**Белорусский государственный университет**

**Факультет прикладной математики и информатики**

Лабораторная работа №3

по Методам численного анализа

Задание №ЗК-15

Выполнил :

Студент 3 курса 8 группы ФПМИ ТП

Бусыгин Владимир Дмитриевич

Преподаватель :

Радкевич Елена Владимировна

2021**Описание алгоритма построения неявного метода Эйлера**

Исходная задача :

Проинтегрируем уравнение на отрезке :

Указав метод решения правой части и отбросив остаток можем получить метод :

Используя метод правых прямоугольников, получим неявный метод Эйлера :

Для реализации неявного метода Эйлера будем использовать алгоритм метода Ньютона :

В качестве начального приближения будем брать :

**Описание алгоритма построения метода последовательного повышения порядка точности**

Исходная задача :

Проинтегрируем уравнение на отрезке :

Проведем замену переменных :

Тогда мы получим :

Проведем замену квадратурной суммой :

Выбор параметров осуществляется таким образом, чтобы КФ оставалась точной для многочленов степени до k-1 включительно :

По зданию требуется построить метод 2-го порядка точности при q=0 :

**Описание алгоритма построения метода Рунге-Кутта**

Исходная задача :

Для построения методов любой заданной точности используется набор заданных значений и для моего задания они :

Общий случай метода Рунге-Кутта :

В моем случае метод второго порядка :

**Описание алгоритма построения экстраполяционного метода Адамса**

Исходная задача :

Заменим интерполяционным многочленом Лагранжа степени k по узлам со значениями .

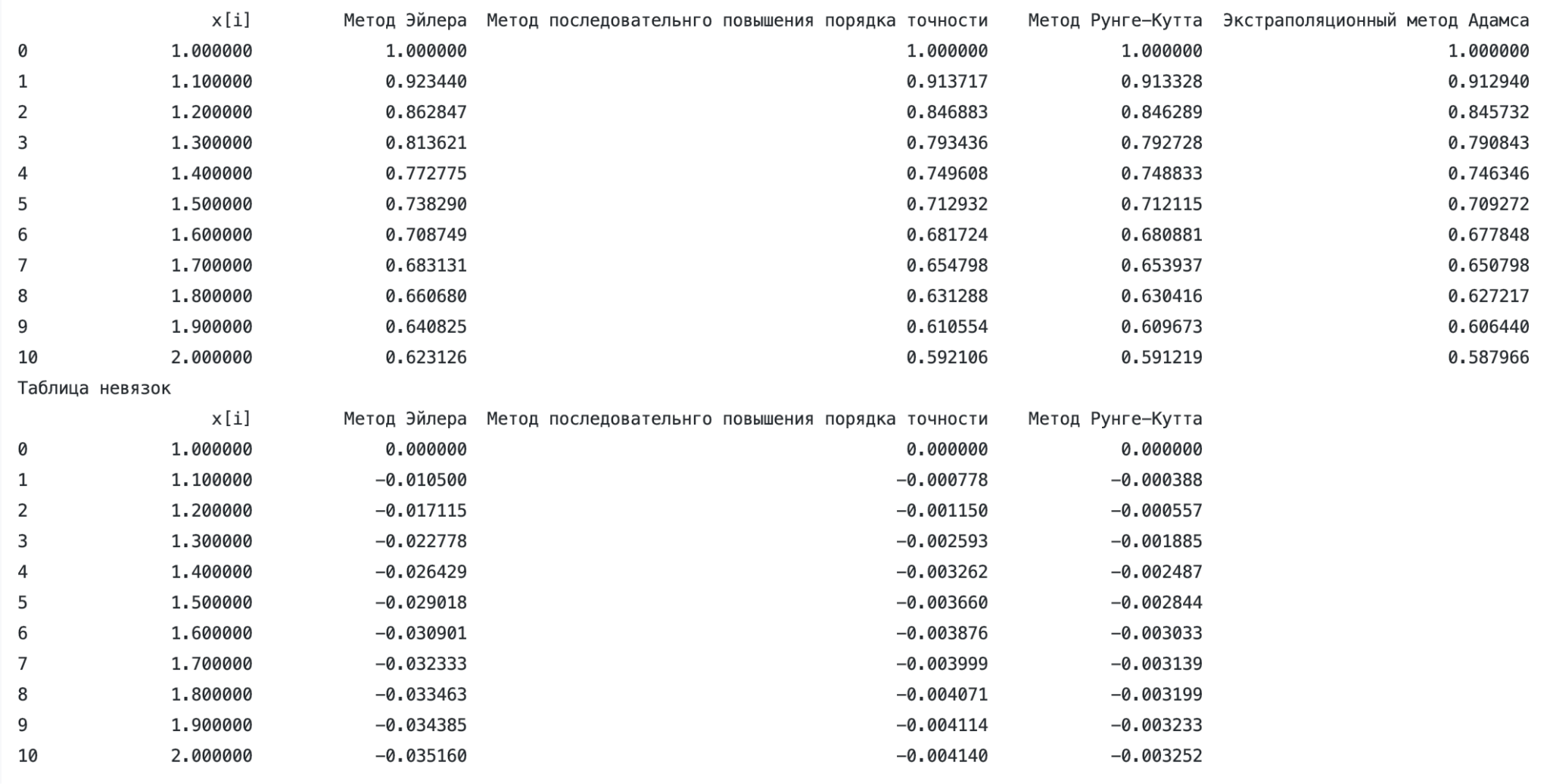
После подстановки и отбросив остатки интерполирования, получим метод вида :  
 где - коэффициенты многочлена Лагранжа.

