Федеральное государственное бюджетное образовательное учреждение высшего образования «Национальный исследовательский университет «МЭИ»

**Курсовая работа**

**«Интерполирование таблично заданной в кольце функции двух переменных методом билинейных многочленов»**

по дисциплине «Численные методы»

Выполнил: Щемилкин Максим Михайлович

Преподаватель: Амосова Ольга Юрьевна

Группа: А-05-19

Москва, 2021

Оглавление

[Постановка задачи. 3](#_Toc90576888)

[Теоретический материал. 3](#_Toc90576889)

[Задача интерполяции функций. 3](#_Toc90576890)

[Многомерная интерполяция. 3](#_Toc90576891)

[Полярные координаты. 3](#_Toc90576892)

[Построение тестового примера. 4](#_Toc90576893)

[Упрощение задачи. 4](#_Toc90576894)

[Выбор функции. 5](#_Toc90576895)

[Разбиение тестового примера. 5](#_Toc90576896)

[Построение модели вычислений. 6](#_Toc90576897)

[Получение функций для каждой области. 6](#_Toc90576898)

[Сохранение. 7](#_Toc90576899)

[Результаты расчётов тестового примера. 7](#_Toc90576900)

[Вычислительные эксперименты. 10](#_Toc90576901)

[Эксперимент 1. 10](#_Toc90576902)

[Результаты 10](#_Toc90576903)

[Вывод 11](#_Toc90576904)

[Эксперимент 2. 11](#_Toc90576905)

[Результаты 11](#_Toc90576906)

[Вывод 12](#_Toc90576907)

[Эксперимент 3. 12](#_Toc90576908)

[Результаты 12](#_Toc90576909)

[Вывод 14](#_Toc90576910)

Постановка задачи.

Требуется написать программу приближения таблично заданной **в кольце** функции f методом **билинейных сплайнов.**

Входные и выходные параметры подпрограммы-функции, реализующей непосредственно алгоритм метода:

*Входные параметры*: таблично заданная функция f.

*Выходные параметры*: таблица значений приближающей функции и графики.

Теоретический материал.

Задача интерполяции функций.

Во многих физических и инженерных задачах приходится сталкиваться с потребностью интерполяции функций. Например, зачастую показатели приборов дискретны и не позволяют сделать вывод о той или иной зависимости значений от параметров.

В подобной ситуации в дело вступает интерполяция.

Многомерная интерполяция.

Интерполяция часто применяется для приближения функций двух переменных. Сформулируем задачу, стоящую перед нами в общем случае.

Пусть существует функция , заданная на некоем прямоугольнике таблицей значений

Тогда интерполяционный многочлен , заданный так, чтобы

принимает следующий вид:

где

При этот многочлен становится билинейным. Формула становится следующей:

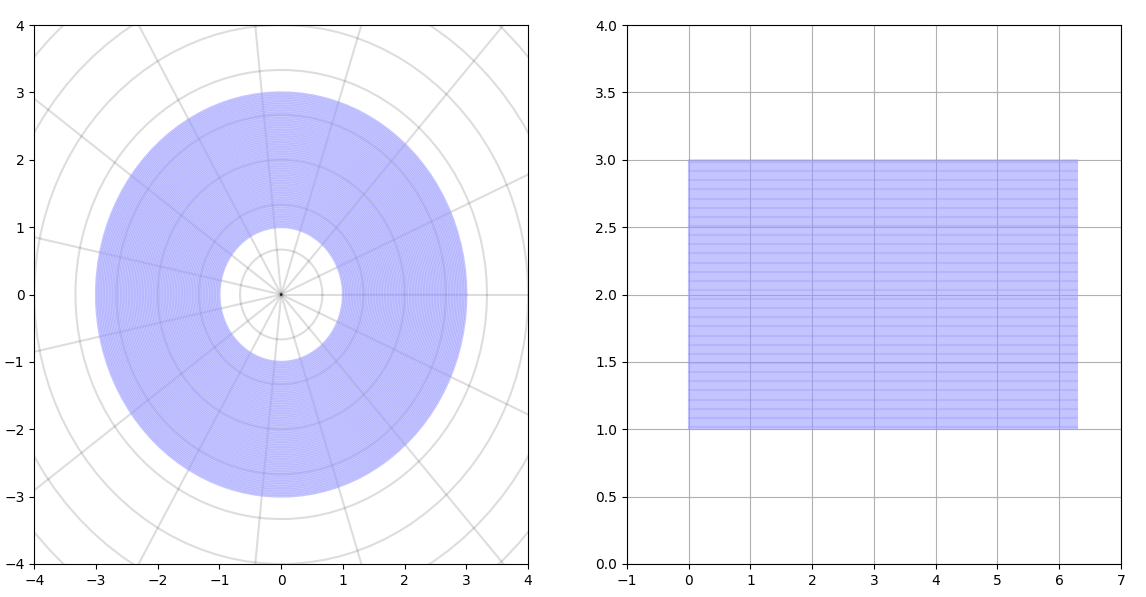
Эту формулу и будем использовать.

Полярные координаты.

В нашей задаче мы имеем область, заданную кольцом. Тем не менее, как указано в прошлом пункте, условием применения формулы билинейной интерполяции является прямоугольная таблица значений. Чтобы полностью покрыть кольцевую область без ее дискретизации, оформим переход к полярным координатам. При добавлении координаты Z такая система координат становится цилиндрической.

Имеем точку в декартовых координатах. Пусть также . Тогда:

На плоскости OXY область в виде кольца переходит в прямоугольник:



Теперь полученную прямоугольную область можно разбивать на прямоугольную сетку и применять метод билинейных многочленов. Но в конце, чтобы получить результат для декартовых координат, придётся к ним вернуться следующим образом:

Построение тестового примера.

