Отчёт по лабораторной работе №6

Аветисян Давид Артурович

23 ноября 2024

РУДН, Москва, Россия

Цель работы

Познакомиться со сложными алгоритмами в Octave, которые были встроены для работы с пределами, последовательностями и рядами.

Пределы

Первым делом я научился работать с пределами. Для это я рассмотрел предел функции $f(n) = (1 + 1/n)^n$ при $n- > \infty$.

```
octave:1> diary
octave:2> f = @(n) (1 + 1 ./ n) .^n
f =

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

octave:3> k = [0:1:9]'
k =

   0
   1
   2
   3
   4
   5
   6
   7
```

После этого я взял степени числа 10, которые стали удобным входным значением и оценил $f(n)$. Предел сходится к конечному значению, которое приблизительно равно $2,71828$.

```
octave:5> n = 10 .^ k
n =

            1
           10
          100
         1000
        10000
       100000
      1000000
     10000000
    100000000
   1000000000

octave:6> f (n)
ans =

   2.000000000000000
   2.593742460100002
   2.704813829421528
   2.716923932235594
   2.718145926824926
```

Далее я познакомился с частичными суммами. Для этого я рассмотрел сумму ряда a от n при n от 2 до ∞, где $a = 1/(n * (n + 2))$.

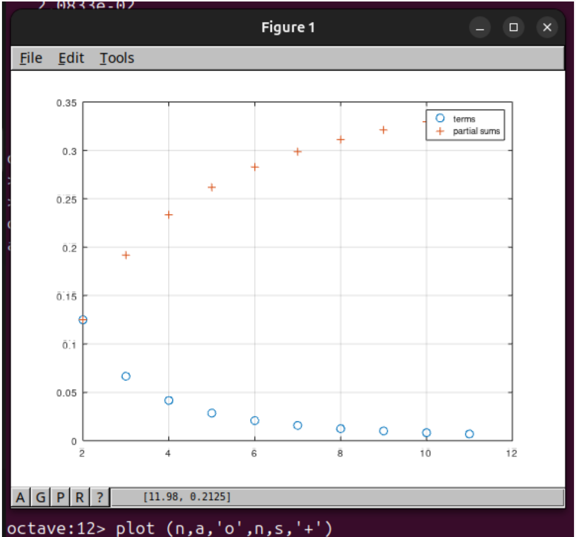
```
octave:8> n = [2:1:11]';
octave:9> a = 1 ./ (n .* (n+2))
a =

   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
   6.9930e-03

octave:10> for i = 1:10
> s (i) = sum(a(1:i));
> end
octave:11> s'
ans =

   0.1250
   0.1917
   0.2333
   0.2619
```

И в конце я построил слагаемые и частичные суммы для $2 <= n <= 11$.



После я познакомился с нахождением суммы первых 1000 членов гармонического ряда $1/n$. Для этого я сгенерировал члены как вектор ряда, а затем взял их сумму.

```
octave:15> n = [1:1:1000];
octave:16> a = 1 ./ n;
octave:17> sum (a)
ans = 7.4855
```

Рис. 5: Нахождение суммы первых 1000 членов гармонического ряда $1/n$

Потом я познакомился с вычислением интегралов. Для этого я взял интеграл от 0 до $pi/2$ функции $f(x) = exp(x^2) * cos(x)$. Для вычисления я определил функцию и использовал команду *quad*.

```
octave:18> function y = f(x)
> y = exp (x .^ 2) .* cos (x);
> end
octave:19> quad ('f',0,pi/2)
ans = 1.8757
```

Рис. 6: Вычисление интеграла $f(n)$

Аппроксимирование суммами

В конце лабораторой работы я научился аппроксимировать суммы. Сначала я написал скрипт чтобы вычислить предыдущий интеграл по правилу средней точки для $n = 100$ и запустил его.

```
  GNU nano 7.2
% file 'midpoint.m'
% calculates a midpoint rule approximation of
% the integral from 0 to pi/2 of f(x) = exp (x^2) cos (x)
% -- traditional looped code
% set limits of integration, numbers of terms and delta x
a = 0
b = pi/2
n = 100
dx = (b-a)/n
% define function to integrate
function y = f (x)
y = exp (x .^ 2) .^ cos(x);
end
msum = 0;
% initialize sum
m1 = a + dx/2; % first midpoint
% loop to create sum of function values
for i = 1:n
m = m1 + (i-1) * dx; % calculate midpoint
msum = msum + f (m); % add to midpoint sum
end
% midpoint approximation to the integral
```

Далее я написал новый скрипт, но использовал векторизованный код, который не требует каких-либо циклов. После чего я его запустил.

```
  GNU nano 7.2
% file 'midpoint_v.m'
% calculates a midpoint rule approximation of
% the integral from 0 to pi/2 of f(x) = exp (x^2) cos (x)
% -- vectorized code
% set limits of integration, numbers of terms and delta x
a = 0
b = pi/2
n = 100
dx = (b-a)/n
% define function to integrate
function y = f (x)
y = exp (x .^ 2) .^ cos(x);
end
% create vector of midpoints
m = [a+dx/2:dx:b-dx/2];
% create vector of function values at midpoints
M = f(m);
% midpoint approximation to the integral
approx = dx * sum (M)
```

Рис. 8: Скрипт midpoint_v.m

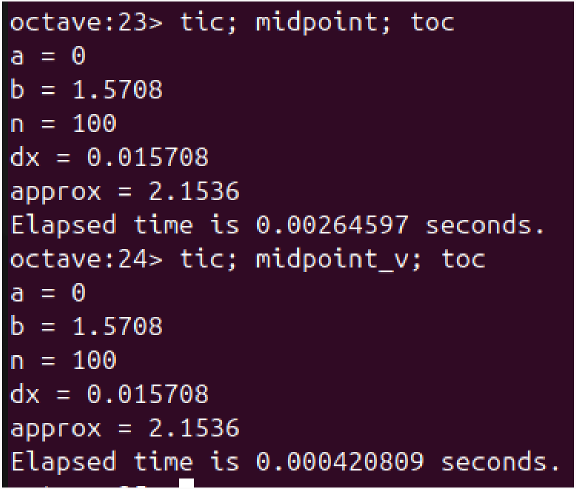

```
octave:23> tic; midpoint; toc
a = 0
b = 1.5708
n = 100
dx = 0.015708
approx = 2.1536
Elapsed time is 0.00264597 seconds.
octave:24> tic; midpoint_v; toc
a = 0
b = 1.5708
n = 100
dx = 0.015708
approx = 2.1536
Elapsed time is 0.000420809 seconds.
```

Рис. 9: Сравнение результатов и времени выполнения

Выводы

Я познакомился со сложными алгоритмами в Octave, которые были встроены для работы с пределами, последовательностями и рядами.