

# Отчёт по лабораторной работе 6

Супонина Анастасия Павловна

## Содержание

Цель работы .....	1
Задание. ....	1
Вычислить .....	1
Выполнение работы .....	2
Предел .....	2
Частичные суммы .....	3
Сумма ряда часть 1 .....	5
Вычисление интегралов .....	7
Аппроксимирование суммами .....	7
Выводы .....	10

## Список иллюстраций

Элементы списка иллюстраций не найдены.

## Список таблиц

Элементы списка иллюстраций не найдены.

## Цель работы

Ознакомиться с вычислением пределов, последовательностей и рядов в Octave.

## Задание.

### Вычислить

- Предел
- Частичные суммы
- Сумму ряда
- Интеграллы
- Аппроксимирование суммами

# Выполнение работы

## Предел

Записываю функцию  $f$  и  $@$  обозначаю переменную данной функции, которую мы можем отдельно задавать и позднее изменять

```
>> 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
```

```
>> 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
```

Задана переменную и выполняю функцию с этой переменной

```
>> f(n)
ans =

    2.000000000000000
    2.593742460100002
    2.704813829421529
    2.716923932235520
    2.718145926824356
    2.718268237197528
    2.718280469156428
    2.718281693980372
    2.718281786395798
    2.718282030814509

>> format
>> n=[2:1:11]'
n =

     2
     3
     4
     5
     6
     7
     8
     9
    10
    11
```

## Частичные суммы

Записываю значения вектора a

```
>> a = 1./ (n.*(n+2))
a =

    1.2500e-01
    6.6667e-02
    4.1667e-02
    2.8571e-02
    2.0833e-02
    1.5873e-02
    1.2500e-02
    1.0101e-02
    8.3333e-03
```

И в цикле от одного до десяти, суммирую i - значений

```

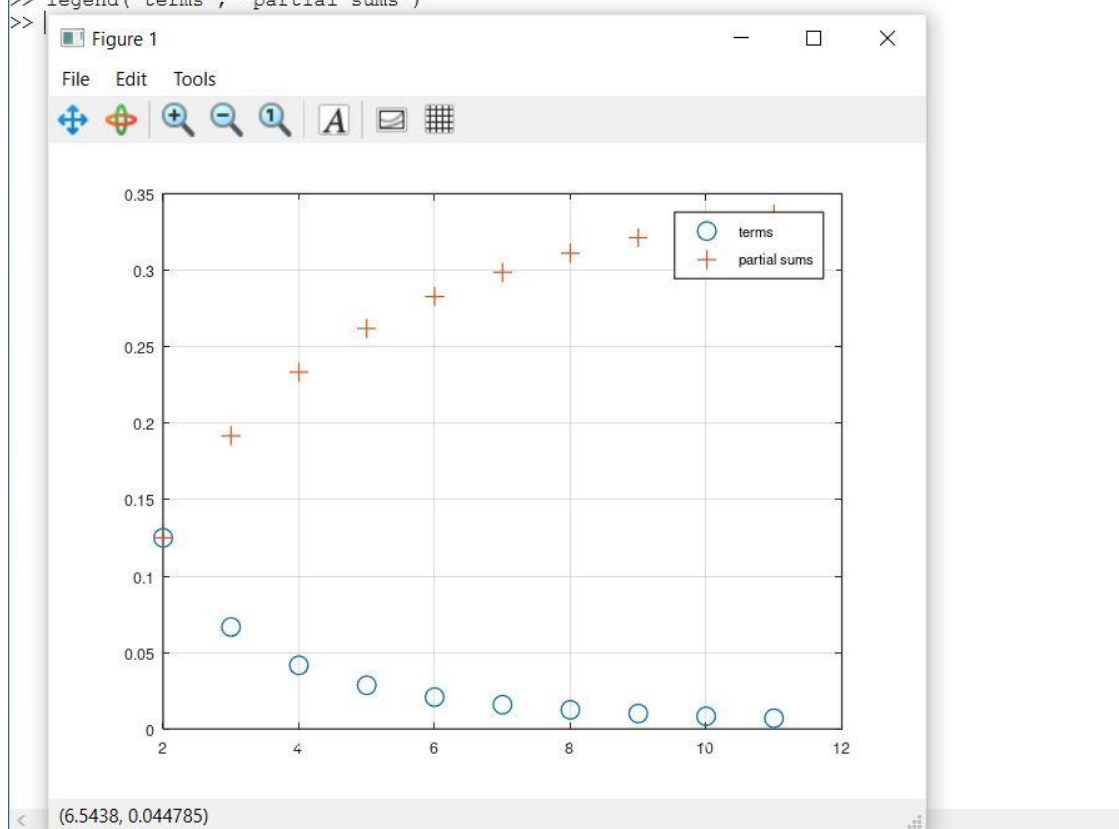
>> for i = 1:10
s (i) = sum(a(1:i))
end
s = 0.1250
s =
    0.1250    0.1917
s =
    0.1250    0.1917    0.2333
s =
    0.1250    0.1917    0.2333    0.2619
s =
    0.1250    0.1917    0.2333    0.2619    0.2827
s =
    0.1250    0.1917    0.2333    0.2619    0.2827    0.2986
s =
    0.1250    0.1917    0.2333    0.2619    0.2827    0.2986    0.3111
s =
    0.1250    0.1917    0.2333    0.2619    0.2827    0.2986    0.3111    0.3212
s =
    0.1250    0.1917    0.2333    0.2619    0.2827    0.2986    0.3111    0.3212    0.3295
s =
    0.1250    0.1917    0.2333    0.2619    0.2827    0.2986    0.3111    0.3212    0.3295    0.3365