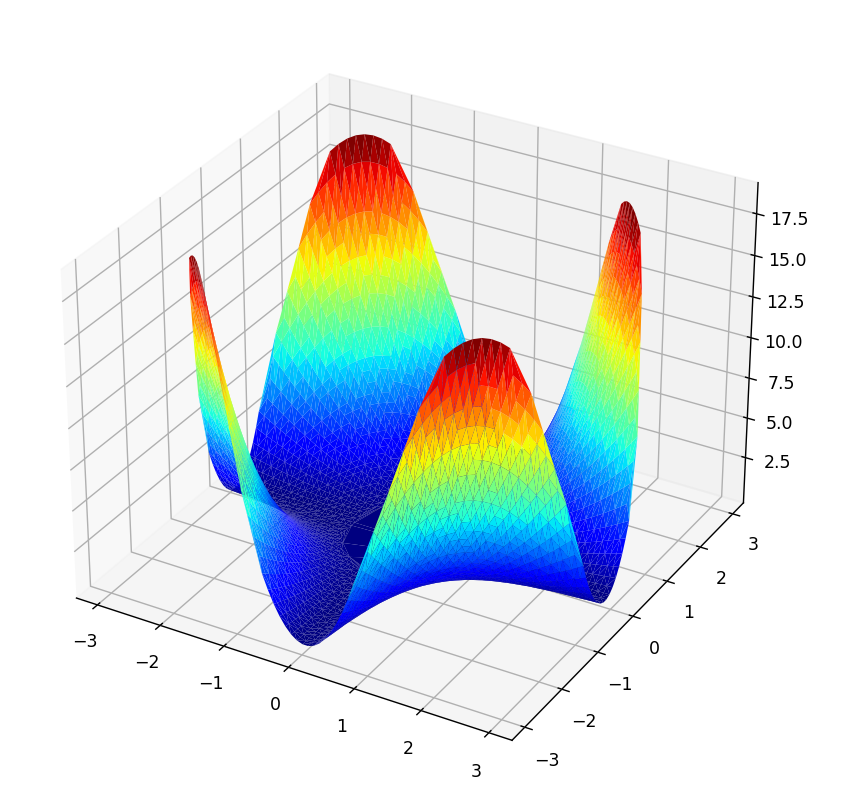
Упрощение задачи.

Входным параметром нашей задачи является таблично заданная функция. Это значит, что на вход подаётся множество точек, по которому мы и будем производить интерполирование. Конечно, в самом общем случае расположение точек может быть совершенно случайным. Тогда перед нами встаёт дополнительная задача: разбиение спроецированного на плоскость ОХУ множества точек на непересекающиеся четырёхугольники, а также трансформация каждого такого четырёхугольника в прямоугольник с помощью какого-нибудь линейного оператора.

Поэтому решим частный случай задачи: пусть значения подаются на вход в точках, удобных нам (как мы уже выяснили, эти точки будут располагаться в узлах прямоугольной сетки при переходе к полярным координатам).

Тут стоит отметить, что так как нас интересует непрерывная функция, то начальный угол нужно включить повторно в конце отрезка, чтобы полученная интерполяция замкнулась.

Также для удобства тестирования зададим координату Z этих точек на какой-нибудь аналитической поверхности. Это упростит зрительный просмотр результатов по сравнению с, например, случайным выбором координаты Z, а также позволит считать погрешность.

Выбор функции.

Рассмотрим следующую функцию:

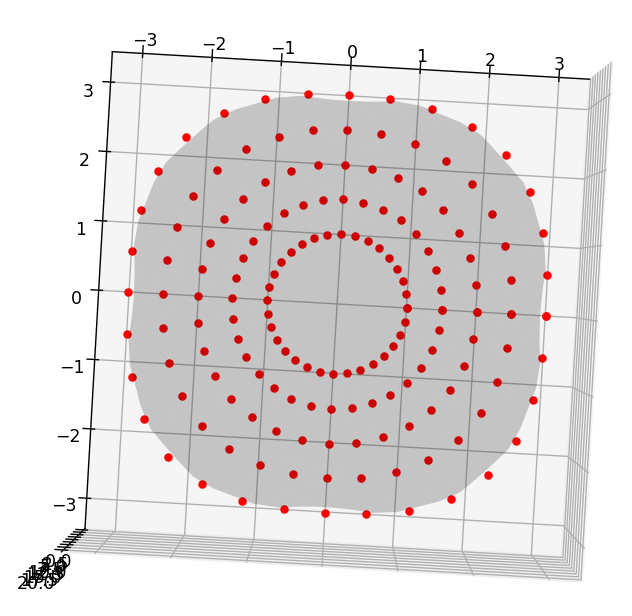
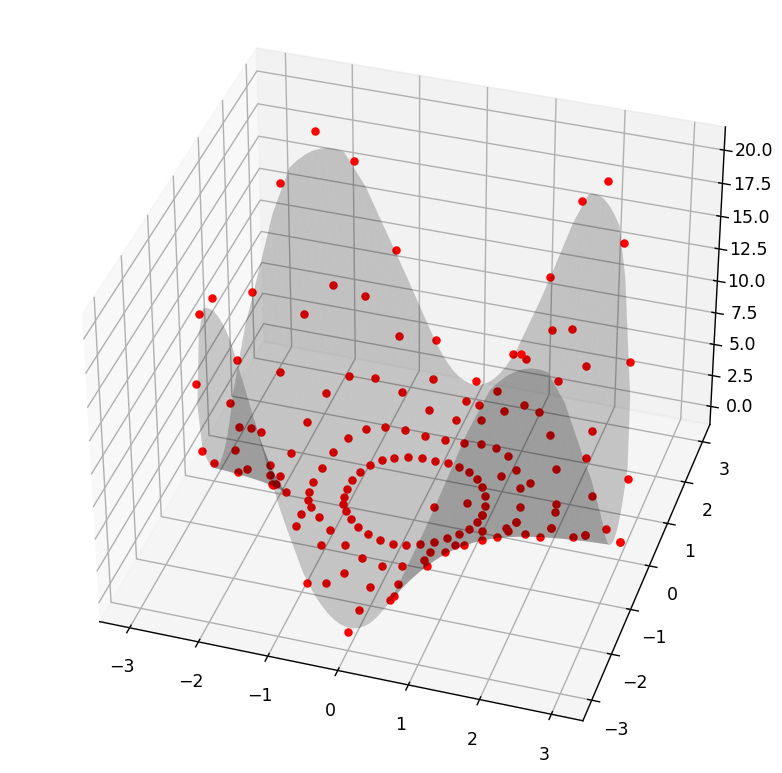
на кольце .

Эта функция имеет достаточно резкий рост при удалении от центра от осей, что поможет немного точнее понять погрешность сверху.

