонормированы. Это значит, что их попарные произведения будут равны 1, если инедксы совпадают и 0 в остальных случаях.

Тогда из ортонормированности функций слева получаем, что

$$a_i = \int_a^b \rho(x) f(x)\varphi_i(x)dx,\ i = \overline{0,n}$$

И таким образом будут вычисляться все коэффициенты.

## Необходимые формулы:

$$a_i = \int_a^b \rho(x) f(x)\varphi_i(x)dx,\ i = \overline{0,n}$$

$$\rho(x) = \frac{1}{sqrt(l)}$$

Программная реализация

```
fourierFunc[x_, a_, b_] := a Cos[x] + b Sin[x];
getCoef[f_, l_, n_] := Module[{ak, bk},
  ak = Table[NIntegrate[f[x] Cos[k Pi/l x], {x, -l, l}] / l, {k, 0, n - 1}];
  bk = Table[NIntegrate[f[x] Sin[k Pi/l x], {x, -l, l}] / l, {k, 0, n - 1}];
  bk[[1]] = 0;
  ak[[1]] /= 2;
  {ak, bk}
 ]

approximation[f_, l_, n_] := Module[{coef, ak, bk, g},
  coef = Quiet@getCoef[f, l, n];
  ak = coef[[1]];
  bk = coef[[2]];
  g[x_] := Total@Table[fourierFunc[x (Pi (i - 1))/l, ak[[i]], bk[[i]]], {i, n}];
  g
 ]
showResults[f_, l_] := Module[{result},
  result = approximation[f, l, 10];
  Show[
   Plot[f[x], {x, -l, l}],
   Plot[result[x], {x, -l, l}, PlotStyle → Orange]
  ]
 ]
```

## Проверка правильности решения

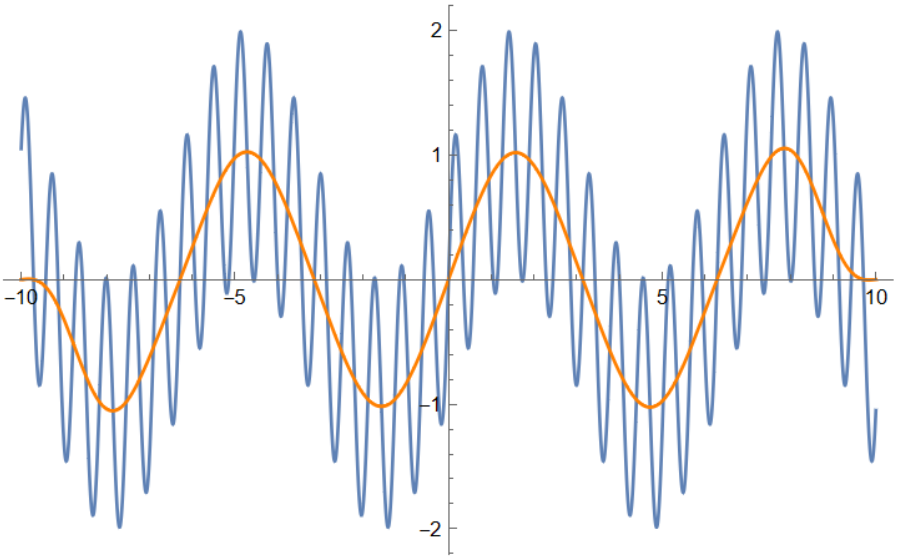
Сначала проверю результат на какой-нибудь периодической функции, у которой будет некая имитация шума в виде периодической функции с очень малым периодом.

Буду брать 10 базисных функций.

