

РОССИЙСКИЙ УНИВЕРСИТЕТ ДРУЖБЫ НАРОДОВ

Факультет физико-математических и естественных наук

Кафедра прикладной информатики и теории вероятностей

ОТЧЕТ

ПО ЛАБОРАТОРНОЙ РАБОТЕ №6

дисциплина: Научное программирование

Студент: Романова Александра

Группа: НПМмд-02-20

МОСКВА

2020 г.

Цель

Ознакомление с некоторыми операциями в среде Octave для работы с пределами, последовательностями и рядами, ознакомление с численным интегрированием.

Выполнение работы

Пределы, последовательности и ряды

Рассмотрим предел

$$\lim_{n \rightarrow \infty} \left(1 + \frac{1}{n}\right)^n$$

Для определения функции используем метод анонимной функции. Мы назвали функцию f . Далее создаем индексную переменную, состоящую из целых чисел от 0 до 9. Синтаксис `[0:1:9]` создает вектор строки, который начинается с 0 и увеличивается с шагом от 1 до 9. Мы использовали операцию транспонирования потому, что наши результаты будут легче читать как векторы-столбцы. Далее возьмём степени 10, которые будут входными значениями, а затем оценим $f(n)$.

Предел сходится к конечному значению, которое составляет приблизительно 2,71828... Подобные методы могут быть использованы для численного исследования последовательностей и рядов (см. Рис.1).

```
цая папка: C:\Users\gora1
Командное окно
>> f=@(n) (1+1./n).^n
f =

@(n) (1 + 1 ./ n) .^ n

>> k = [0:1:9]'
k =

    0
    1
    2
    3
    4
    5
    6
    7
    8
    9

>> format long
>> n = 10.^k
n =

     1
    10
   100
  1000
 10000
100000
1000000
10000000
100000000
1000000000
10000000000

>> f(n)
ans =

 2.0000000000000000
 2.5937424601000002
 2.704813829421529
 2.716923932235520
```

Рис.1 Предел

Частичные суммы

Пусть a $\sum_{n=2}^{\infty} a_n$ - ряд, n -й член равен

$$a_n = \frac{1}{n(n+2)}.$$

Для этого определим индексный вектор n от 2 до 11, а затем вычислим члены. Мы будем использовать цикл *for* с индексом i от 1 до 10. Для каждого i мы получим

частичную сумму последовательности a_n от первого слагаемого до i -го слагаемого. На выходе получается 10-элементный вектор этих частичных сумм. Далее мы построим слагаемые и частичные суммы для $2 \leq n \leq 11$ (см. Рис2).

