Министерство образования и науки Российской Федерации

Федеральное государственное автономное образовательное учреждение высшего образования

«Национальный исследовательский Томский политехнический Университет»



Инженерная школа ядерных технологий

01.03.02 «Прикладная математика и информатика»

**Лабораторная работа №3**

**«СЛУЧАЙНЫЕ ПРОЦЕССЫ. ПРОЦЕСС ВИНЕРА.**

**СТОХАСТИЧЕСКОЕ ИНТЕГРИРОВАНИЕ»**

по дисциплине:

**Теория случайных процессов**

Вариант 8

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Выполнил:** |  | | | | |
| Студент группы | 0В02 |  | Редько Д.А. |
|  |  |  |
| **Проверил:** | Крицкий О.Л. | | | | |
| преподаватель |  |  |  |  |  | |
|  |  |  |  |  |  | |

Томск – 2023

# Задание:

1. Сгенерировать 103 значений стандартной случайной величины.
2. Построить график винеровского процесса для моментов *t* из интервала 0*≤t≤*4 года с шагом *h*=4x10-3.
3. Построить и изобразить реализацию винеровского процесса как случайное блуждание: пусть *xk* – СВ, имеющая биномиальное распределение и принимающая значения ± 1 с одинаковыми вероятностями *p* = 0.5 и *q* = 0.5. Пусть *N* – число таких случайных величин. Тогда  для некоторого момента времени *t*. Для отрисовки использовать *N* = 104­­­ величин, задействовать функцию pause в Matlab, чтобы график винеровского процесса выводился в реальном времени.
4. В соответствии с номером варианта сгенерировать процесс ценообразования рискового актива по формуле (**не забудьте перевести волатильность в исходном задании в доли**)

.

Интегралы вычислять численно методом трапеций и Монте-Карло для первого и второго интегралов соответственно с погрешностью не ниже 10-2 и вероятностью не ниже 0,95. Последовательно положить моменты времени равными *t*=0,5;1;…; 4 года.

1. Сравнить полученные данные для *St* c данными для котировок облигаций в те же моменты времени.
2. Решить методом Рунге-Кутты четвертого порядка точности дифференциальное уравнение . Сравнить с результатом, найденным в п.3 в те же моменты времени *t*. **Не забудьте перевести волатильность в исходном задании в доли.**

|  |  |  |  |
| --- | --- | --- | --- |
| 8 | 20 | 0.05 *t* | *|t2-3t|* |

**Теоретическая часть.**

1. **Винеровский процесс.**

Винеровским процессом  называют случайный процесс, для

которого выполнены следующие аксиомы:

* для любого разбиения временного интервала [0, T] точками

 приращения

 независимы;

* Пусть . Тогда случайная величина W (t)−W (s)

распределена нормально с нулевым средним и дисперсией t − s;

* Реализации  непрерывны по t на T.

1. **Метод трапеции**

Метод трапеций:



Где m – количество точек, a,b – пределы интегрирования, f(ξ)’’ – максимум второй производной из интервала интерполяции.

1. **Метод Монте-Карло**

Метод Монте-Карло можно обобщить на случай стохастического интеграла следующим образом:



где  - волатильность,  – количество точек разбиения на сетке [0,t], normrnd(0,1,1) – стандартная случайная величина, rand(t) – случайная величина равномерного распределения на отрезке [0,t].

1. **Метод Рунге-Кутты**

Расчетная формула Рунге-Кутты 4-го порядка:

где

.

**Ход работы:**

1. Сгенерировать 103 значений стандартной случайной величины.

Генерировать значения будем при помощи пакета numpy языка Python 3. Для наглядности выведем только первые 100 значений (рис 1).



Рисунок 1 Значение стандартной случайной величины

Если полученные значения отобразить на графике, то получим белый шум (рис 2).

