Лабораторная работа №6

Воробьев А.О. НПМмд-02-23

Содержание

# 1 Цель работы

Научиться работать в Octave с пределами, последовательностями, рядами и численным интегрированием.

# 2 Задание работы

Выполнить лабораторную работу и сделать отчет по лабораторной работе в форматах md, docx и pdf.

# 3 Теоретичсекое введение

В octave можно реализовать пределы, последовательности и ряды, а так же считать определенный интеграл.

* Предел — одно из основных понятий математического анализа, на него опираются такие фундаментальные разделы анализа, как непрерывность, производная, интеграл, бесконечные ряды и др. Различают предел последовательности и предел функции.
* Последовательность — это пронумерованный набор каких-либо объектов, среди которых допускаются повторения, причём порядок объектов имеет значение. Нумерация чаще всего происходит натуральными числами.
* Ряд, называемый также бесконечная сумма — одно из центральных понятий математического анализа. В простейшем случае ряд записывается как бесконечная сумма чисел:
* (иногда нумерацию слагаемых начинают не с 1, а с 0)
* Определённый интеграл — одно из основных понятий математического анализа, один из видов интеграла. Определённый интеграл является числом, равным пределу сумм особого вида (интегральных сумм). Геометрически определённый интеграл выражает площадь «криволинейной трапеции», ограниченной графиком функции.
* Квадратура (лат. quadratura, придание квадратной формы) — математический термин, первоначально обозначавший нахождение площади какой-либо фигуры или поверхности. То есть, мы разбиваем/приближаем площадь под графиком к набору прямоугольников одинаковой ширины.
* правило средней точки:
* Аппроксимация f в середине интервалов дает f (a + Δx / 2) для первого интервала, для следующий f (a + 3Δx / 2), и так далее до f (b - Δx / 2). Суммирование площадей дает:

# 4 Выполнение лабораторной работы

1. Создаем каталог для работы в папке laboratory. (mkdir) (fig. 1)

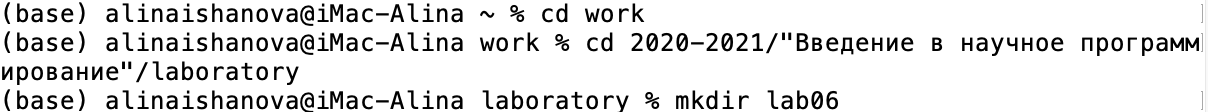


Figure 1: подготовка к лабораторной работе

1. Начинаем сессию журналирования. (fig. 2)

начало журналирования

Figure 2: начало журналирования

## 4.1 Пределы, последовательности и ряды

### 4.1.1 Предел

1. Для расчета предела функции, нужно сначала определить функцию. (fig. 3)

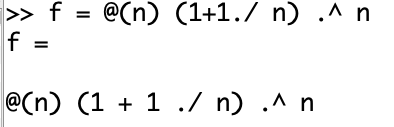


Figure 3: определение функции

1. Создаем индексную переменную. (fig. 4)

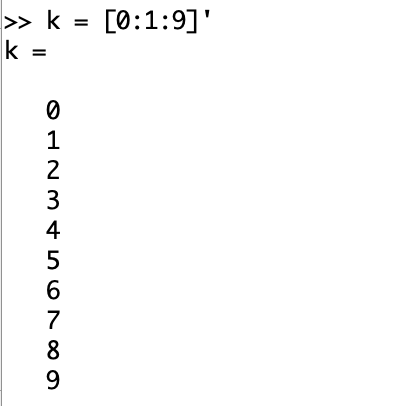


Figure 4: индексная переменная

1. Берем степени 10, которые будут входными значениями. (fig. 5)

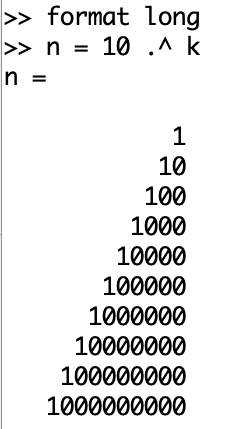


Figure 5: определение входных значений

1. Оцениваем . (fig. 6)

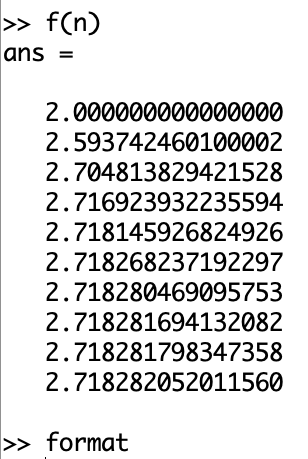


Figure 6: оценивание функции

### 4.1.2 Частичные суммы

1. Задаем вектор входных значений/индексов. (fig. 7)

