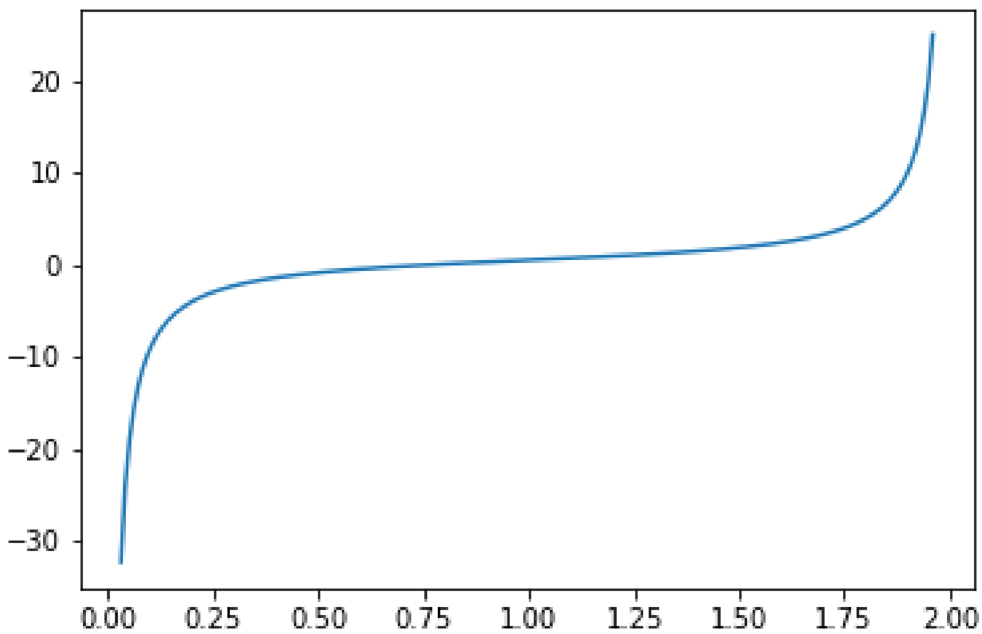
Задание на 22.01

Кириленко Андрей, ВШЭ

22 января 2019

$\boxed{2a}$

Достаточно очевидно, что хотя бы 3 слагаемых посчитать потребуется, а значит можно считать что знаменатели окажутся положительными(в хвосте) Тогда можно оценить член ряда так: $\frac{1}{x^2-x-z} < \frac{1}{x^2-x-2}$ Получили оценку, осталось найти такое n, что интеграл ограничивающей функции не больше заданной точности. $\frac{1}{x^2-x-2} = 1/3*(\frac{1}{x-2} - \frac{1}{x+1})$, неопределенный интеграл равен $1/3*\log|\frac{2-x}{x+1}|(+c)$, теперь надо найти такое n, что подстановка от n до бесконечности будет не больше заданной точности. Подстановка бесконечности даст 0, так как в логарифме предел 1, останется min n: $-1/3\log|\frac{2-n}{n+1}| \leq eps$ т.е. $\log|\frac{n+1}{2-n}| \leq 3*eps$, что дает оценку n чуть больше 1000000. Остается посчитать функцию как частичную сумму до $n = 10^6$ и построить график. По оси абсцисс – точка, в которой считаем, по оси ординат – сумма ряда с заданной точностью.