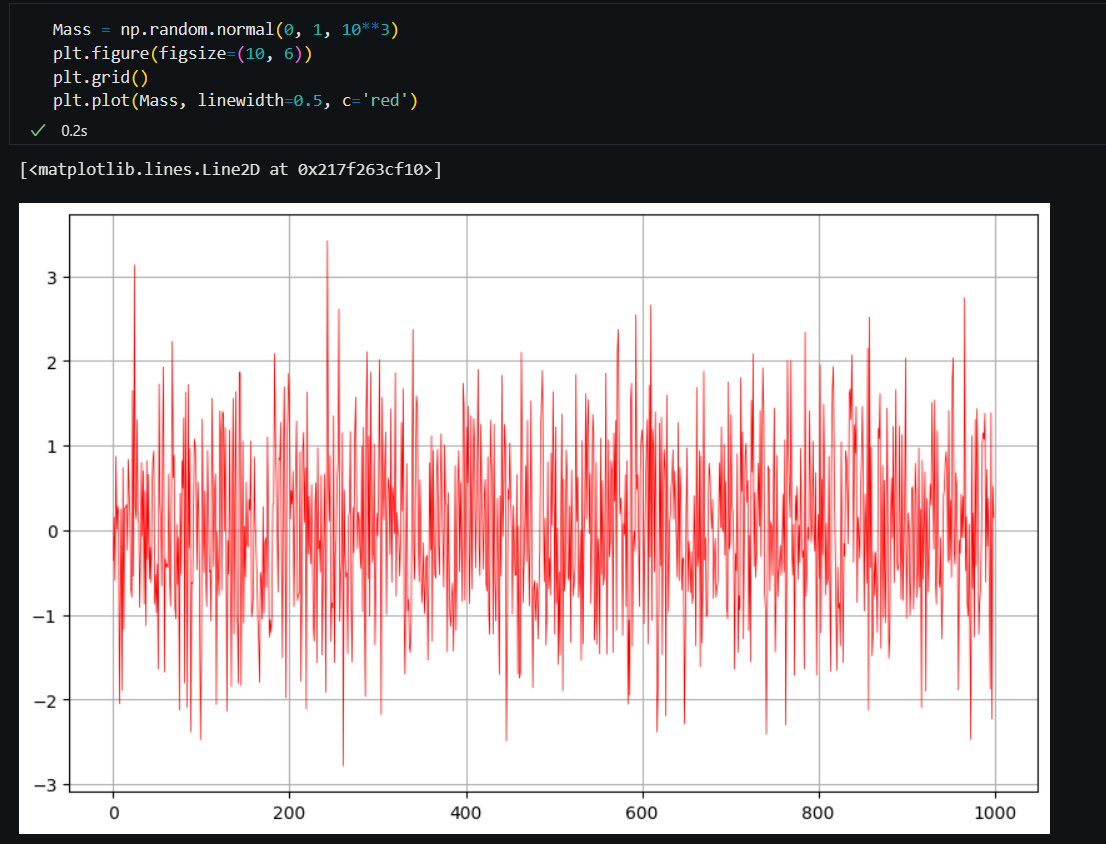


Рисунок 2 График белого шума

1. Построить график винеровского процесса для моментов t из интервала

0≤t≤4 года с шагом  .

Пользуясь формулой  получим винеровский процесс (рис 3).