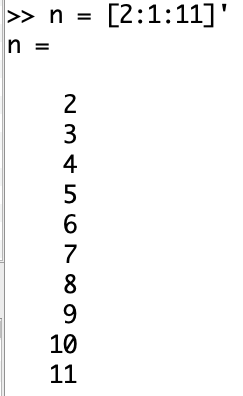


Figure 7: определение значений индексов

1. Высчитываем члены ряда. (fig. 8)

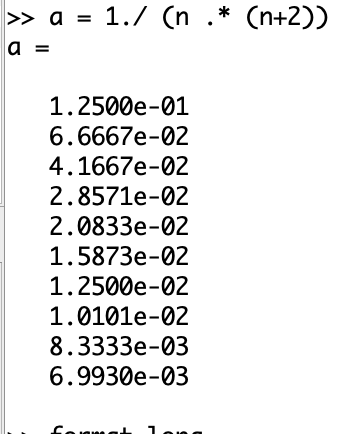


Figure 8: вычисление членов ряда

1. Пишем цикл для суммирования членов ряда, получаем суммы 2 член, 2-3 член, 3-4 член и тд. до 2-11 член. (fig. 9)

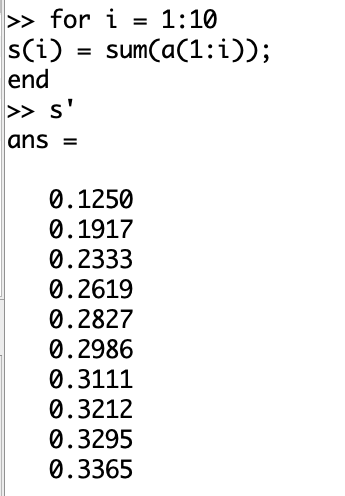


Figure 9: вычисление частичных сумм

1. Строим график с полученными данными. (fig. 10 и fig. 11)

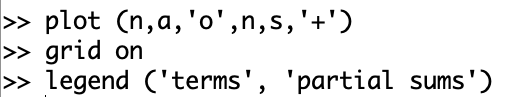


Figure 10: команды построения графика

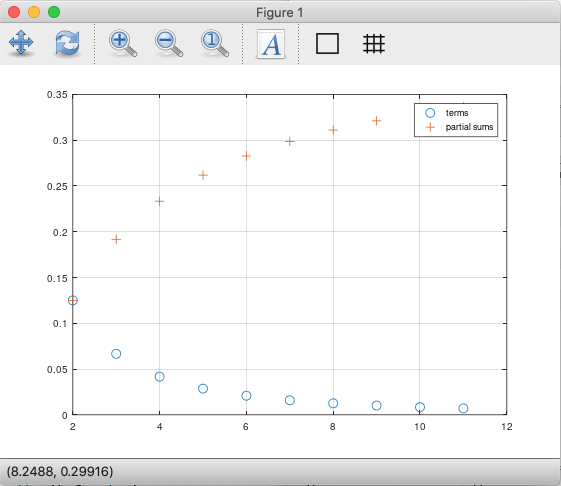


Figure 11: график значений членов ряда и частичной суммы от 2 до них

### 4.1.3 Сумма ряда

1. Задаем вектор входных значений/индексов. (fig. 12)

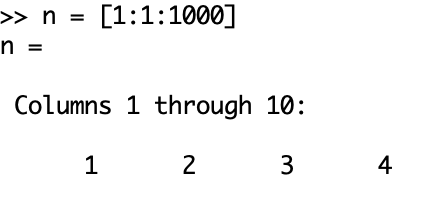


Figure 12: определение значений индексов (часть)

1. Задаем гармонический ряд. Задаем вектор входных значений/индексов. (fig. 13)

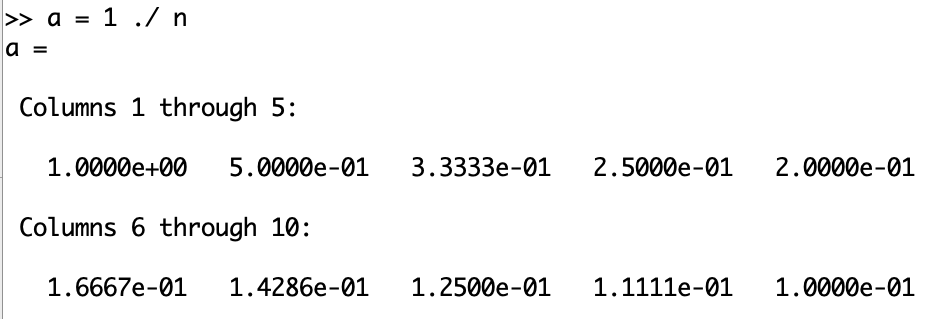


Figure 13: определение значений членов ряда (часть)