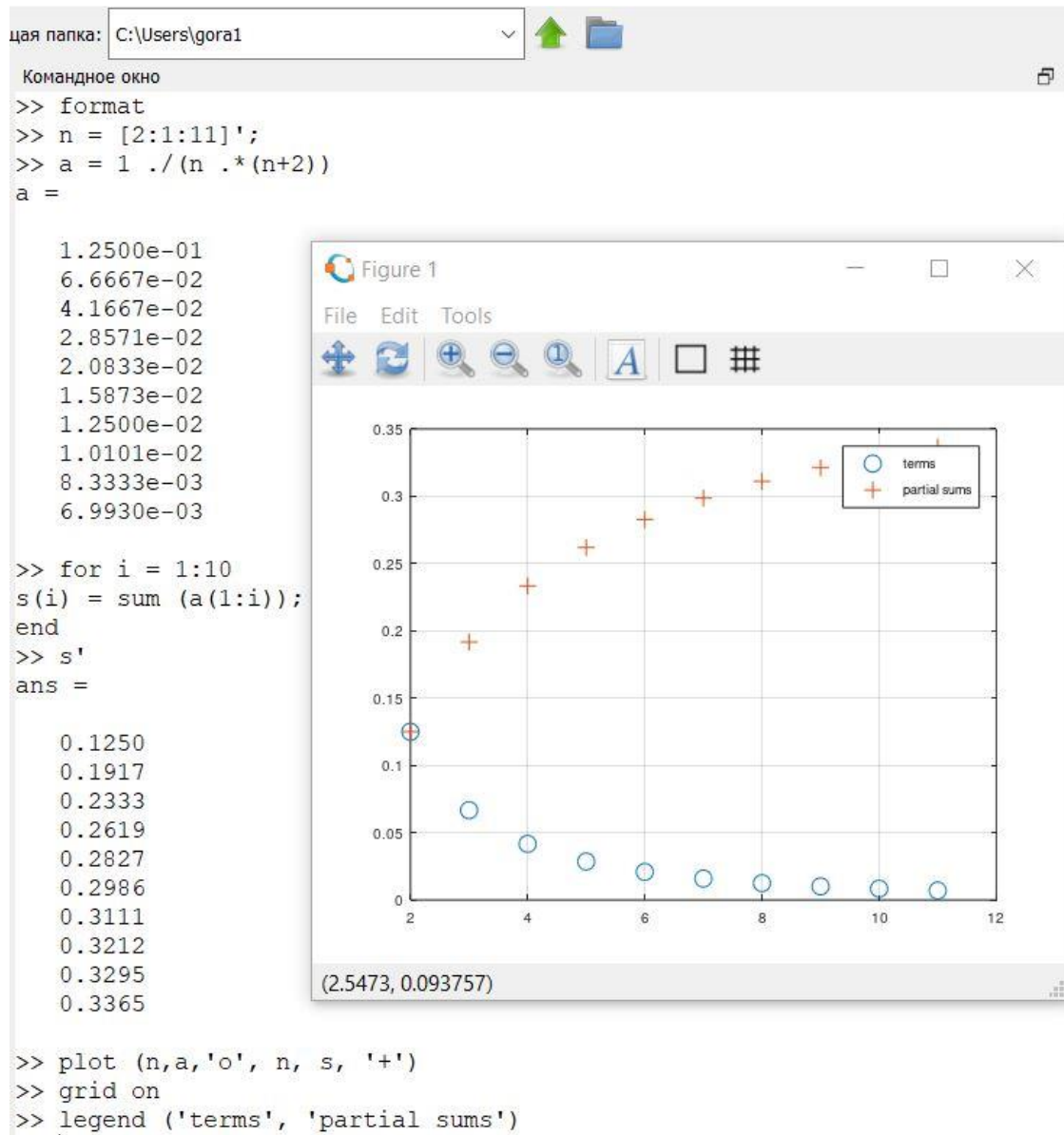


Рис.2 Частичные суммы

Сумма ряда

Найдем сумму первых 1000 членов гармонического ряда:

$$\sum_{n=1}^{1000} \frac{1}{n}.$$

Сгенерируем члены ряда как вектор, а затем возьмем их сумму(см. Рис.3).

```
щя папка: C:\Users\gora1
Командное окно
>> n = [1:1:1000];
>> a = 1 ./n;
>> sum (a)
ans = 7.4855
```

Рис.3 Сумма ряда

Вычисление интегралов

Вычислим интеграл (см.Рис.4):

$$\int_0^{\pi/2} e^{x^2} \cos(x) dx$$

```
щя папка: C:\Users\gora1
Командное окно
>> function y = f(x)
y = exp(x.^2).*cos(x);
end

>> quad ('f',0,pi/2)
ans = 1.8757
```

