Московский Авиационный Институт (национальный исследовательский университет)

Факультет Прикладной математики и физики
Кафедра Вычислительной математики и программирования

Курсовая работа
по дисциплине
«Численные методы»

Аппроксимация функций с использованием вейвлет-анализа

Студент: Будникова В. П.
Группа: М08-307Б-19
Руководитель: Ревизников Д. Л.

Москва 2022

Описание

Вейвлет-преобразование принято разделять на дискретное вейвлет-преобразование и непрерывное вейвлет-преобразование.

Дискретное вейвлет-преобразование – вейвлет-преобразование, в котором вейвлеты представлены прерывными выборками сигналов (т. е. метод направлен на работу с дискретными данными). Вейвлет может быть сконструирован из функции масштаба, которая описывает свойства его масштабируемости. Ограничение состоит в том, что функция масштаба должна быть ортогональна к своим дискретным преобразованиям, что подразумевает некоторые математические ограничения на них. Дискретное вейвлет-преобразование может использоваться для простого и быстрого удаления шума с зашумлённого сигнала.

Принц работы дискретного преобразования:

Расчет коэффициентов:

1. Рассчитать постоянную составляющую сигнала.

2. Рассчитать свертку сигнала с базисным вейвлетом, растянутым на всю временную ось.

3. Сжать базисный вейвлет в два раза.

4. Рассчитать коэффициенты сверки с первой и второй половинами сингала.

5. Сжать базисный вейвлет еще в 2 раза и посчитать соответствующие коэффициенты сигнала.

6. Продолжать аналогично п. 5

Непрерывное вейвлет-преобразование – вейвлет-преобразование, в котором вейвлеты представлены непрерывными выборками сигналов. Данный метод используется для обработки непрерывных сигналов и применяется в научных исследованиях в областях радиотехники, связи, геофизики и других отраслях науки и техники

Для аппроксимации функций рассмотрим сингулярное вейвлет преобразование.

Вейвлет преобразование – преобразование, которое использует функции, локализованные как в реальном, так и в Фурье-пространстве.

$$Wf(a,\ b) = \int_{-\infty}^{\infty} f(x)\psi^{*}_{a,\ b}(x)dx$$

Последовательность вейвлет преобразований

Лемма

Если функция $f(x)$ непрерывна в точке x, то преобразование $W(f - f(x))(x,\ a)$ непрерывно в точке x.

$F^{k+1}(x) = F^{k}(x) - WF^{k}(x,\ a_k)$, где $WF^{k}(x,\ a_k) = \frac{1}{a_k}\int_{-\infty}^{\infty} \psi\left(\frac{t - x}{a_k}\right)dt$ – вейвлет преобразование.

Теорема

Для непрерывной в точке x функции $f(x)$ существует разложение по формуле

$f(x) = \sum_{k=0}^{K-1} F^{k}(x\text{б}\ a_k) + F^{K}(x)$, где a_k – произвольная последовательность действительных чисел, k – номер преобразований $F^{K}(x) = W(F^{K-1} - F^{K-1}(x))(x, a_k)$ – остаточный член, К – порядок аппроксимации

Равномерная сходимость вейвлет ряда

Пусть $F^{k}(x)$ – последовательность вейвлет преобразований, с начальной функцией $F^{0}(x) = f(x)$, и $WF^{k}(x,\ a_k) = \frac{1}{a_k}\int_{-\infty}^{\infty} \psi\left(\frac{t - x}{a_k}\right)dt$, ряд $\sum_{k=0}^{\infty} WF^{k}(x,\ a_k)$ называется вейвлет-рядом.