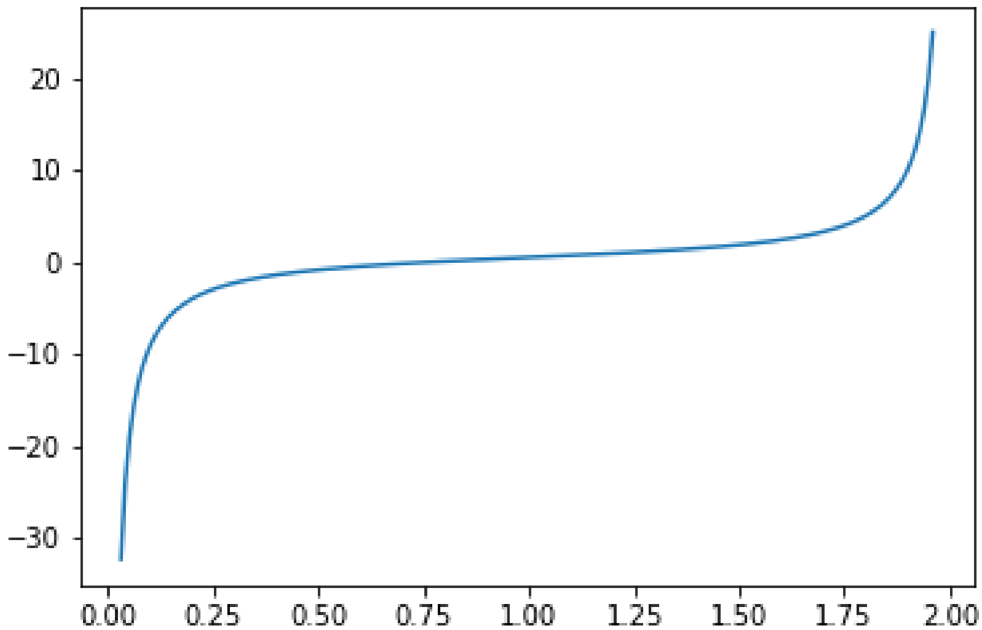
Код:

```
def Wa(z):
    sum = 0
    for i in range(1, 1100000):
        sum += 1.0 / (i * i - i - z)
    return sum


def plotWa(fn):
```

```
a1 = []
a2 = []

for i in range(3, 197): # start point 3/100 = 0.03, end point: 197/100=1.
    arg = i / 100.0
    a1.append(arg)
    a2.append(Wa(arg))

graph.plot(a1, a2, label=fn)
graph.savefig(fn)
graph.close()
```

$\boxed{\text{2b}}$

Возьмем модельный ряд такой: $\sum \frac{1}{k(k-1)}$ (определим его слагаемое для n=1 как ноль). Сумма такого ряда равна 1. Разность будет соответственно $\frac{z}{(k^2-k)(k^2-k-z)}$ (первое слагаемое опять же особенное и равно $-1/z$) Останется поработать с остальной разностью, ее можно очевидно оценить как $4/k^4$ (2 берется от z и еще 2 от оценки знаменателя как половина старшего члена: $\frac{z}{(k^2-k)(k^2-k-z)} < \frac{2}{(k^2-k)(k^2-k-z)} < \frac{2}{k^4/2}$) (Можно точнее оценить константу числом 2 и получить 88 слагаемых). Интеграл равен $-4/3 * x^{-3}$(+c). Решаем уравнение $4/3 * x^{-3} \leq eps$, что дает ответ около 110. Итого для подсчета ряда надо к $1 - 1/z$ прибавить не менее 110 слагаемых ряда разности(начиная с 2)



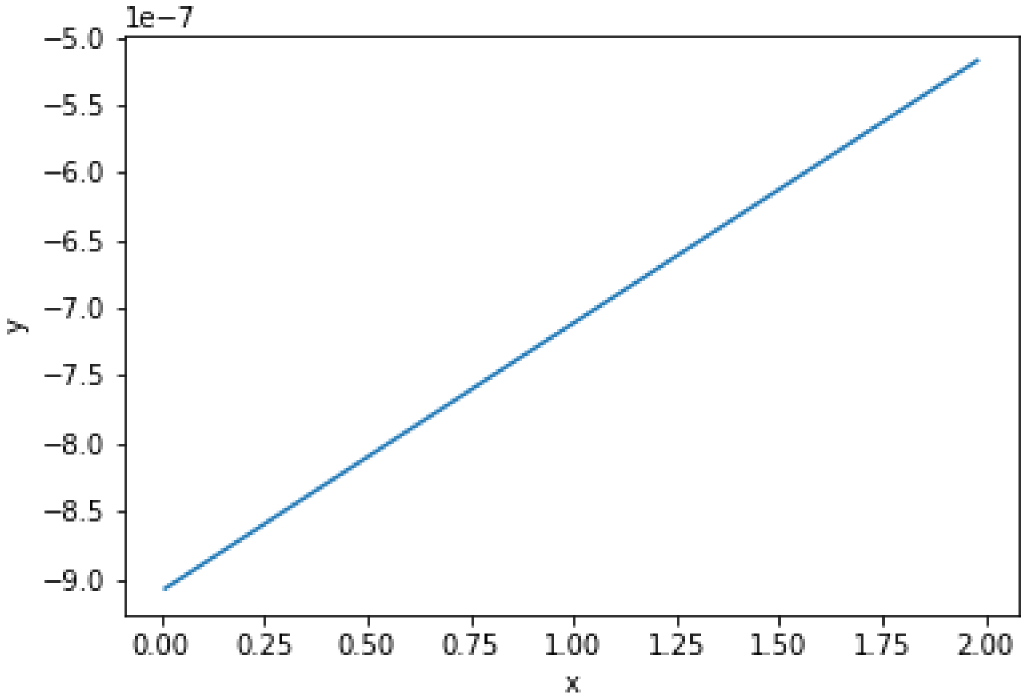
Код:

```
def Wb(z):
    sum = 1-1.0/z
    for i in range(2, 120):
```

```
        sum += z / ((i * i - i) * (i * i - i - z))
    return sum

def plotWb(fn):
    a1 = []
    a2 = []

    for i in range(3, 197): # start point 3/100 = 0.03, end point: 197/100=1.
        arg = i / 100.0
        a1.append(arg)
        a2.append(Wb(arg))

    graph.plot(a1, a2, label=fn)
    graph.savefig(fn)
    graph.close()
```