```

Отобразила резултат на графике

```
s =
    0.1250    0.1917    0.2333    0.2619    0.2827    0.2986    0.3111    0.3212    0.3295    0.3365
```

```
>> plot(n, a, 'o', n, s, '+')
>> grid on
>> legend('terms', 'partial sums')
>>
```



## Сумма ряда часть 1

Генерирую матрицу n, значениями от 1 до 1000

```
>> n = [1:1:1000]
n =

Columns 1 through 13:

     1     2     3     4     5     6     7     8     9    10

Columns 14 through 26:

    14    15    16    17    18    19    20    21    22    23

Columns 27 through 39:

    27    28    29    30    31    32    33    34    35    36

Columns 40 through 52:

    40    41    42    43    44    45    46    47    48    49

Columns 53 through 65:

    53    54    55    56    57    58    59    60    61    62

Columns 66 through 78:

    66    67    68    69    70    71    72    73    74    75

Columns 79 through 91:

    79    80    81    82    83    84    85    86    87    88

Columns 92 through 104:
```

Генерирую матрицу a

```
>> a = 1./n
a =

Columns 1 through 7:

    1.0000e+00    5.0000e-01    3.3333e-01    2.5000e-01    2.0000e-01    1.6667e-01    1.4286e-01

Columns 8 through 14:

    1.2500e-01    1.1111e-01    1.0000e-01    9.0909e-02    8.3333e-02    7.6923e-02    7.1429e-02

Columns 15 through 21:

    6.6667e-02    6.2500e-02    5.8824e-02    5.5556e-02    5.2632e-02    5.0000e-02    4.7619e-02

Columns 22 through 28:

    4.5455e-02    4.3478e-02    4.1667e-02    4.0000e-02    3.8462e-02    3.7037e-02    3.5714e-02

Columns 29 through 35:

    3.4483e-02    3.3333e-02    3.2258e-02    3.1250e-02    3.0303e-02    2.9412e-02    2.8571e-02

Columns 36 through 42:

    2.7778e-02    2.7027e-02    2.6316e-02    2.5641e-02    2.5000e-02    2.4390e-02    2.3810e-02

Columns 43 through 49:

    2.3256e-02    2.2727e-02    2.2222e-02    2.1739e-02    2.1277e-02    2.0833e-02    2.0408e-02

Columns 50 through 56:
```