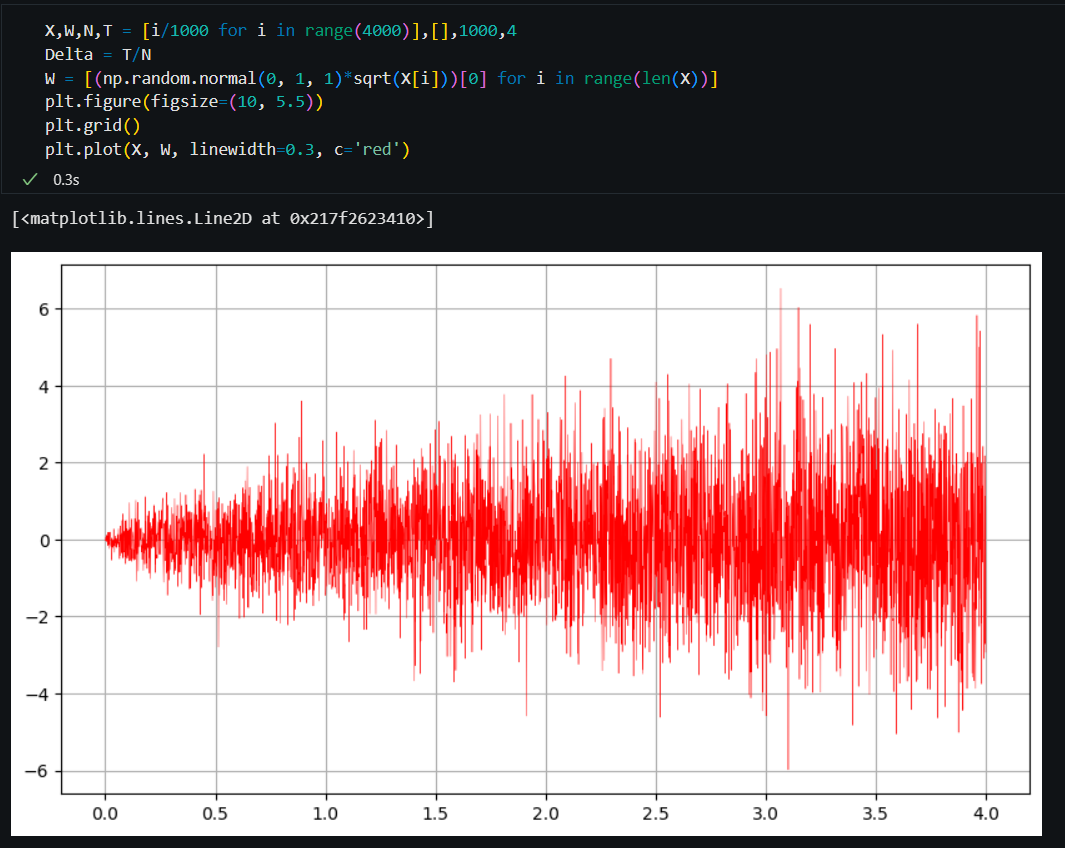


Рисунок 3 Винеровскй процесс

1. Построить и изобразить реализацию винеровского процесса как

случайное блуждание: пусть xk – СВ, имеющая биномиальное распределение и принимающая значения ± 1 с одинаковыми вероятностями p = 0.5 и q = 0.5. Пусть N – число таких случайных величин. Тогда  для некоторого момента времени t. Для отрисовки использовать N = 104 величин. График винеровского процесса выводился в реальном времени.

Для реализации анимированного вывода будем пользоваться библиотекой mathplotlib (рис 4,5).