$\boxed{\text{2c}}$

Посчитаем разность, проверим что максимум разности по модулю не больше $2 * eps$, построим график. Видно, что разница очень мала и не превосходит погрешности



Код:

```
def diffW(z):
    return Wa(z) - Wb(z)

def plotDiffW(fn):
    a1 = []
```

```
a2 = []

for i in range(1, 199): # start point 1/100 = 0.01, end point: 199/100=1.
    arg = i / 100.0
    a1.append(arg)
    a2.append(diffW(arg))

max_diff_abs = max(abs(max(a2)), abs(min(a2)))
print(max_diff_abs) # difference less than 2*eps
print(max_diff_abs < 2.0 / 1000000)
graph.xlabel('x')
graph.ylabel('y')
graph.plot(a1, a2, label=fn)
graph.savefig(fn)
graph.close()
```

$\boxed{3}$

//Длинный код, скрипт лежит на гите.
Рассчитаем точное значение как много шагов метода Эйткина, а неточное как мало шагов. Также рассмотрим итерированный и неитерированный методы и получим такие результаты:

-0.9
Res -0.7201172295789915
Iterated -0.7201172295789915
Simple -0.7201172295789915

i
Res (-0.24374774719955403+0.8669729873397874j)
Iterated (-0.24374774719967915+0.8669729873399104j)
Simple (-0.2437477425162039+0.8669729825771895j)

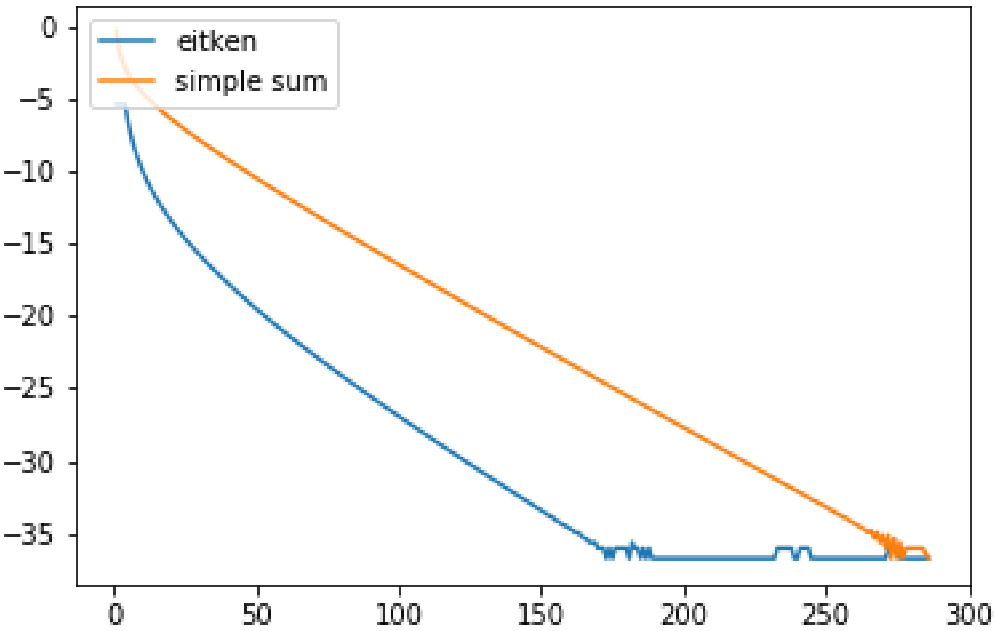
-1
Res -0.7853981633975087
Iterated -0.7853981633974485
Simple -0.7853981657591506