Алгоритм аппроксимации:

Дано: Набор точек $X_n = \{x_0, x_1, \ldots, x_n\}$ и их значения $Y_n = \{y_0, y_1, \ldots, y_n\}$

1. Присвоить начальные значения коэффициентам вейвлета первого порядка

$$W_i^0 = y_i, \quad i = 1, n$$

2. Вычислить остальные коэффициентов вейвлет-преобразования

$$W_j^k = W_j^{k-1} - \frac{\sum_i W_i^k \psi(\frac{x_i - x_j}{a_{k-1}})}{\sum_i \psi(\frac{x_i - x_j}{a_{k-1}})}$$

W_j^k – *значение коэффициета вейвлета* k – *го порядка в точке* x_i

$$k = 1, K-1$$

$$i, j = 1, n$$

$$a_k = \alpha 2^{-k}, \ \alpha = const$$

Восстановление значений функции

Дано: Набор точек $X_n = \{x_0, x_1, \ldots, x_n\}$

$$f_K(x) = \sum_{k=1}^{K} \frac{\sum_i W_i^{k-1} \psi(\frac{x_i - x}{a_{k-1}})}{\sum_i \psi(\frac{x_i - x}{a_{k-1}})}$$

Реализация

Код функции аппроксимации:

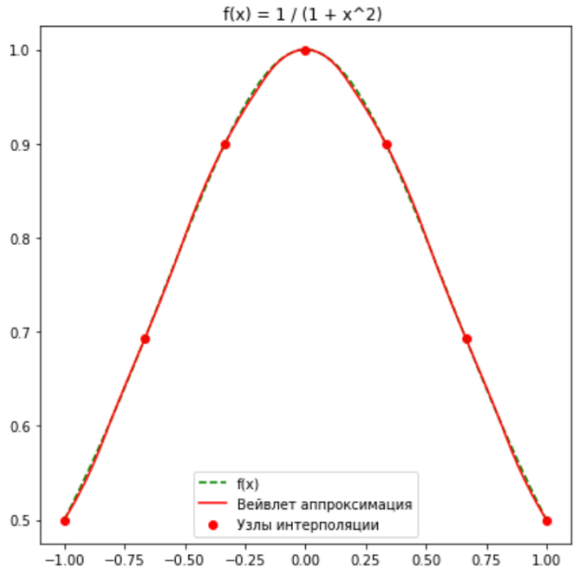
```
class WaveletApproximation:
    def __init__(self, phy, k, alph):
        self.phy = phy
        self.k = k
        self.alph = alph

    def MakeWeights(self, Xn, Yn):
        self.Xn = Xn
        self.W = [Yn]
        for k in range(1, self.k):

            ak = self.alph * pow(2, -(k - 1))
            D = []

            for j in range(len(Yn)):
                sum_W_Phi  = 0
                sum_Phi = 0
                for i, w in enumerate(self.W[-1]):
                    sum_W_Phi += w * self.phy((Xn[i] - Xn[j])/ak)
                    sum_Phi += self.phy((Xn[i] - Xn[j])/ak)
                D.append(sum_W_Phi / sum_Phi)

            Wik = [x1 - x2 for (x1, x2) in zip(self.W[-1], D)]
            self.W.append(Wik)

    def Get_Y(self, x):
        y = 0;
        for k in range(1, self.k):
            ak = self.alph * pow(2, -(k - 1))
            sum_W_Phi  = 0
            sum_Phi = 0
            for i, w in enumerate(self.W[k-1]):
                sum_W_Phi += w * self.phy((self.Xn[i] - x)/ak)
                sum_Phi += self.phy((self.Xn[i] - x)/ak)
            y += sum_W_Phi / sum_Phi
        return y
```

Функции для отрисовки графиков, подсчета погрешностей и среднеквадратичного отклонения:

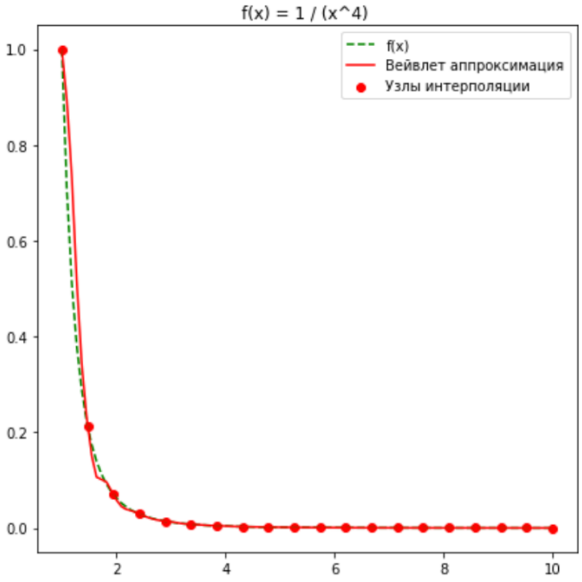
```
def Plot2(Xi, Yi, X, Y, Xans, Yans, t1 = ''):
    fig = plt.figure(figsize= (7, 7))
    plt.plot(Xans, Yans, c = 'g', linestyle='--', label="f(x)")
    plt.scatter(Xi, Yi, c = 'r', label="Узлы интерполяции")
    plt.plot(X, Y, c = 'r', label="Вейвлет аппроксимация")
    plt.title(t1)
    plt.legend()
    plt.show()

def plot_box(Xi, Yi, X, Y, Xans, Yans,t1 = ['']):

    plt.figure(figsize=(20, 20))
    l = len(X)
    a = l // 2
    b = l // 2 if l % 2 == 0 else l // 2 + 1
    for n in range(l):
        plt.subplot(a, b, n + 1)
        plt.plot(Xans[n], Yans[n], c = 'g', linestyle='--', label="f(x)")
        plt.scatter(Xi[n], Yi[n], c = 'r', label="Узлы интерполяции")
        plt.plot(X[n], Y[n], c = 'r', label="Вейвлет аппроксимация")
        plt.title(t1[n])
        plt.legend()

    plt.show()

def Plot1(aXi, aYi, aX, aY, bXi, bYi, bX, bY, f, t1 = '', t2 = ''):
    fig = plt.figure(figsize= (14, 5))
    plt.subplot(1, 2, 1)
    plt.scatter(aXi, aYi, c = 'r')
    plt.plot(aX, aY, c = 'r')
    YY = [f(x) for x in bX]
    plt.plot(aX, YY, c = 'b')
    plt.title("a)" + t1)

    plt.subplot(1, 2, 2)
    plt.scatter(bXi, bYi, c = 'r')
    plt.plot(bX, bY, c = 'r')
    YY = [f(x) for x in bX]
    plt.plot(bX, YY, c = 'b')
    plt.title("б)" + t2)

    plt.show()

def S(s1, s2):
    return math.sqrt(np.sum((np.array(s1) - np.array(s2))**2) / len(s1))

def E(s1, s2):
    return np.max(abs((np.array(s1) - np.array(s2))))
```

Результаты работы

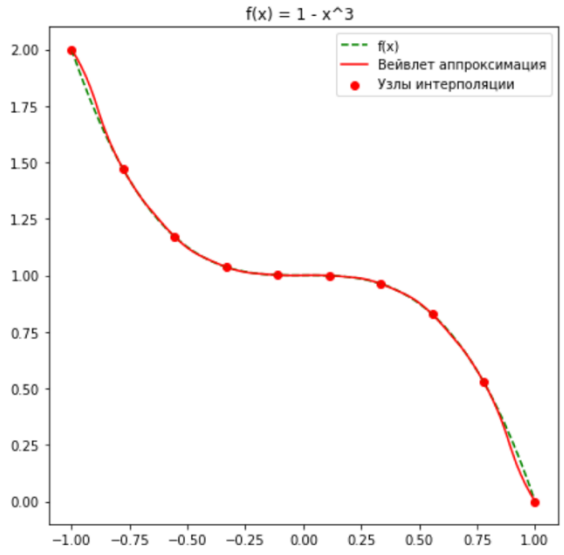
Среднеквадратичное отклонение: 0.002359194651591247
Погрешность:: 0.004998589432453238