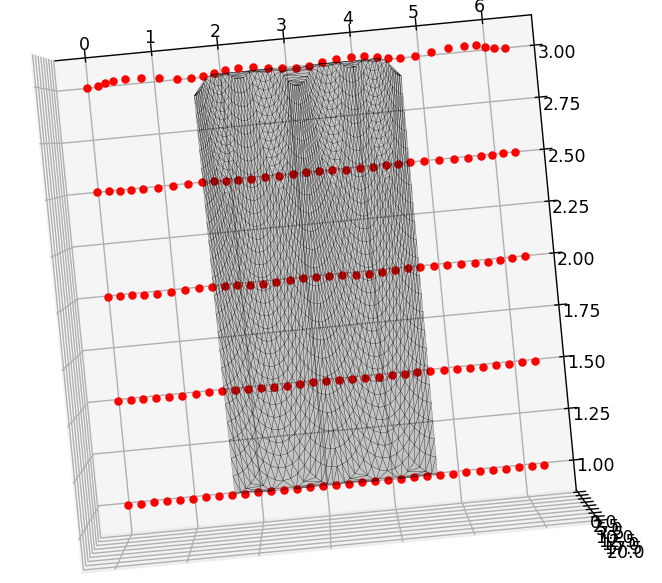
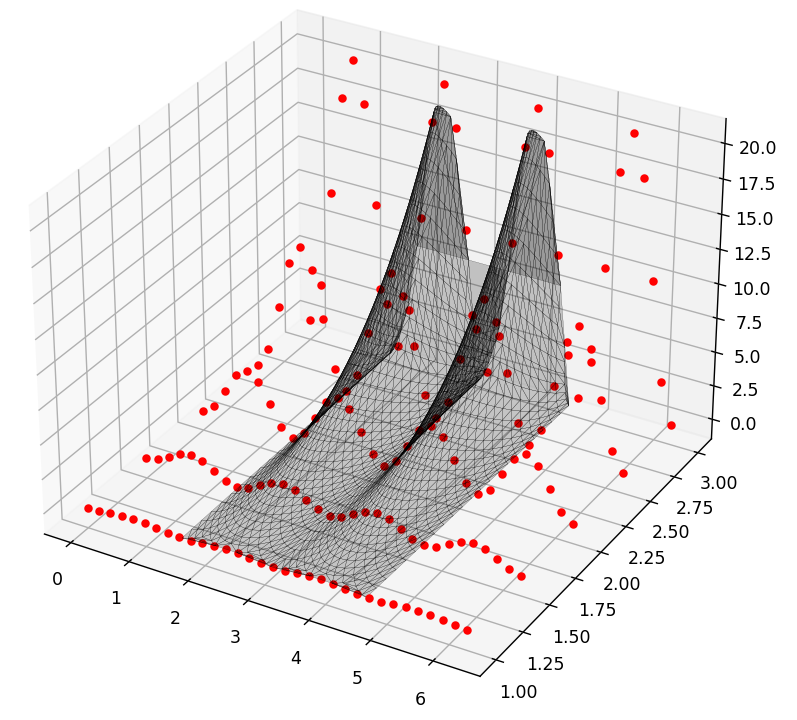
Разбиение тестового примера.

Возьмём следующее разбиение: по углу разбиваем область на 32 отрезка, а по радиусу – на 4.

На рисунке справа видно, что при просмотре в декартовых координатах узлы разбиения попадают на пересечение 32 лучей, исходящих из центра и 5 окружностей. На рисунке слева видно, как эти точки расположились непосредственно на графике интерполируемой функции. Точки разбиения отмечены красным, сама исходная функция – серым.



При просмотре в полярных координатах эти же самые точки выстраиваются в прямоугольную сетку, необходимую по условиям метода. Слева видно, как эти точки расположились на графике функции в полярных координатах.

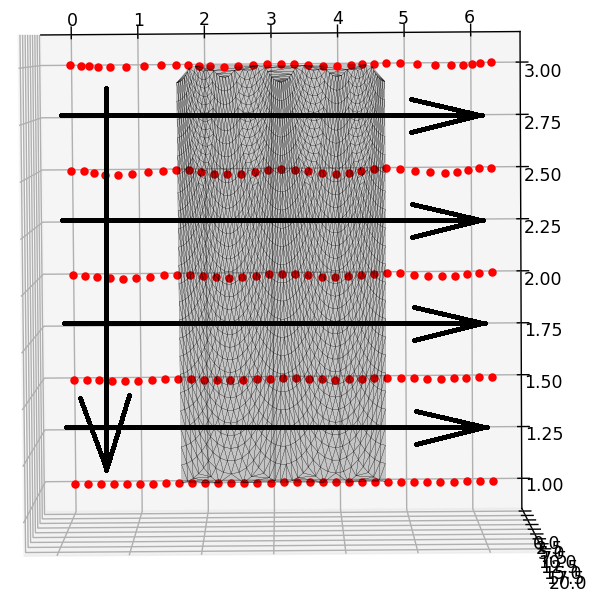


Для решения задачи разбиения, получения точек в этом и последующих этапах решения будем использовать средства языка программирования Python, а также популярные библиотеки numpy и matplotlib (последняя используется для вывода графиков, в том числе представленных на рисунках выше). Сохраним значения в трёхмерный массив.

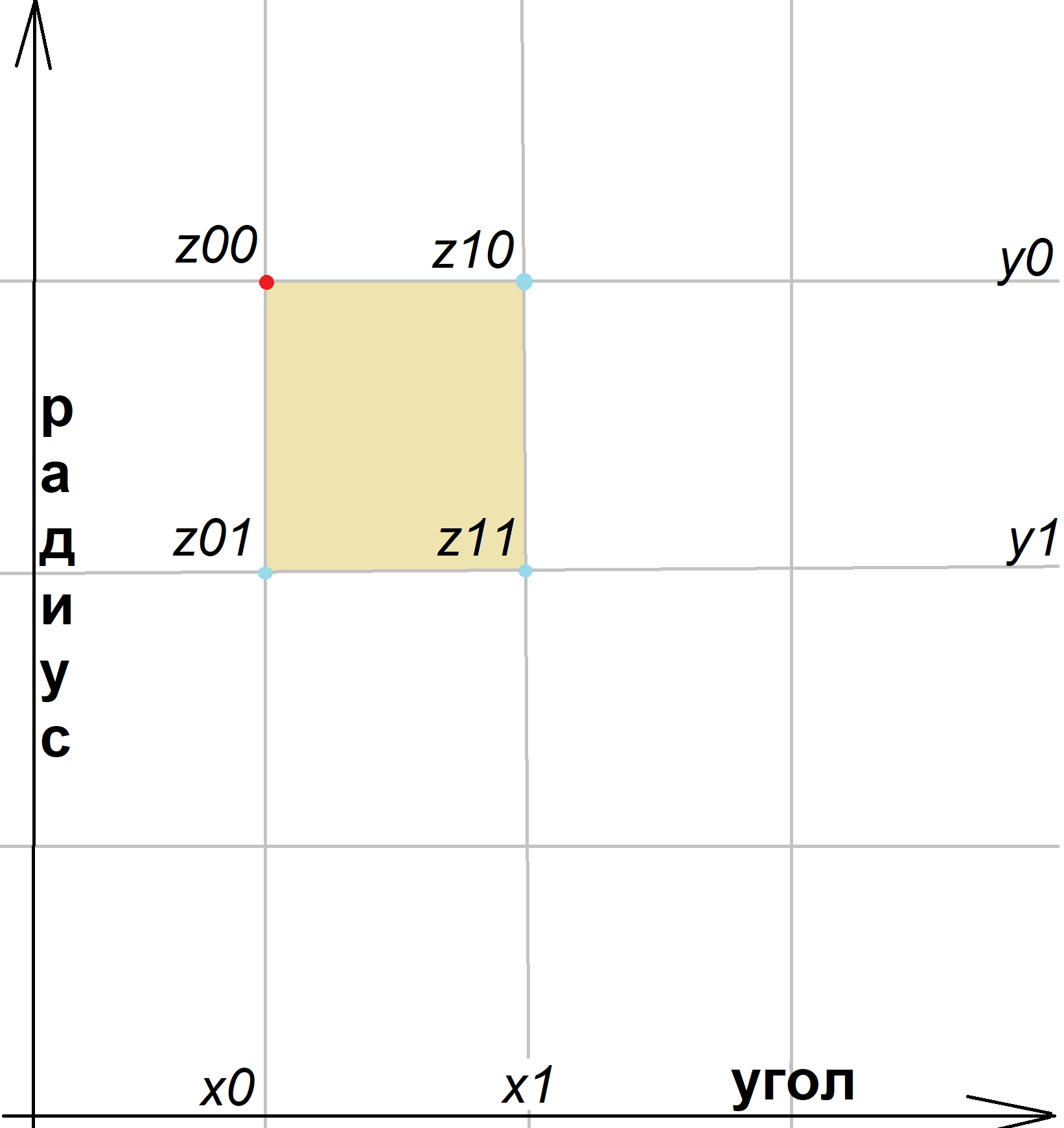
Теперь, когда мы обладаем необходимой таблично-заданной функцией (множество красных точек), можем приступать к интерполяции.