$e^{3ip/4}$
Res (-0.6484643618062063+0.48681284896590743j)
Iterated (-0.648464361806174+0.48681284896598226j)
Simple (-0.648464360650483+0.4868128517288752j)

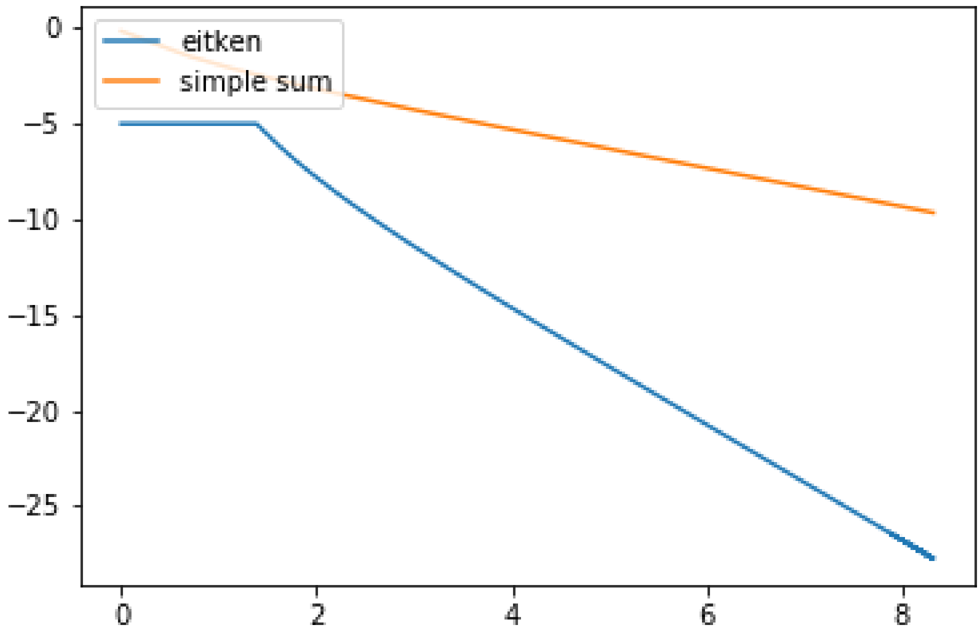
Все результаты точны, однако итерированный ближе к ответу.