Считаю сумму и вывожу результат

```
>> sum(a)
ans = 7.4855
>> |
```

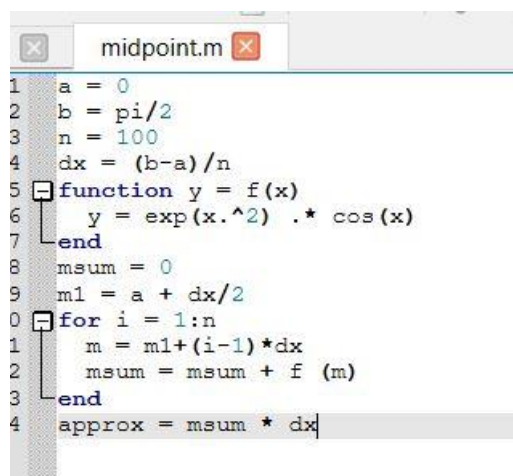
## Вычисление интегралов

Создаю функцию f и при помощи quad получаю значение интеграла от данной функции

```
>> function y = f(x)
y = exp(x.^2) .* cos(x)
end
>> quad('f',0,pi/2)
y = 1.3103
y = 1.0002
y = 0.2267
y = 1.0056
y = 0.9042
y = 1.0319
y = 1.4191
y = 1.1003
y = 1.5288
y = 1.2269
y = 1.3991
y = 1.0000
y = 0.039792
y = 1.0015
y = 0.5458
y = 1.0149
y = 1.2115
y = 1.0595
y = 1.5188
y = 1.1560
y = 1.4792
ans = 1.8757
|
```

## Аппроксимирование суммами

Создаю файл для вычисления аппроксимации первым способом



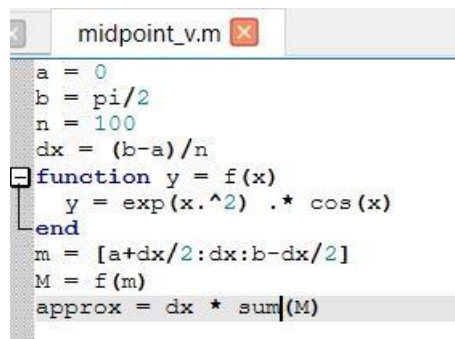
```
midpoint.m
1 a = 0
2 b = pi/2
3 n = 100
4 dx = (b-a)/n
5 function y = f(x)
6     y = exp(x.^2) .* cos(x)
7 end
8 msum = 0
9 m1 = a + dx/2
10 for i = 1:n
11     m = m1+(i-1)*dx
12     msum = msum + f(m)
13 end
14 approx = msum * dx
```

Сохраняю файл под названием midpoint и запускаю, чтобы в консоли увидеть результат выполнения

```
>> midpoint

a = 0
b = 1.5708
n = 100
dx = 0.015708
msum = 0
m1 = 7.8540e-03      m = 1.5472
m = 7.8540e-03      y = 0.2581
y = 1.0000           msum = 119.33
msum = 1.0000        m = 1.5629
m = 0.023562         y = 0.090360
y = 1.0003           msum = 119.42
msum = 2.0003        approx = 1.8758
```

Создаю ещё один файл для вычисления аппроксимации вторым способом



```
midpoint_v.m
a = 0
b = pi/2
n = 100
dx = (b-a)/n
function y = f(x)
    y = exp(x.^2) .* cos(x)
end
m = [a+dx/2:dx:b-dx/2]
M = f(m)
approx = dx * sum(M)
```

Сохраняю файл под названием midpoint\_v и запускаю, чтобы в консоли увидеть результат выполнения



```
>> midpoint_v

a = 0
b = 1.5708
n = 100
dx = 0.015708
m =

Columns 1 through 7:

    7.8540e-03    2.3562e-02    3.9270e-02    5.4978e-02    7.0686e-02

Columns 8 through 14:

    1.1781e-01    1.3352e-01    1.4923e-01    1.6493e-01    1.8064e-01

Columns 15 through 21:

    2.2777e-01    2.4347e-01    2.5918e-01    2.7489e-01    2.9060e-01

Columns 22 through 28:

Columns 97 through 100:

    0.546827    0.409843

approx = 1.8758
(i-search)`: |
```

Используя tic и toc, чтобы узнать время выполнения каждого файла и сравниваю результаты

```
>> tic; midpoint; toc
a = 0
b = 1.5708
n = 100
dx = 0.015708
msum = 0
m1 = 7.8540e-03
m = 7.8540e-03
y = 1.0000
msum = 1.0000
m = 0.023562
y = 1.0003
msum = 2.0003
m = 0.039270
y = 1.0008
msum = 3.0011

Elapsed time is 0.109516 seconds.
```

```
>> tic; midpoint_v; toc
a = 0
b = 1.5708
n = 100
dx = 0.015708
m =

Columns 1 through 7:

    7.8540e-03    2.3562e-02    3.9270e-01

Columns 8 through 14:

    1.1781e-01    1.3352e-01    1.4923e-01

Elapsed time is 0.0540252 seconds.
```

Исходя из результатов мы можем сделать заключение, что второй способ для нахождения аппроксимации является более быстрым.

## Выводы

В процессе выполнения работы, я научилась вычислять пределы, последовательности и ряды в Octave.