Построение модели вычислений.

Получение функций для каждой области.

Пусть на вход нам уже дана таблица значений *table* размерностью 33х5х3. Нам необходимо для каждой из прямоугольных (на OXY) областей задать функцию, которая будет приближать промежуточные значения на этом прямоугольнике. Формула для самой функции есть ни что иное, как формула билинейного многочлена, приведённая [выше](#_Toc90576891). Осталось для каждой области получить необходимые коэффициенты и структурировать хранение полученных функций.

Итак, будем итерироваться по точкам, начиная с верхней левой. Так как они заданы трёхмерным массивом, пробегаемся по первым двум координатам, чтобы получить всевозможные точки в порядке, указанном на рисунке справа. Это делается двумя вложенными циклами.

Далее для каждой точки будем вычислять формулу для области, лежащей правее и ниже текущей точки. Красным обозначена текущая точка, а голубым – другие точки из таблицы, которые мы задействуем для вычисления по формуле.

Так как мы используем точки, расположенные правее и ниже текущей, то необходимо скорректировать границы итерирования по таблице точек до значений разбиений - 1. В Python эти циклы выглядят так:

for i in range(dot\_table.shape[0] - 1):  
 ...  
 for j in range(dot\_table.shape[1] - 1):  
 ...

Теперь получаем необходимые коэффициенты. Если текущая точка имеет в таблице координаты то:

Эти данные можно подставлять в формулу билинейного многочлена. Таким образом, получаем формулу для текущей области и сохраняем ее в какую-то структуру (например, тоже табличную).

Сохранение.

Сохранять полученные функции будем в таблицу функций. Причем коэффициенты функции области в таблице будут совпадать с коэффициентами верхней левой точки области.

Результаты расчётов тестового примера.

Произведём расчёты [тестового примера](#_Toc90549649). Запустим программу со следующими начальными данными:

def F(x, y):  
 return x\*\*2 \* y\*\*2  
  
R1 = 1  
R2 = 3  
  
A\_steps = 32  
R\_steps = 4  
  
graph = True

Здесь мы задаём интерполируемую функцию , границы кольца и шаг интерполяции. Флажок graph определяет, будут ли выведены графики.

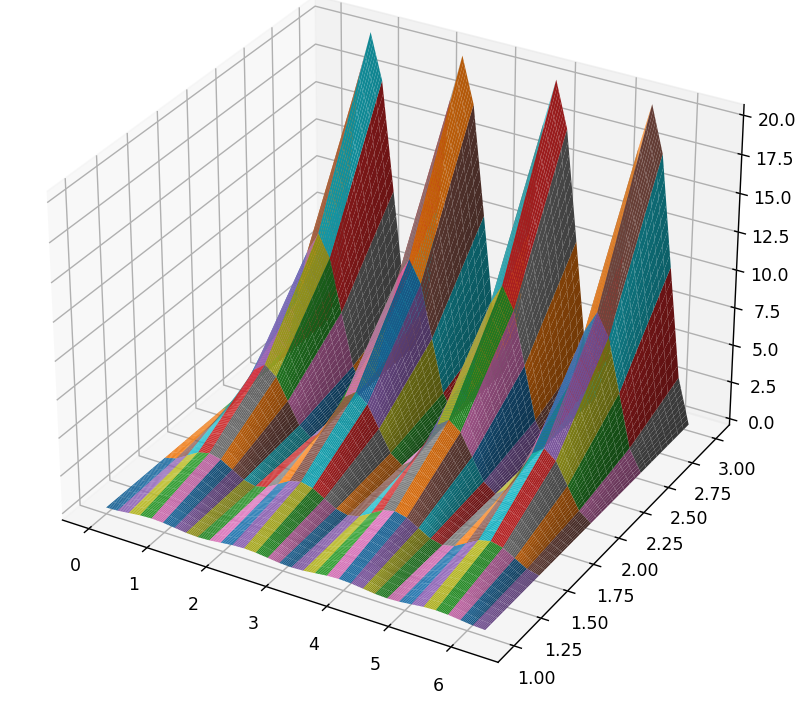
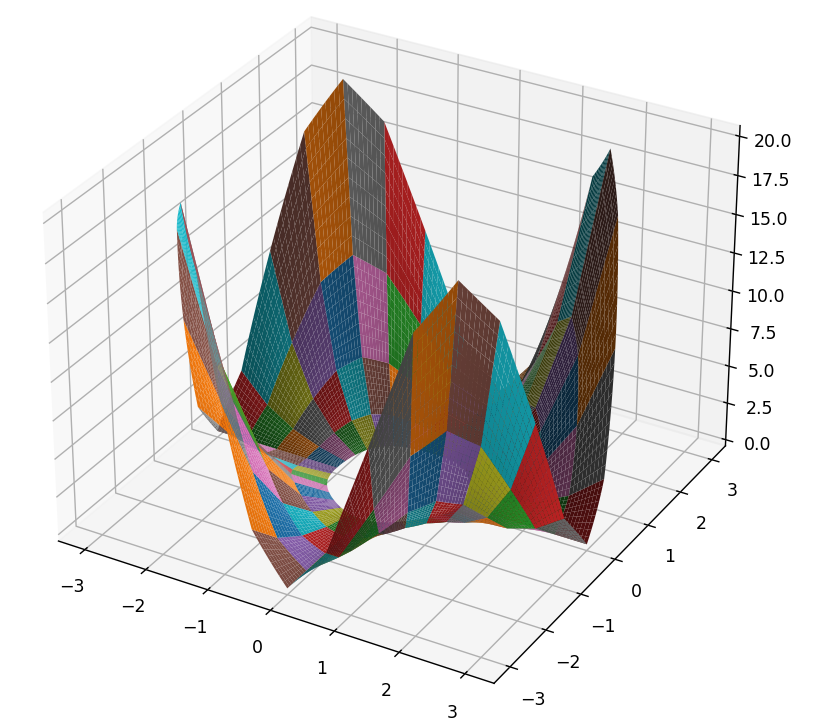
Первые 4 полученных графика уже были выше выведены как иллюстрация к построению тестового примера. Далее получаем следующие графики:

Рисунок 1.

Аппроксимированные области в цилиндрической системе координат (ЦСК)

Рисунок 2.

Аппроксимированные области в декартовой системе координат (ДСК)

Далее в декартовых координатах задаём тестовые точки. Пусть это будет 50х50 точек в диапазоне , причём выберем только точки, попадающие в заданное примером кольцо.

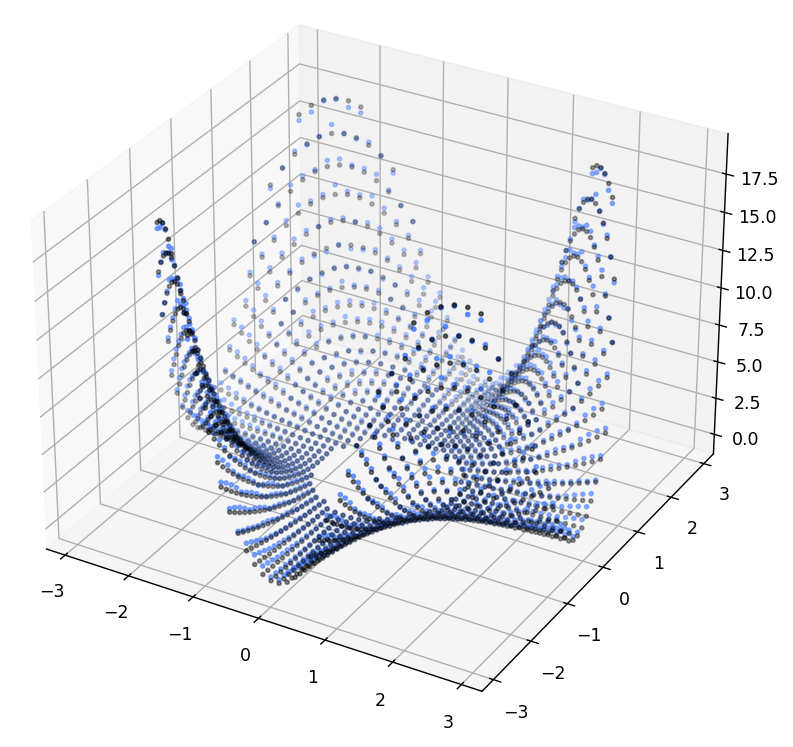


Рисунок 3.

Сравнение поточечной выборки точек в ДСК (синие точки найдены по аппроксимации, чёрные – по исходной функции)

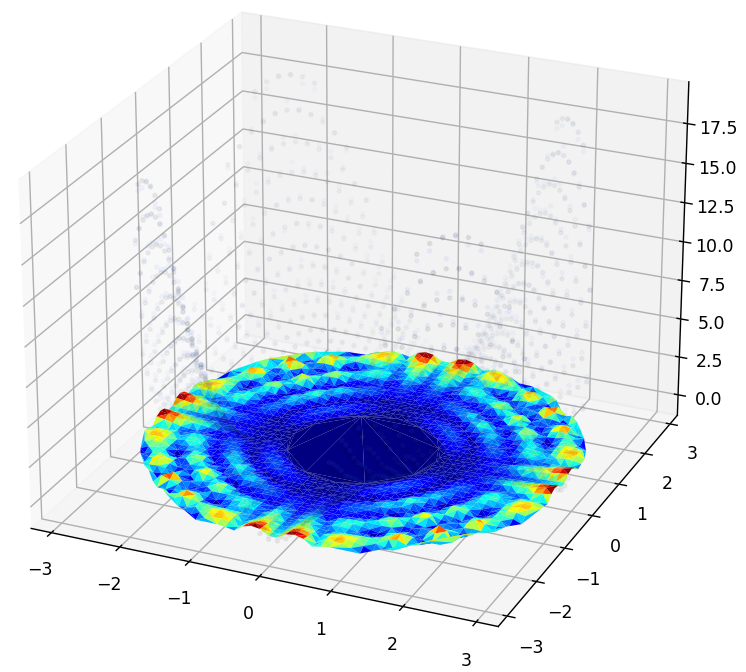


Рисунок 4.

Реальная погрешность аппроксимации в ДСК (красные области – области максимальной погрешности, синие – области минимальной)

Также рассмотрим вывод полученных результатов в виде таблицы. Чтобы не выводить все точек, выведем только каждую сотую. Но при нахождении максимальной погрешности, конечно, используем все 2500 точек:

F(x,y) | MineF(x,y) | Delta(x,y)