Рис.4 Вычисление интеграла

Аппроксимирование суммами

Напишем скрипт, чтобы вычислить интеграл

$$\int_0^{\pi/2} e^{x^2} \cos(x) dx$$

по правилу средней точки для $n = 100$. Используем цикл, который добавляет значение функции к промежуточной сумме с каждой итерацией. В конце сумма умножается на Δx . Введем код в текстовом файле и назовем его midpoint.m(см. Рис.5) и запустим его (см. Рис.6). Создадим вектор x-координат средних точек. Затем мы оцениваем f по этому вектору средней точки, чтобы получить вектор значений функции. Аппроксимация средней точки - это сумма компонент вектора, умноженная на Δx . Введем код в текстовом файле и назовем его midpoint_V.m(см. Рис.7) и запустим его (см. Рис.8).

```
midpoint.m
1 % file 'midpoint.m'
2 % calculates a midpoint rule approximation of
3 % the integral from 0 to pi/2 of f(x) = exp (x^2) cos (x)
4 % -- traditional looped code
5 % set limits of integration, number of terms and delta x
6 a = 0
7 b = pi/2
8 n = 100
9 dx = (b-a)/n
10 % define function to integrate
11 function y = f (x)
12     y = exp(x.^2) .* cos(x);
13 end
14 msum = 0;
15 % initialize sum
16 m1 = a + dx/2; % first midpoint
17 % loop to create sum of function values
18 for i = 1:n
19     m = m1 + (i-1) * dx; % calculate midpoint
20     msum = msum + f (m); % add to midpoint sum
21 end
22 % midpoint approximation to the integral
23 approx = msum * dx
```

Рис.5 Аппроксимирование суммами. Скрипт *midpoint.m*

```

C:\Users\goral
Командное окно
>> midpoint
a = 0
b = 1.5708
n = 100
dx = 0.015708
approx = 1.8758
```

Рис.6 Аппроксимирование суммами. Запуск *midpoint.m*

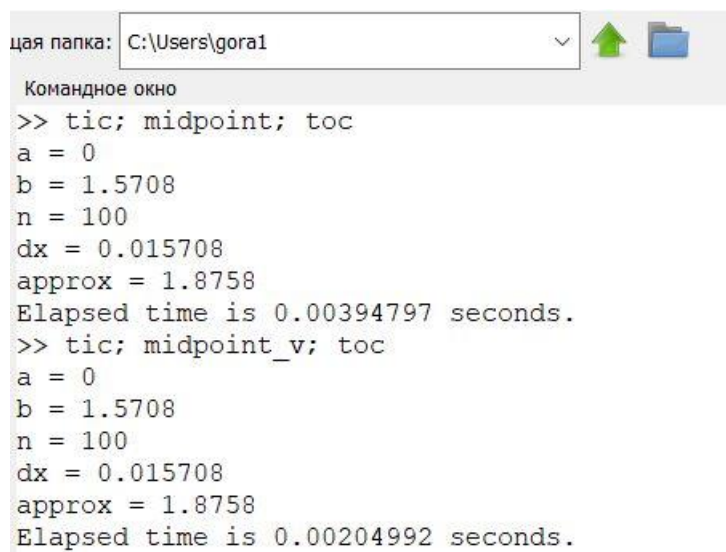
```
midpoint.m x midpoint_v.m x
1 % file 'midpoint_v.m'
2 % calculates a midpoint rule approximation of
3 % the integral from 0 to pi/2 of f(x) = exp (x^2) cos (x)
4 % -- vectorized code
5 % set limits of integration, number of terms and delta x
6 a = 0
7 b = pi/2
8 n = 100
9 dx = (b-a)/n
10 % define function to integrate
11 function y = f (x)
12     y = exp(x.^2) .* cos(x);
13 end
14 % create vector of midpoints
15 m = [a + dx/2:dx:b-dx/2];
16 % create vector of function values at midpoints
17 M = f(m);
18 % midpoint approximation to the integral
19 approx = dx * sum (M)
```

Рис.7 Аппроксимирование суммами. Скринт midpoint_v.m

```
дкая папка: C:\Users\gora1
Командное окно
>> midpoint_v
a = 0
b = 1.5708
n = 100
dx = 0.015708
approx = 1.8758
```

Рис. 8 Аппроксимирование суммами. Запуск midpoint_v.m.

В результате сравнения времени работы программы видим, что midpoint_v.m работает быстрее(см. Рис.9).



```
Командное окно
>> tic; midpoint; toc
a = 0
b = 1.5708
n = 100
dx = 0.015708
approx = 1.8758
Elapsed time is 0.00394797 seconds.
>> tic; midpoint_v; toc
a = 0
b = 1.5708
n = 100
dx = 0.015708
approx = 1.8758
Elapsed time is 0.00204992 seconds.
```

Рис. 9 Сравнение времени выполнения

Вывод

Таким образом, в ходе данной работы я ознакомилась с некоторыми операциями в среде Octave для работы с пределами, последовательностями и рядами, ознакомилась с численным интегрированием.