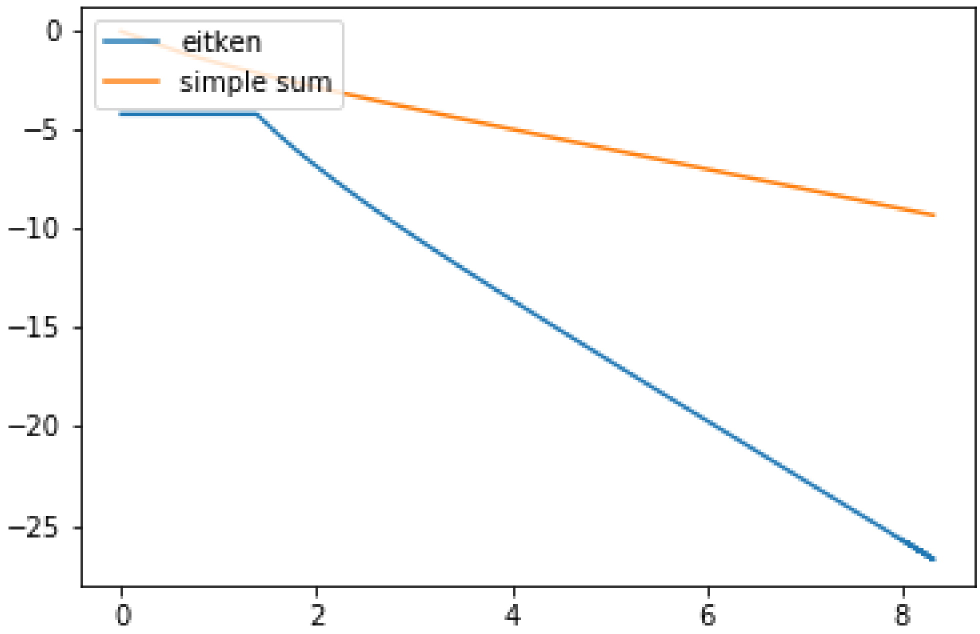
Перейдем к скорости сходимости. Ясно, что при z по модулю меньше 1 не интересно, быстро сходимся, так как экспонента. Если z по модулю больше 1, то ряд расходится. Остается из интересного только окружность, на ней и есть смысл проверять численно. Заметим, что при -1 ряд сходится, а при 1 расходится. Значит, скорость сходимости должна уменьшаться при движении по окружности. Проверим это численно.

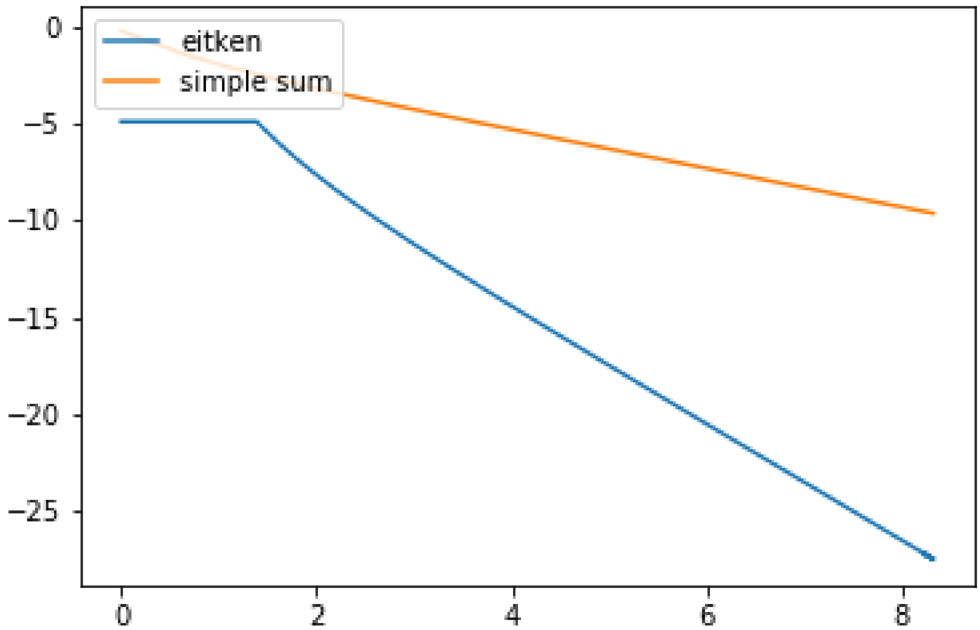
Для начала график логарифма погрешности от N при -0,9 и сходимость экспоненциальная:



А еще логарифма пограешности от логарифма N для точек соответственно -1, i, $e^{3ip/4}$:


eitken
simple sum
0
−5
−10
−15
−20
−25
0
2
4
6
8


eitken
simple sum
0
−5
−10
−15
−20
−25
0
2
4
6
8



Видно что Эйткин ускоряет, но при этом просто меняется угол наклона прямой.
Перейдем с скорости в зависимости от точки z. Рассматриваем верхнюю полуокружность от -1 до 1 с шагом угла pi/100:
По оси абсцисс значение угла (в сотых долях от пи), по оси ординат $\log_{10}(1/|value - calculatedResult|)$
Получается, итерированный точнее(быстрее сходится) (так как знаменатель меньше, раз значение больше), однако порядок сходимости не поменялся.