===============================================

5.2455 | 5.5613 | 0.315889602490245

0.1923 | 0.4897 | 0.297383394662482

0.3825 | 0.5048 | 0.122298947360051

7.4672 | 7.8950 | 0.427878982312272

0.1858 | 0.2044 | 0.018533833046697

3.6977 | 3.7007 | 0.002985843633078

3.6117 | 3.7673 | 0.155521170192095

1.9132 | 2.0442 | 0.131010958253298

0.2561 | 0.7568 | 0.500735347650229

0.6495 | 1.1154 | 0.465922688495898

3.1436 | 3.4931 | 0.349534625213455

1.6724 | 1.8807 | 0.208259228783999

7.7150 | 7.9424 | 0.227372464500593

5.4886 | 5.8982 | 0.409599706286604

1.9970 | 2.0432 | 0.046134677933252

1.3942 | 1.5876 | 0.193410421404224

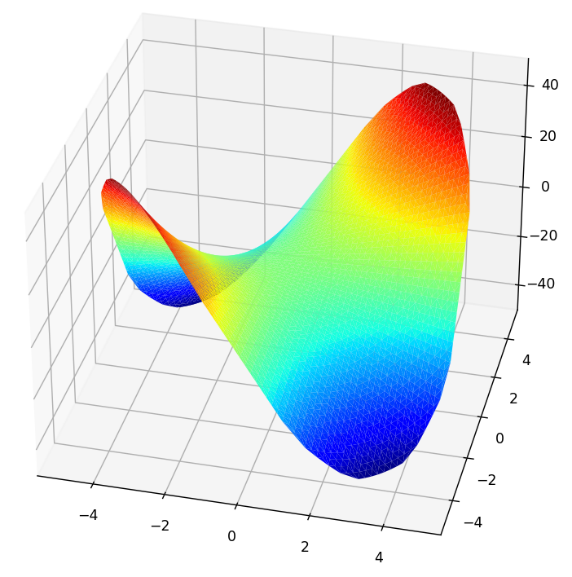
1.1573 | 1.3547 | 0.197372757158233

Максимум погрешности: 0.6501849368437238

Получили максимальную погрешность ~0.65 для тестовой задачи. В целом, для функции, чья производная () растёт в некоторых областях квадратично это неплохой результат.

Вычислительные эксперименты.

Проведём ещё несколько характерных вычислительных экспериментов и рассмотрим полученные результаты.

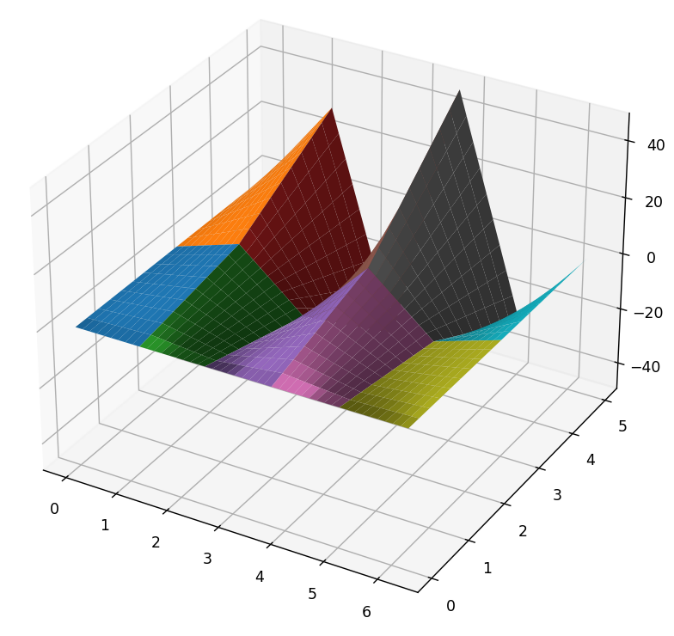
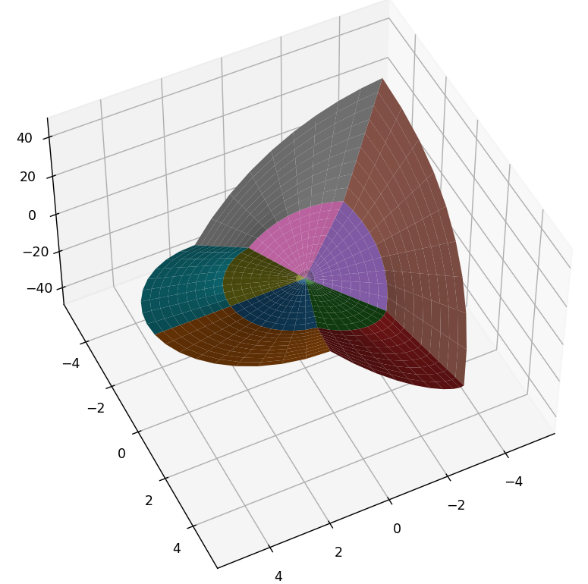
Эксперимент 1.

Функция:

Область:

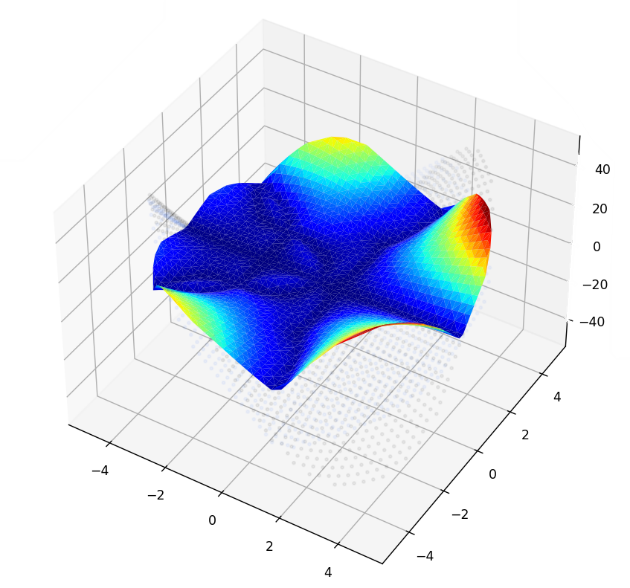
Разбиение:

Результаты



Аппроксимация в ДСК

Аппроксимация в ЦСК

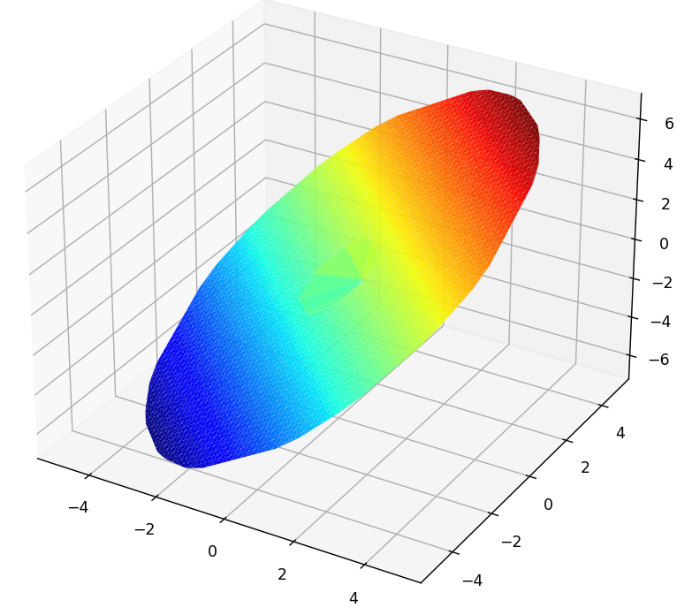


Итоговая погрешность

Максимум погрешности: 32.33482148184983

Вывод

На данном примере можем наблюдать, что при маленьких отрезках разбиения погрешность выходит достаточно большая.