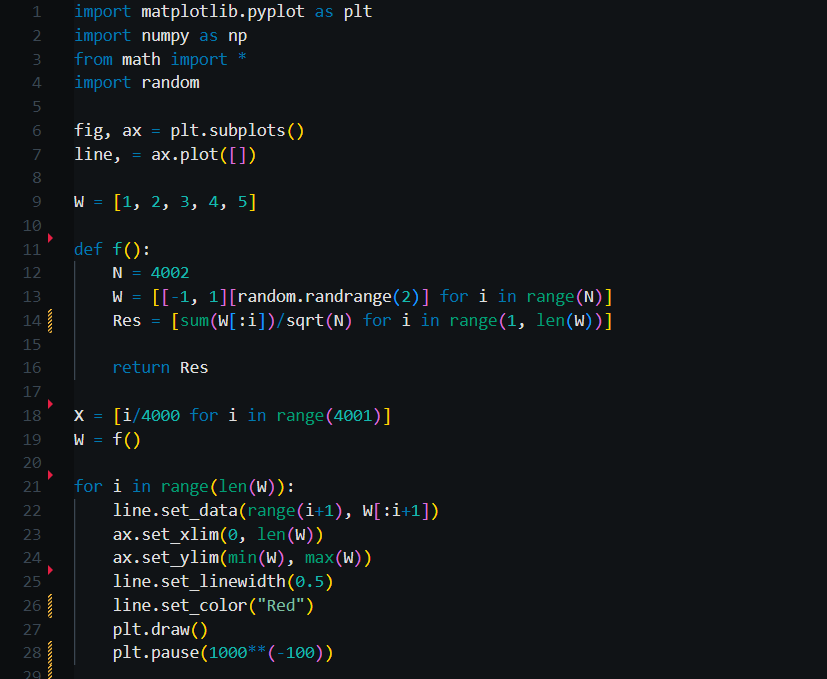


Рисунок 4 Код реализации случайных блужданий

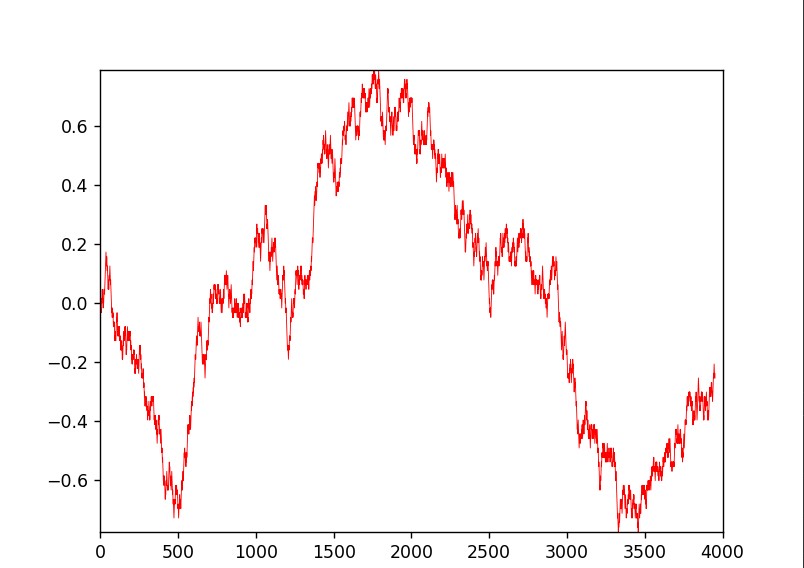


Рисунок 5 Полученный график случайных блужданий

1. В соответствии с номером варианта сгенерировать процесс ценообразования рискового актива по формуле (**не забудьте перевести волатильность в исходном задании в доли**)

.

Интегралы вычислять численно методом трапеций и Монте-Карло для первого и второго интегралов соответственно с погрешностью не ниже 10-2 и вероятностью не ниже 0,95. Последовательно положить моменты времени равными *t*=0,5;1;…; 4 года.

Для реализации опишем классы для метода трапеций и для метода Монте-Карло (рис 6, 7).