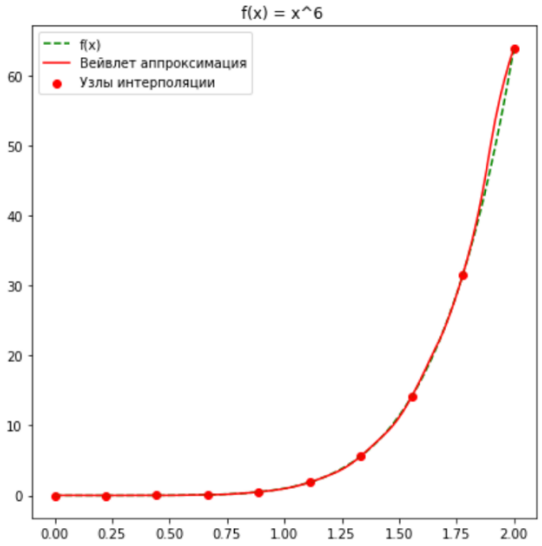
Среднеквадратичное отклонение: 0.03294537641066552
Погрешность:: 0.22342363608615945



Среднеквадратичное отклонение: 0.015775722301618406
Погрешность:: 0.056177304900358216



Среднеквадратичное отклонение: 0.7572635100729604
Погрешность:: 3.7838676956837745

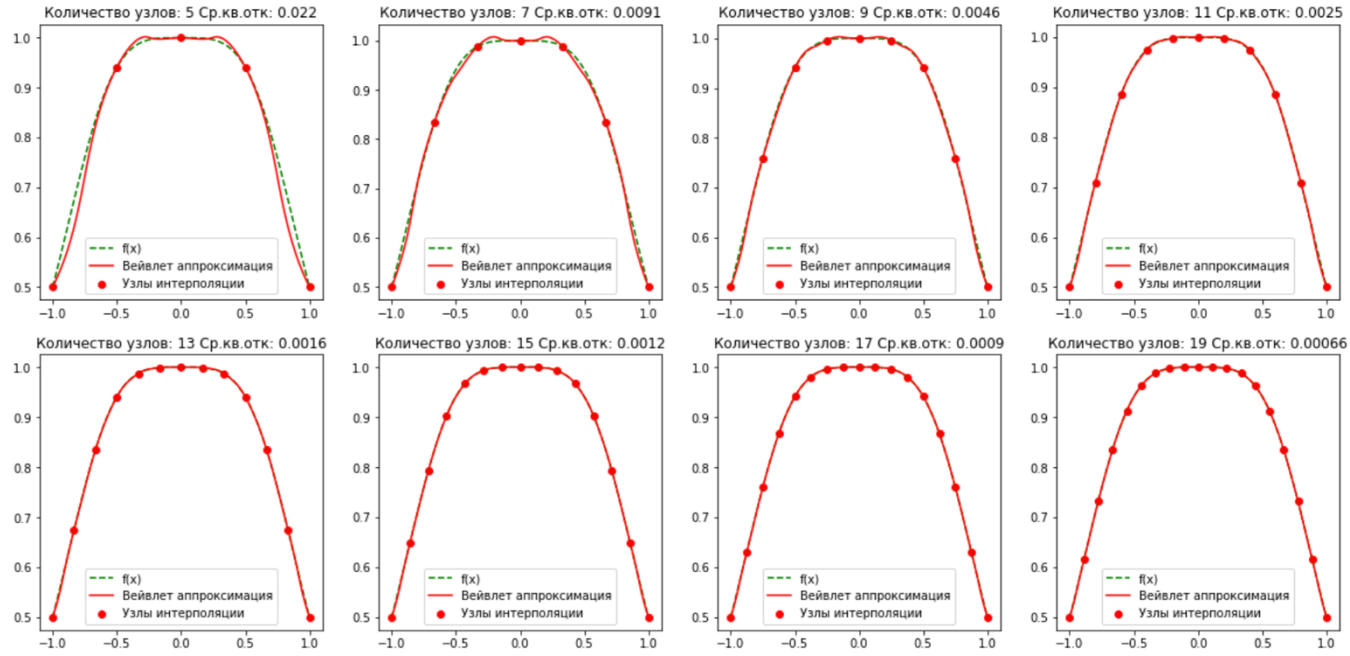


В данном примере погрешность считается как наибольшее отклонение аппроксимирующей функции от исходной.