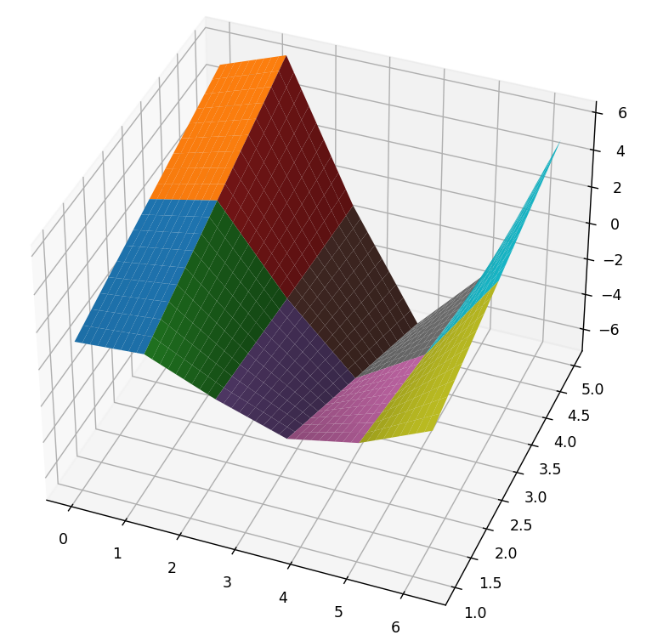
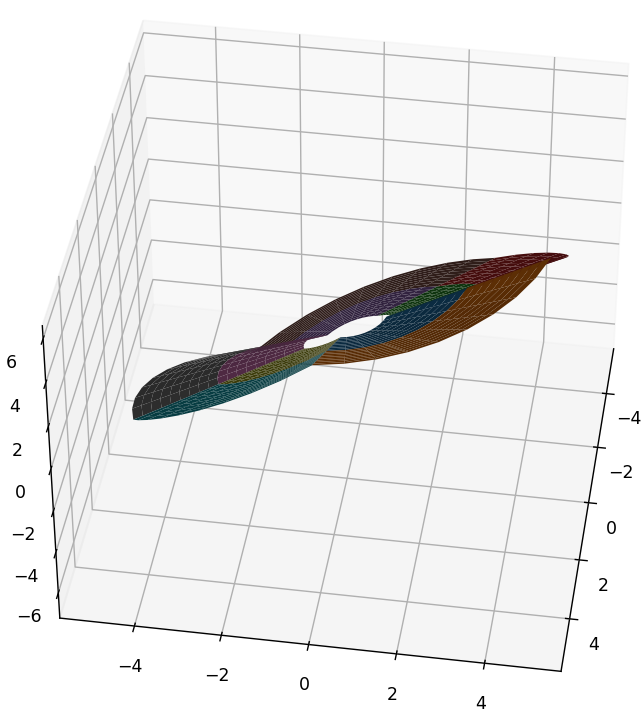
Эксперимент 2.

Функция:

Область:

Разбиение:

Результаты



Аппроксимация в ДСК



Аппроксимация в ЦСК

Итоговая погрешность

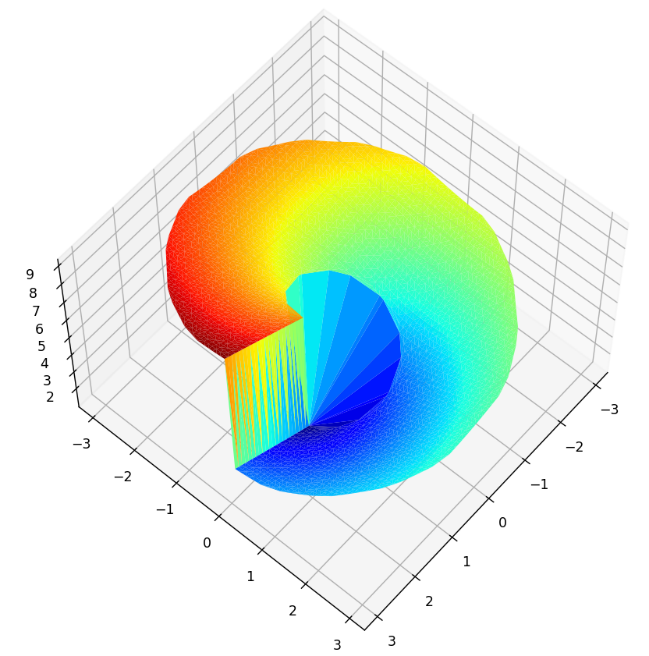
Максимум погрешности: 1.3233091086346622

Вывод

На первый взгляд может показаться, что так как исходная функция является плоскостью, то она должна интерполироваться идеально, так как билинейная интерполяция предоставляет функцию плоскости для каждой области. Тем не менее, как видим на графике аппроксимации в ДСК, полученные области вовсе не являются плоскими, в связи с чем мы наблюдаем такую неоднородную и большую погрешность на соответствующем графике.

Это связано с тем, что аппроксимацию плоскостями мы применяем *в системе цилиндрических координат*, что видно на графике ЦСК. Но при переходе обратно в декартовы координаты эта плоскость перестаёт быть плоской и превращается в какие-то синусоиды. Из-за этого даже на плоском графике мы наблюдаем «необычные» волны.

Из этого рассуждения можно сделать предположение, что если бы мы аппроксимировали функцию, являющуюся плоскостью именно в цилиндрических координатах, то погрешность была бы очень мала. Что ж, проведём такой эксперимент.

 Эксперимент 3.