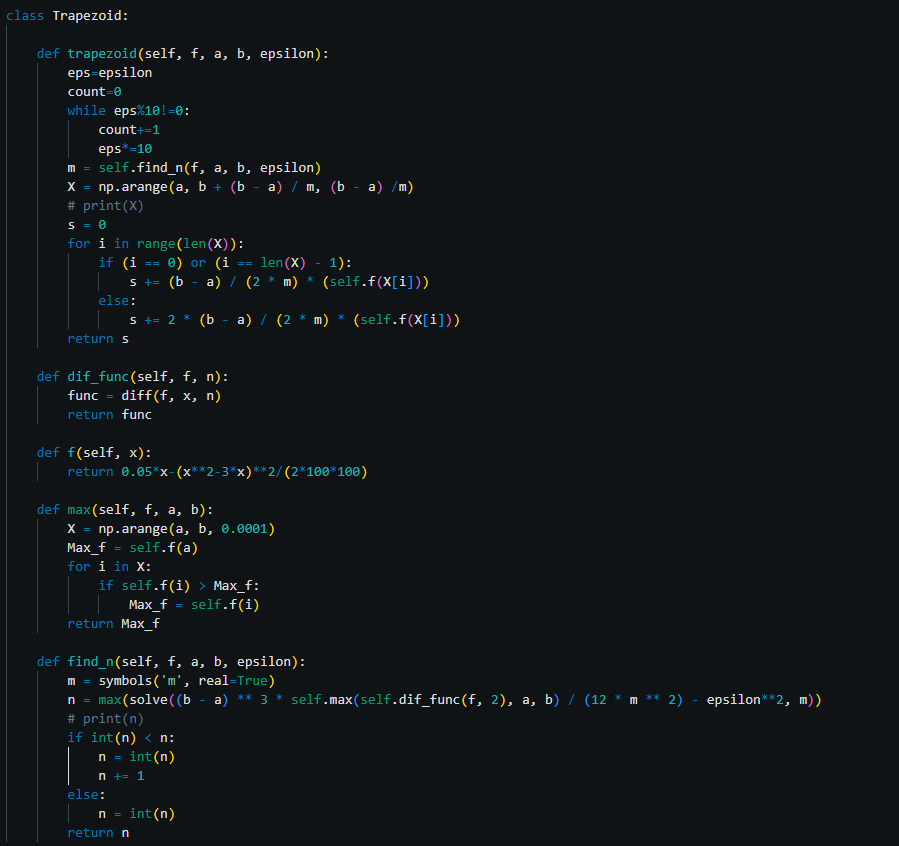


Рисунок 6 Класс для метода трапеций

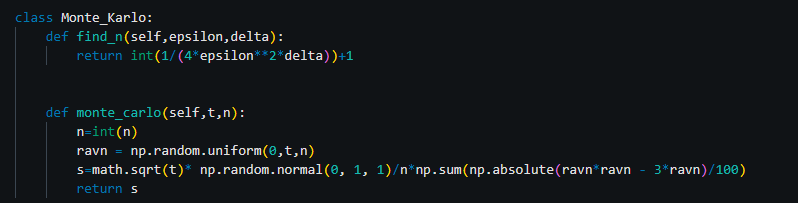


Рисунок 7 Класс для метода Монте-Карло

Используя написанные классы, получим значения интегралов для 8 точек (рис 8).



Рисунок 8 Цена базового актива, полученная вычислением интегралов методами трапеции и Монте-Карло

1. Сравнить полученные данные для St c данными для котировок облигаций  в те же моменты времени.

Для вычисления заданного интеграла так же воспользуемся методом трапеций (рис 9).

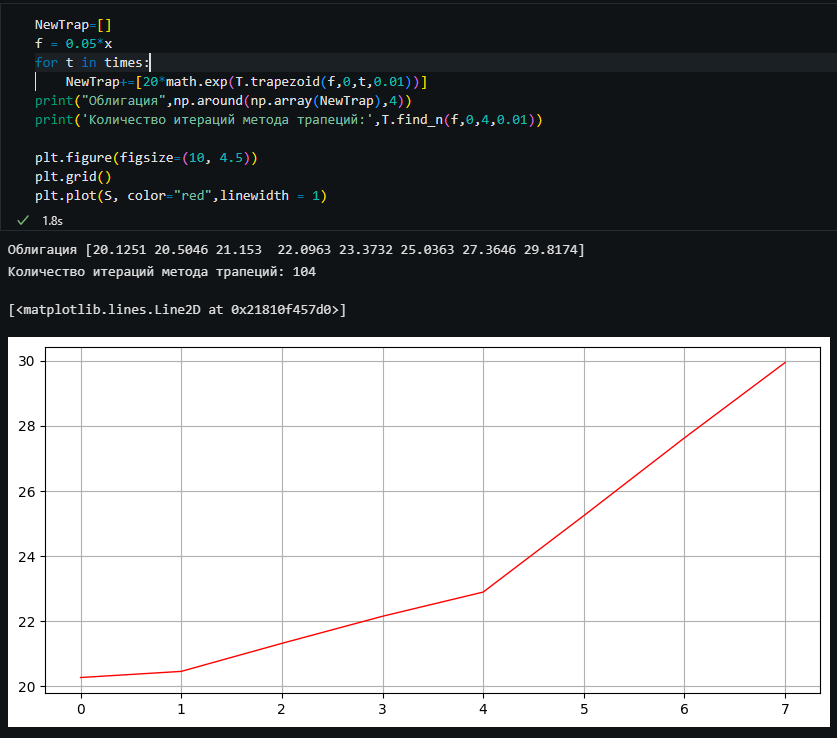


Рисунок 9 Вычисление цены облигации методом трапеции

Заметим, что цена облигации незначительно отличается от цены базового актива.