1. Вычисляем сумму ряда. Задаем вектор входных значений/индексов. (fig. 14)

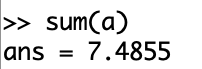


Figure 14: определение значений индексов

## 4.2 Численное интегрирование

### 4.2.1 Вычисление интегралов

1. Определяем функцию, чей интеграл мы будем считать, и считаем определенный интеграл командой quad. (fig. 15)

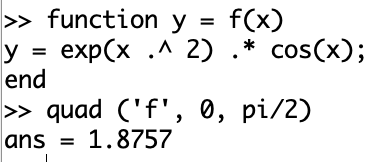


Figure 15: задание функции и рассчет определенного интеграла от 0 до

1. Рассчитаем определенный интеграл, используя анонимную функцию. (fig. 16)

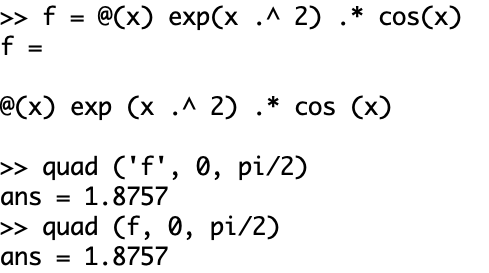


Figure 16: задание анонимной функции и рассчет определенного интеграла

### 4.2.2 Аппроксимирование суммами

1. Создаем файл Octave. (fig. 17)

создание файла midpoint.m

Figure 17: создание файла midpoint.m

1. В нем записываем реализацию метода средней точки с помощью цикла. (fig. 18)

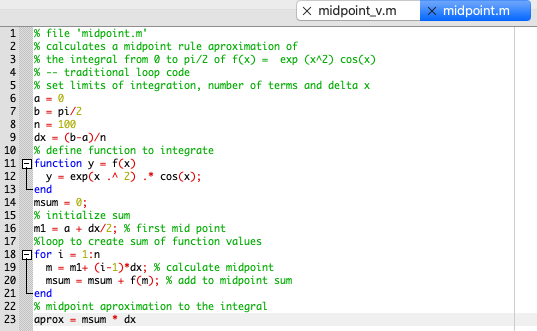


Figure 18: содержимое файла midpoint.m

1. Запускаем файл. (fig. 19)

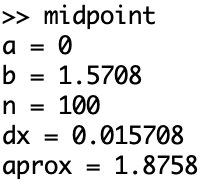


Figure 19: работа midpoint.m

1. Создаем второй файл Octave. (fig. 20)

создание файла midpoint_v.m

Figure 20: создание файла midpoint\_v.m

1. В нем записываем векторизованную реализацию метода средней точки. (fig. 21)

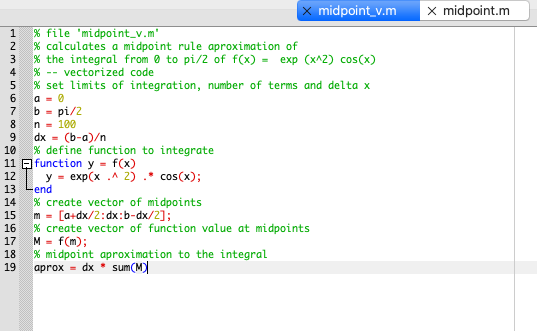


Figure 21: содержимое файла midpoint\_v.m

1. Запускаем файл. (fig. 22)

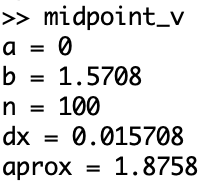


Figure 22: работа midpoint\_v.m

1. Для того, чтобы увидеть разницу между двумя этими подходами, засекаем время выполнения файлов. (fig. 23)

|  |
| --- |
| сравнение midpoint.m и midpoint\_v.m по времени |

Figure 23: сравнение midpoint.m и midpoint\_v.m по времени

Видим, что векторизированный код работает на порядок быстрее.

# 5 Вывод

В ходе выполнения работы мы научились считать пределы и частичные суммы рядов, считать определенный интеграл встроенной окмандой quad и методом средней точки, а так же увидели разницу в скорости работы трандиционного кода (с циклами) и векторизированного кода.

# 6 Библиография

1. *Lachniet J.* Introduction to GNU Octave. 2nd ed. 2019. pp. 39-45
2. wikichi.ru:Cумма Римана (https://wikichi.ru/wiki/Riemann\_sum)
3. Wikipedia: Предел (https://ru.wikipedia.org/wiki/%D0%9F%D1%80%D0%B5%D0%B4%D0%B5%D0%BB\_(%D0%BC%D0%B0%D1%82%D0%B5%D0%BC%D0%B0%D1%82%D0%B8%D0%BA%D0%B0))