По заданию требуется построить метод третьего порядка, соответствующая формула ей будет :

При этом начало таблицы метода необходимо построить по методу последовательного повышения порядка точности третьего порядка :

**Анализ полученных результатов**

**Входные данные :**

**Выходные данные :**  


Наиболее точным методом оказался метод Рунге-Кутта, менее точным метод последовательного повышения порядка точности, а за ним располагается метод Эйлера.

Неявный метод Эйлера является методом первого порядка точности, что можно наблюдать по величине невязки метода. Метод последовательного повышения порядка точности и метод Рунге-Кутта, в моем случае являются методами второго порядка точности, также по величине невязки мы можем наблюдать данное соответствие. Однако по величине невязки можно наблюдать, что метод Рунге-Кутта дал более точное решение, объяснить это можно разницей коэффициентов при главных членах погрешности, для методов второго порядка главным членом погрешности будет .

**Код программы (Python)**

**Задание 1 :**

from numpy import log, arange

# **TODO: Неявный метод Эйлера с реализацией метода Ньютона**

a = 1

b = 2

N = 10

h = (b - a) / N

f = lambda x, u: (u\*u\*log(x) - u) / x

x = [i for i in arange(a, b + h, h)]

y = [0.0 for i in range(0, N + 1)]

y[0] = 1 # начальное условие

df = lambda x, u: (2\*u\*log(x) - 1) / x

F = lambda y\_l, j : y\_l - y[j - 1] - h \* f(x[j], y\_l)

dF = lambda y\_l, j: 1 - h \* df(x[j], y\_l)

def Nuton(y: float, j: float) -> float:

return y - F(y, j) / dF(y, j)

def method() -> list:

for j in range(1, N + 1):

start = y[j - 1]

end = Nuton(start, j)

while abs(start - end) >= h\*\*4:

start = end

end = Nuton(end, j)

y[j] = y[j - 1] + h \* f(x[j], end)

return y

**Задание 2 :**

from numpy import log, arange

# **TODO: 2.Метод последовательного повышения точности 2-го порядка q = 0**

a = 1

b = 2

N = 10

h = (b - a) / N

f = lambda x, u: (u\*u\*log(x) - u) / x

x = [i for i in arange(a, b + h, h)]

y = [0.0 for i in range(0, N + 1)]

y[0] = 1 # начальное условие

def method() -> list:

for j in range(0, N):

y\_half = y[j] + h \* f(x[j], y[j]) / 2

y[j + 1] = y[j] + h \* f(x[j] + h / 2, y\_half)

return y

**Задание 3 :**

from numpy import log, arange, array

# **TODO : Метод Рунге-Кутты**

a = 1

b = 2

N = 10

h = (b - a) / N

A = [1/2, 1/2] # q + 1

alfa = [1, 0] # q + 1

beta = [[1, 0], # q x q

[0, 0]]

q = 1

f = lambda x, u: (u\*u\*log(x) - u) / x

x = [i for i in arange(a, b + h, h)]

y = [0.0 for i in range(0, N + 1)]

y[0] = 1 # начальное условие

def method() -> list:

for j in range(0, N):

fi = [h for i in range(0, q + 1)]

fi[0] \*= f(x[j], y[j])

y[j + 1] = y[j] + A[0]\*fi[0]

for i in range(1, q + 1):

temp\_y = y[j]

for k in range(0, i):

temp\_y += beta[i][k]\*fi[k]

fi[i] \*= f(x[j] + alfa[i]\*h, temp\_y)

y[j + 1] += A[i]\*fi[i]

return y

**Задание 4 :**

from numpy import log, arange

# **TODO: 4.Экстраполяцион метод Адамса 3-го порядка с методом повышения точности 3-го порядка**

a = 1

b = 2

N = 10

h = (b - a) / N

f = lambda x, u: (u\*u\*log(x) - u) / x

x = [i for i in arange(a, b + h, h)]

y = [0.0 for i in range(0, N + 1)]

y[0] = 1 # начальное условие

k = 2

def begin\_table() -> None:

for j in range(0, k):

y\_temp = y[j] + h \* f(x[j], y[j]) / 3

y\_temp = y[j] + 2 \* h \* f(x[j] + h / 3, y\_temp) / 3

y[j + 1] = y[j] + h \* (f(x[j], y[j]) + 3 \* f(x[j] + 2 \* h / 3, y\_temp)) / 4

def method() -> list:

begin\_table()

for j in range(k, N):

y[j + 1] = y[j] + h \* (23 \* f(x[j], y[j]) - 16 \* f(x[j - 1], y[j - 1]) + 5 \* f(x[j - 2], y[j - 2])) / 12

return y