1. Решить методом Рунге-Кутты четвертого порядка точности дифференциальное уравнение . Сравнить с результатом, найденным в п.3 в те же моменты времени *t*. **Не забудьте перевести волатильность в исходном задании в доли.**

Для решения, будем использовать известный из ранее пройденных курсов метод Рунге-Кутты четвертого порядка (рис 10, 11).

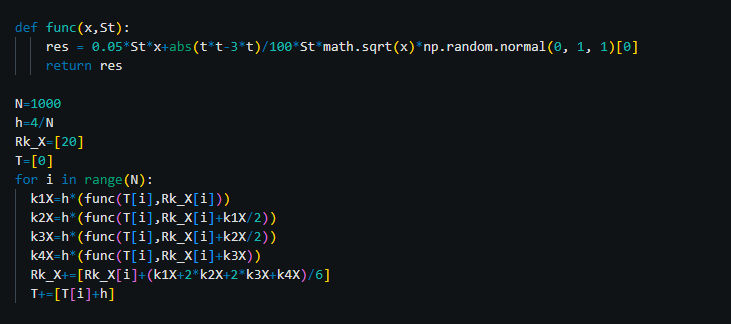


Рисунок 10 Метод Рунге-Кутты 4-го порядка

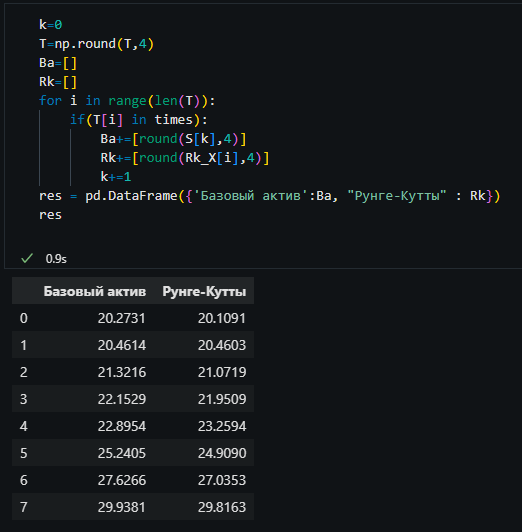


Рисунок 11 Сравнение результатов, полученных методом Рунге-Кутты и методом прямого численного интегрирования.

**Вывод:** в ходе данной лабораторной работы познакомились с разными интерпретациями винеровского процесса, применили методы численного интегрирования к стохастическим процессам, а также решили стохастическое дифференциальное уравнение методом Рунге-Кутты 4го порядка.

**Приложение**

* Библиотеки:

from sympy import \*

import numpy as np

import pandas as pd

import random

import math

import matplotlib.pyplot as plt

import numpy as np

from math import \*

* Задание 1

Mass = np.random.normal(0, 1, 10\*\*3)

print("Числел в массиве: ", len(Mass))

Mass[:100]

Mass = np.random.normal(0, 1, 10\*\*3)

plt.figure(figsize=(10, 6))

plt.grid()

plt.plot(Mass, linewidth=0.5, c='red')

* Задание 2

X,W,N,T = [i/1000 for i in range(4000)],[],1000,4

Delta = T/N

W = [(np.random.normal(0, 1, 1)\*sqrt(X[i]))[0] for i in range(len(X))]

plt.figure(figsize=(10, 5.5))

plt.grid()

plt.plot(X, W, linewidth=0.3, c='red')

* Задание 3

import matplotlib.pyplot as plt

import numpy as np

from math import \*

import random

fig, ax = plt.subplots()

line, = ax.plot([])

def f():

N = 4002

W = [[-1, 1][random.randrange(2)] for i in range(N)]

Res = [sum(W[:i])/sqrt(N) for i in range(1, len(W))]

return Res

X = [i/4000 for i in range(4001)]

W = f()

for i in range(len(W)):

line.set\_data(range(i+1), W[:i+1])

ax.set\_xlim(0, len(W))

ax.set\_ylim(min(W), max(W))

line.set\_linewidth(0.5)

line.set\_color("Red")

plt.draw()

plt.pause(1000\*\*(-100))

* Задание 4

class Trapezoid:

def trapezoid(self, f, a, b, epsilon):

eps=epsilon

count=0

while eps%10!=0:

count+=1

eps\*=10

m = self.find\_n(f, a, b, epsilon)