$f(x) = \sin(x) + \sin(20x)$

Результат:

```
f1[x_] := Sin[10 x] + Sin[x];
showResults[f1, 10]
```

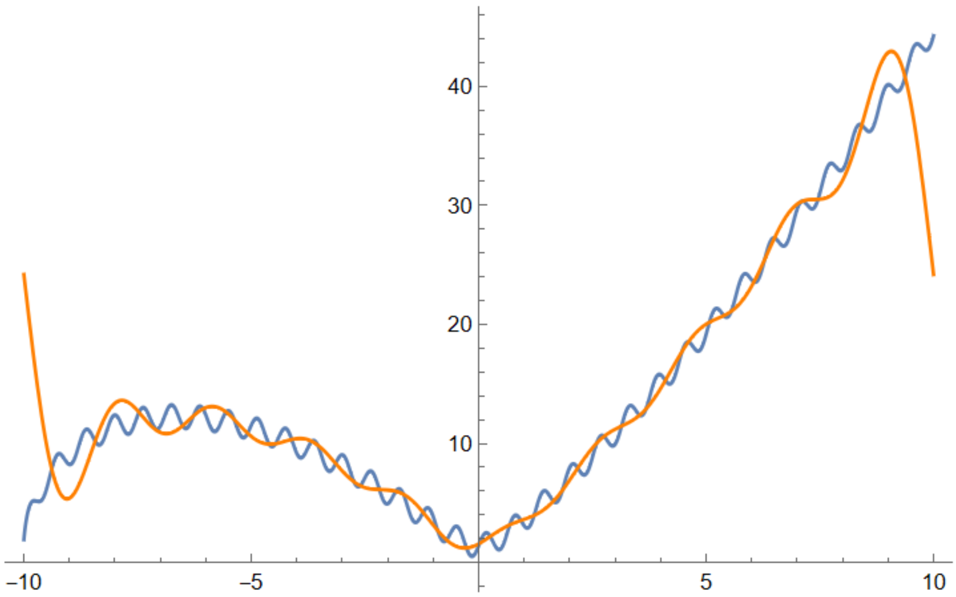



Как видно, программная реализация метода отлично справляется с задачей, действительно можно заметить, что итоговая функция очень похожа на исходную, но без шума.

Теперь возьму какую-нибудь аналитическую функцию с тем же шумом:

$f(x) = \sqrt{x^3 + 10x^2 + 2} + \sin(10x)$

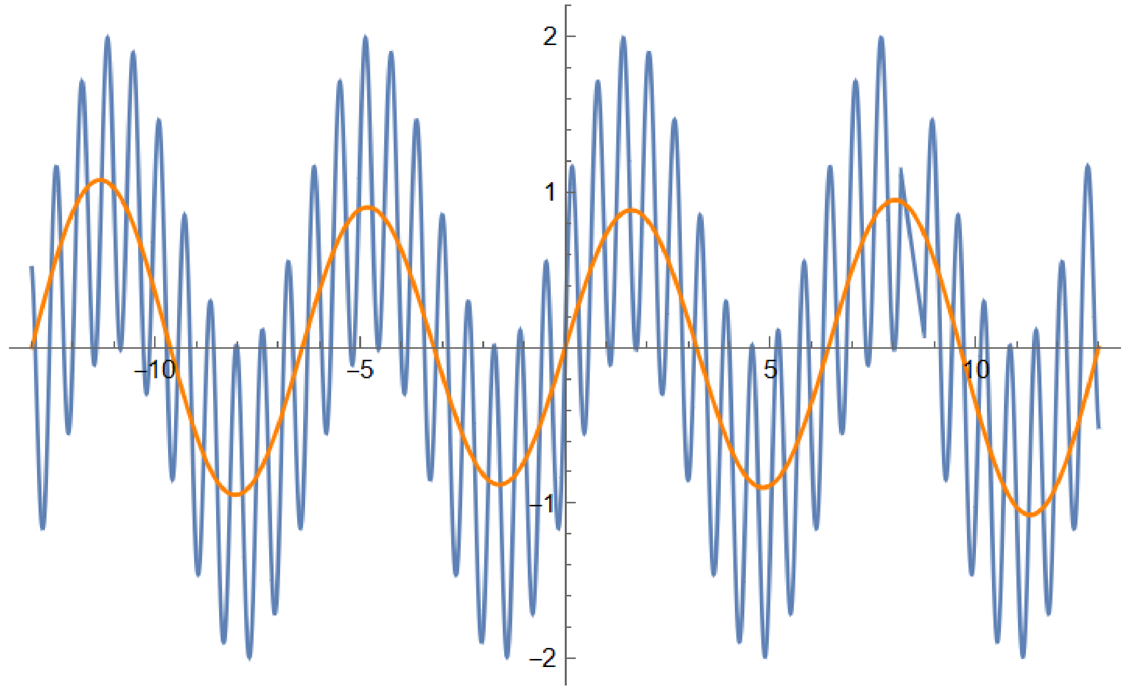
Результат:



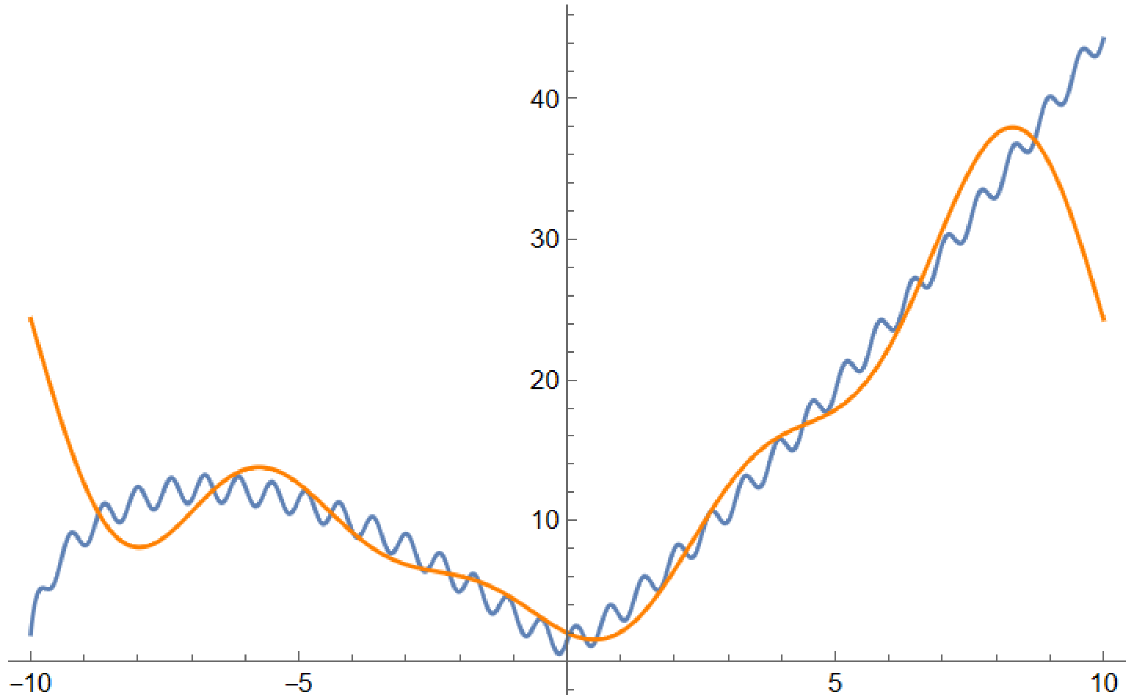
Возьму чуть меньше базисных функций и посмотрю что получилось для тех же функций, например 5:

```
showResults[f1, 13]
showResults[f2, 10]
```

]=




]=




Видно, что полученные функции дают чуть менее хороший результат, но в целом он приемлемый.

Подведя итог, можно сказать, что реализованная функция дает очень даже хорошие результаты. Причем, полученные результаты довольно хорошо подтверждают теоретические данных на основе которых был реализован метод.