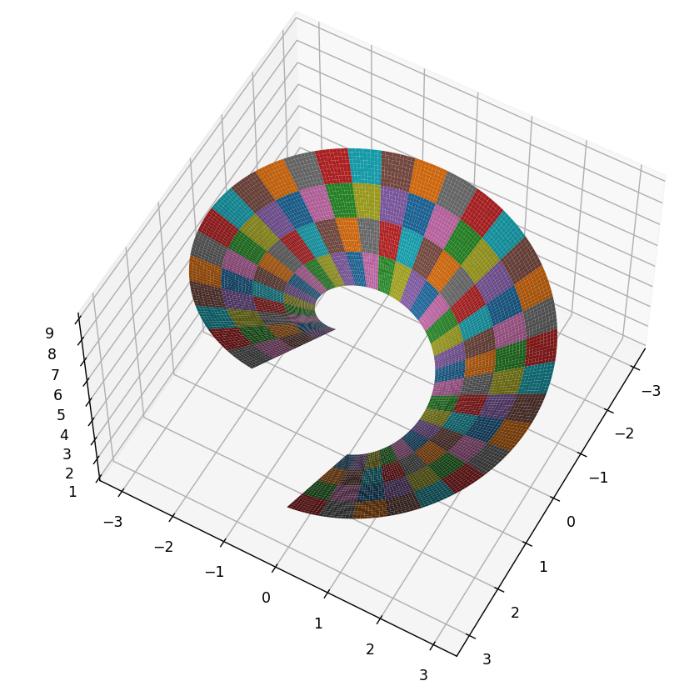
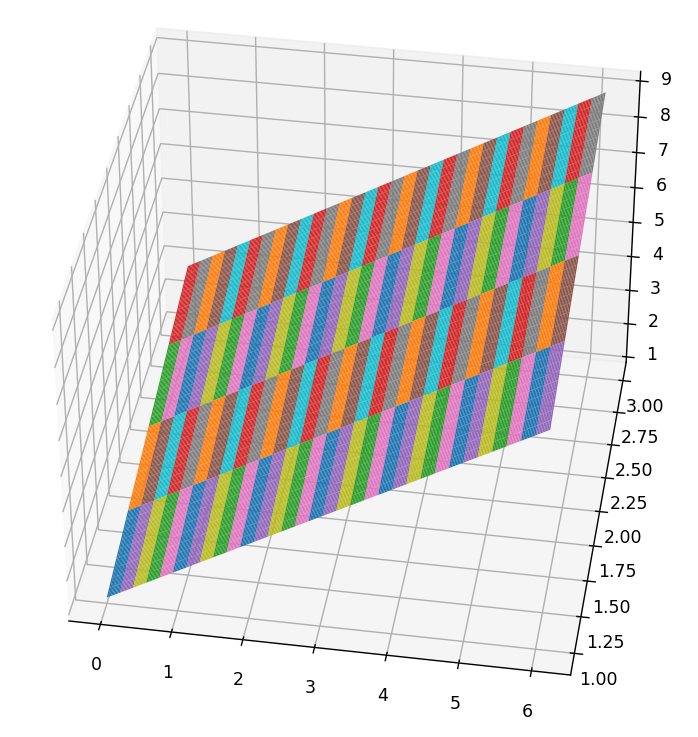
Функция: ,

т.е.

Область:

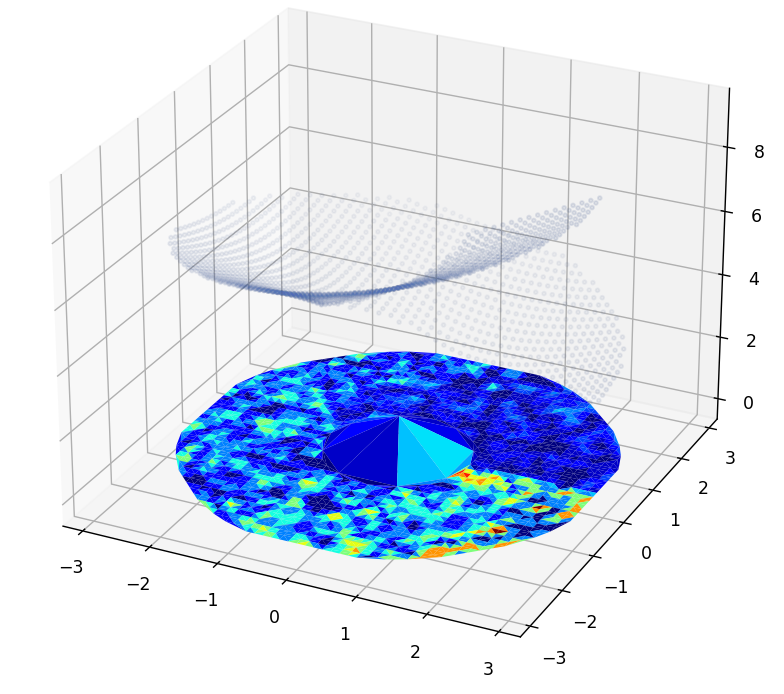
Разбиение:

Результаты



Аппроксимация в ЦСК

Аппроксимация в ДСК



Итоговая погрешность

F(x,y) | MineF(x,y) | Delta(x,y)

===============================================

6.3970 | 6.3970 | 0.000000000000001

5.4596 | 5.4596 | 0.000000000000000

5.0347 | 5.0347 | 0.000000000000000

6.2648 | 6.2648 | 0.000000000000000

4.3686 | 4.3686 | 0.000000000000000

4.2053 | 4.2053 | 0.000000000000001

6.9076 | 6.9076 | 0.000000000000002

4.3569 | 4.3569 | 0.000000000000000

4.3986 | 4.3986 | 0.000000000000001

7.4785 | 7.4785 | 0.000000000000000

7.6803 | 7.6803 | 0.000000000000002

2.6741 | 2.6741 | 0.000000000000000

3.7735 | 3.7735 | 0.000000000000001

7.6624 | 7.6624 | 0.000000000000000

2.3672 | 2.3672 | 0.000000000000000

8.2441 | 8.2441 | 0.000000000000000

2.7156 | 2.7156 | 0.000000000000000

Максимум погрешности: 3.552713678800501e-15

Вывод

Как и ожидалось в предположении, мы получили очень маленькую погрешность для данного случая. Несмотря на то, что получившаяся функция разрывная в декартовой системе координат, программе удалось решить данную задачу. Это связано с тем, что разрыв происходит на луче , который мы из построения метода считаем дважды. Но при и значение функции будет разным, так как оно напрямую зависит от угла , в отличие от предыдущих экспериментов, где функция была задана в декартовых координатах. Поэтому разрыв наблюдается и при разбиении билинейным многочленом, а аппроксимация на последнем отрезке точна.

В общем случае аппроксимация разрывной функции физически возможна, но приводит к непредсказуемым результатам.

Итог.

Подводя итог, можно констатировать, что интерполяция билинейными многочленами была проведена успешно. Полученные в ходе экспериментов погрешности в целом невелики и вполне логичны для данного метода. Стоит особенно отметить, что из-за перехода к цилиндрическим координатам погрешность функции тем меньше, чем стабильнее ведёт себя данная функция при переходе в полярные координаты. Таким образом, конкретно для полученной программы наиболее подходят функции, имеющие тригонометрические зависимости.

Код программы с комментариями.

Литература.

Амосов А. А., Дубинский Ю. А., Копченова Н. В. Вычислительные методы: Учебное пособие. - 4-е изд., стер. - СПб.: Издательство Лань, 2014.

Емелин А. Полярные координаты – подробное пособие для чайников // Mathprofi : [сайт]. [2010]. URL: <http://mathprofi.ru/poljarnye_koordinaty.html>