X = np.arange(a, b + (b - a) / m, (b - a) /m)

# print(X)

s = 0

for i in range(len(X)):

if (i == 0) or (i == len(X) - 1):

s += (b - a) / (2 \* m) \* (self.f(X[i]))

else:

s += 2 \* (b - a) / (2 \* m) \* (self.f(X[i]))

return s

def dif\_func(self, f, n):

func = diff(f, x, n)

return func

def f(self, x):

return 0.05\*x-(x\*\*2-3\*x)\*\*2/(2\*100\*100)

def max(self, f, a, b):

X = np.arange(a, b, 0.0001)

Max\_f = self.f(a)

for i in X:

if self.f(i) > Max\_f:

Max\_f = self.f(i)

return Max\_f

def find\_n(self, f, a, b, epsilon):

m = symbols('m', real=True)

n = max(solve((b - a) \*\* 3 \* self.max(self.dif\_func(f, 2), a, b) / (12 \* m \*\* 2) - epsilon\*\*2, m))

# print(n)

if int(n) < n:

n = int(n)

n += 1

else:

n = int(n)

return n

class Monte\_Karlo:

def find\_n(self,epsilon,delta):

return int(1/(4\*epsilon\*\*2\*delta))+1

def monte\_carlo(self,t,n):

n=int(n)

ravn = np.random.uniform(0,t,n)

s=math.sqrt(t)\* np.random.normal(0, 1, 1)/n\*np.sum(np.absolute(ravn\*ravn - 3\*ravn)/100)

return s

x = symbols('x')

f = 0.05\*x-(x\*\*2-3\*x)\*\*2/(2\*100\*100)

T = Trapezoid()

MK=Monte\_Karlo()

times = [0.5,1.0,1.5,2.0,2.5,3.0,3.5,4.0]

Trap=[]

for t in times:

Trap+=[T.trapezoid(f,0,t,0.01)]

Mont=[]

N=MK.find\_n(0.01,0.05)

for j in range(1,9):

Nj=j/8\*N

Mont+=[MK.monte\_carlo(times[j-1],Nj)[0]]

print('Метод трапеций:',np.around(np.array(Trap),4))

print('Количество итераций метода трапеций:',T.find\_n(f,0,4,0.01))

print("Метод Монте-Карло: ",np.around(np.array(Mont),4))

print('Количество итераций метода Монте-Каро:',MK.find\_n(0.01,0.05))

S=[]

for i in range(len(Mont)):

S+=[20\*math.exp(Trap[i]+Mont[i])]

print('Цена БА:',np.around(np.array(S),4))

* Задание 5

NewTrap=[]

f = 0.05\*x

for t in times:

NewTrap+=[20\*math.exp(T.trapezoid(f,0,t,0.01))]

print("Облигация",np.around(np.array(NewTrap),4))

print('Количество итераций метода трапеций:',T.find\_n(f,0,4,0.01))

plt.figure(figsize=(10, 4.5))

plt.grid()

plt.plot(S, color="red",linewidth = 1)

* Задание 6

def func(x,St):

res = 0.05\*St\*x+abs(t\*t-3\*t)/100\*St\*math.sqrt(x)\*np.random.normal(0, 1, 1)[0]

return res

N=1000

h=4/N

Rk\_X=[20]

T=[0]

for i in range(N):

k1X=h\*(func(T[i],Rk\_X[i]))

k2X=h\*(func(T[i],Rk\_X[i]+k1X/2))

k3X=h\*(func(T[i],Rk\_X[i]+k2X/2))

k4X=h\*(func(T[i],Rk\_X[i]+k3X))

Rk\_X+=[Rk\_X[i]+(k1X+2\*k2X+2\*k3X+k4X)/6]

T+=[T[i]+h]

k=0

T=np.round(T,4)

Ba=[]

Rk=[]

for i in range(len(T)):

if(T[i] in times):

Ba+=[round(S[k],4)]

Rk+=[round(Rk\_X[i],4)]

k+=1

res = pd.DataFrame({'Базовый актив':Ba, "Рунге-Кутты" : Rk})

res