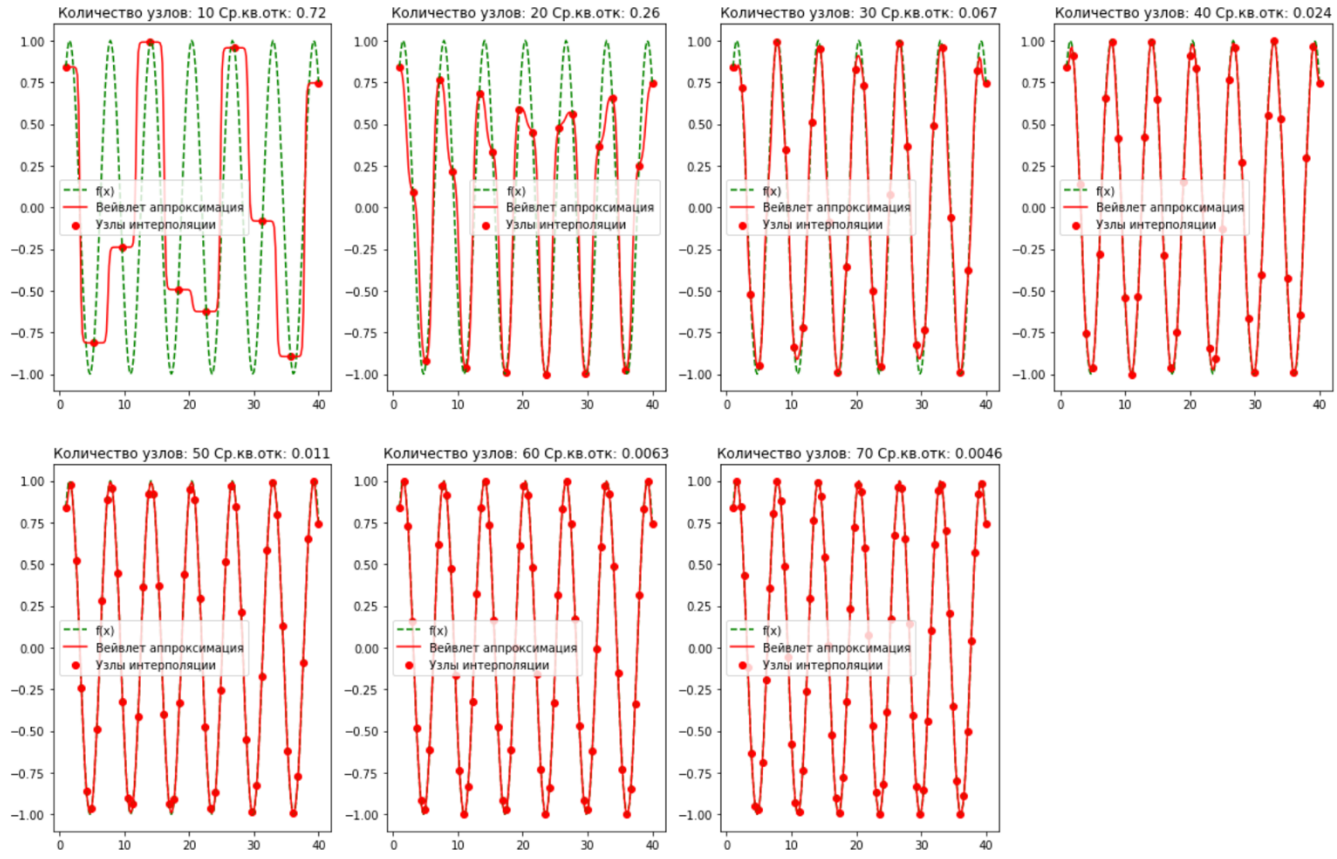
Базисный сингулярный вейвлет - e^{-x^2}

Зависимость аппроксимации от количества узлов

`f(x) = 1 / (1 + x^4)`

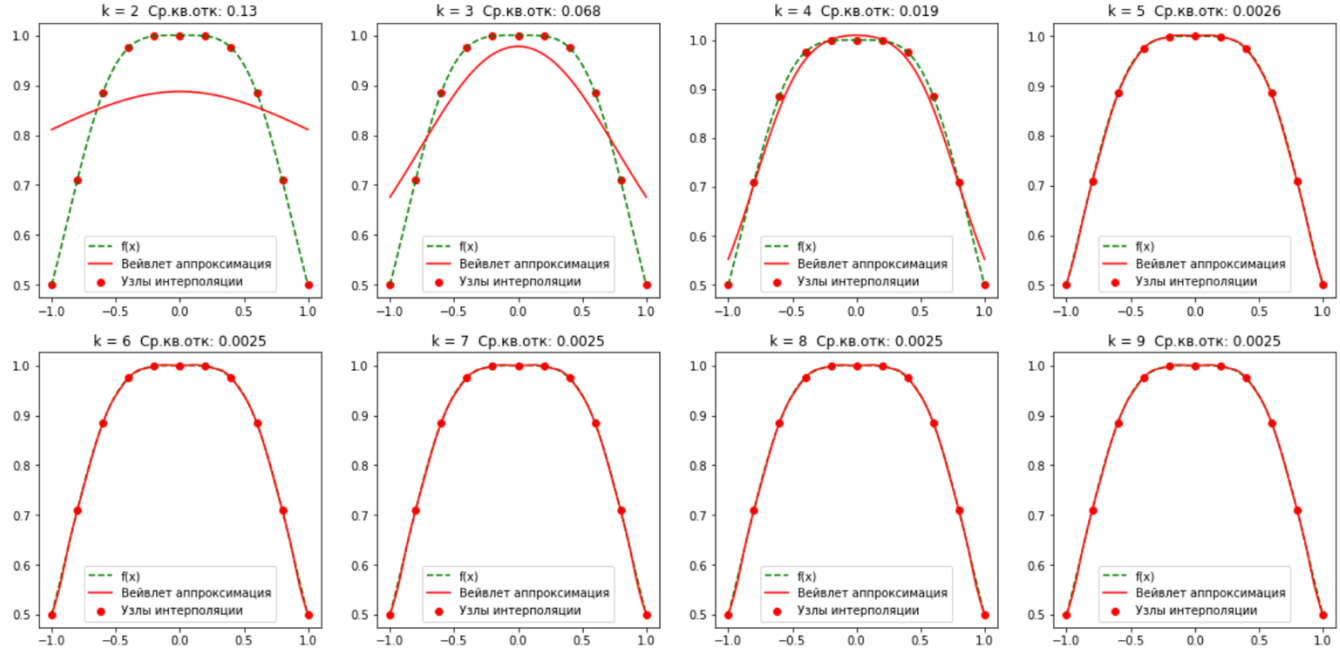


`f(x) = sin(x)`

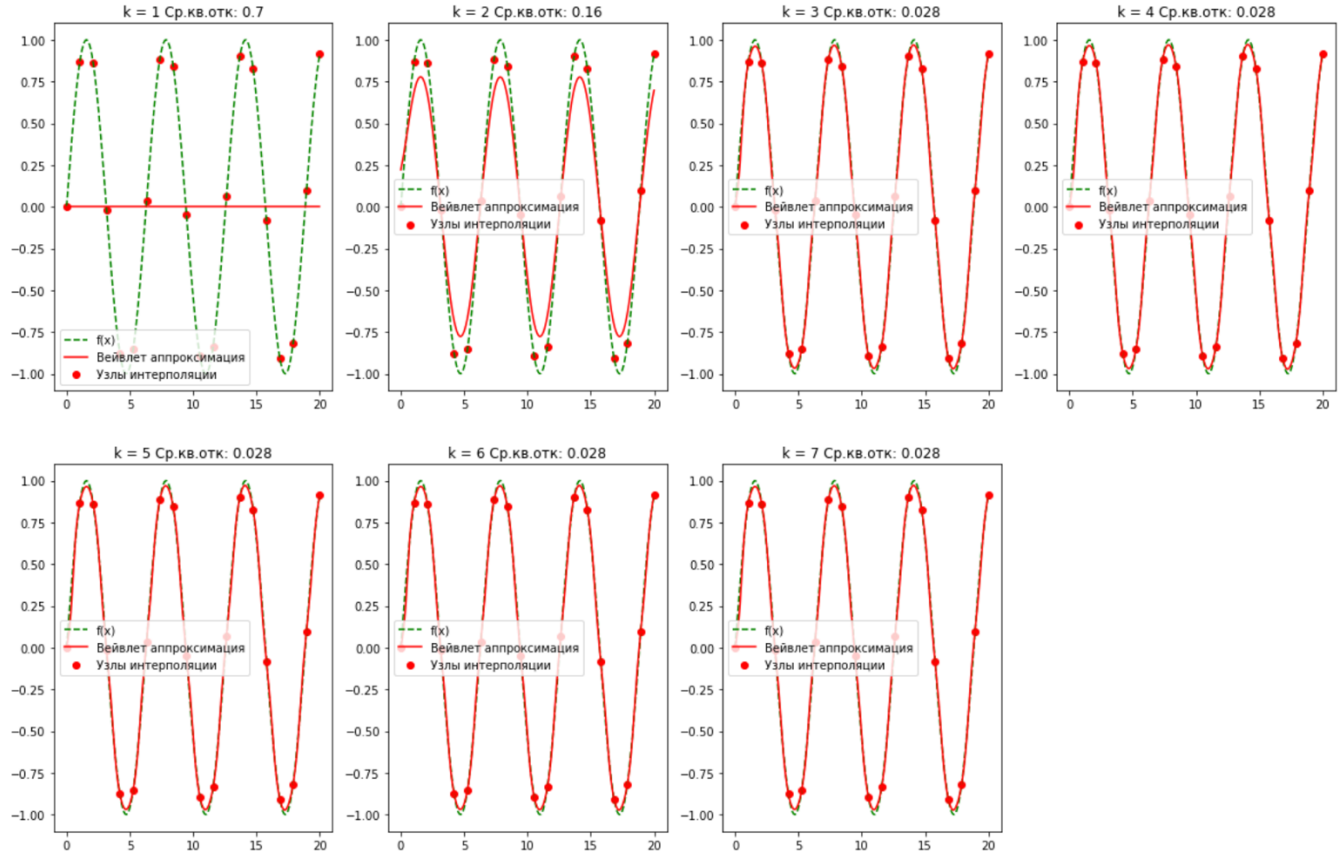


Зависимость аппроксимации от выбора порядка вейвлета

f(x) = 1 / (1 + x^4)



f(x) = sin(x)



Как можно увидеть из графиков, при высоком порядке вейвлета даже при малом количестве узлов, можно получить хорошую точность аппроксимации.

Выводы

При выполнении данной работы я познакомилась с вейвлет преобразованиями, как дискретными, так и непрерывными, реализовала сингулярное вейвлет преобразование, для аппроксимации функций. Были построены графики аппроксимации заданной функции, рассчитана среднеквадратичное отклонение и погрешность аппроксимации. Также были построены графики, на которых отражается зависимость аппроксимации от порядка вейвлета и от количества входных данных (заданных точек функции для аппроксимации).