**Моделирование двумерного винеровского процесса**

**Задание:**

1. На интервале смоделируйте траекторий двумерного винеровского процесса интенсивности с шагом и шагом
2. Выведите на печать 5-7 траекторий (каждую на отдельном рисунке c шагом и шагом , мультимедийность приветствуется)
3. Для каждой траектории вычислите
4. вариации компонент

Найдите среднее значение вариации по всем траекториям

1. суммы квадратов приращений компонент

Найдите среднее значение этих сумм

1. Уменьшите значение в два раза и вычислите и . Сравните полученные значения для исходного и уменьшенного шага и объясните результат.

**Замечание**. Для параллельного моделирования траектории с шагом и шагом надо изначально сгенерировать пар СВ с независимыми компонентами, распределенными по закону . Тогда при моделирования винеровского процесса с шагом в качестве приращений используем суммы вида , компоненты которых распределены по свойствам нормального закона.

1. Вычислите теоретическую вероятность и сравните ее с эмпирической вероятностью достижения указанного уровня в момент .

**Исходные данные:**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Вар** |  |  |  |  |  |
| **15** | 14 | 100 | 0.25 | 0.1 | 1.5 |

**Решение:**

1. На интервале смоделируем 100 траекторий двумерного винеровского процесса интенсивности с шагом и шагом

*Алгоритм:*

1. Полагаем

trajectory\_h = np.zeros((n, N + 1, 2))

trajectory\_h2 = np.zeros((n, 2\*N + 1, 2))

1. Смоделируем пар с независимыми компонентами, распределенными по закону .

for i in range(0, n):

    for j in range(0, 2\*N):

        Ksi[i][j][0] = sps.norm(0, sigma \* np.sqrt(h/2)).rvs(size = 1)

        Ksi[i][j][1] = sps.norm(0, sigma \* np.sqrt(h/2)).rvs(size = 1)

1. Вычисляем – для шага *h*

for i in range(0, n):

    for j in range(1, N+1):

        trajectory\_h[i][j][0] = trajectory\_h[i][j - 1][0] + Ksi[i][2\*j - 2][0] + Ksi[i][2\*j - 1][0]

        trajectory\_h[i][j][1] = trajectory\_h[i][j - 1][1] + Ksi[i][2\*j - 2][1] + Ksi[i][2\*j - 1][1]

1. Вычисляем – для шага *h/2*

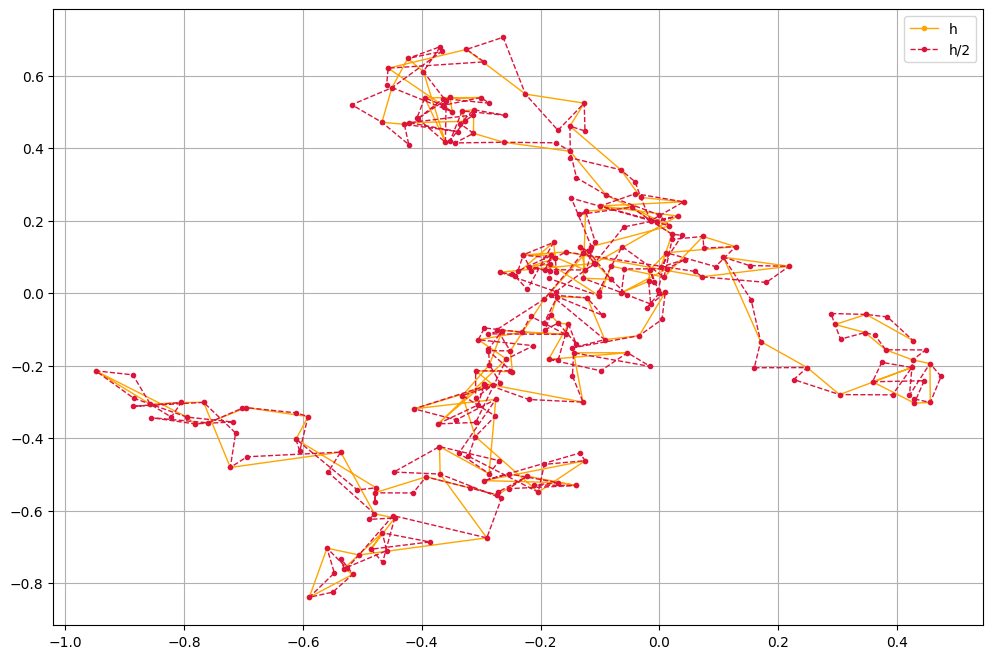
for i in range(0, n):

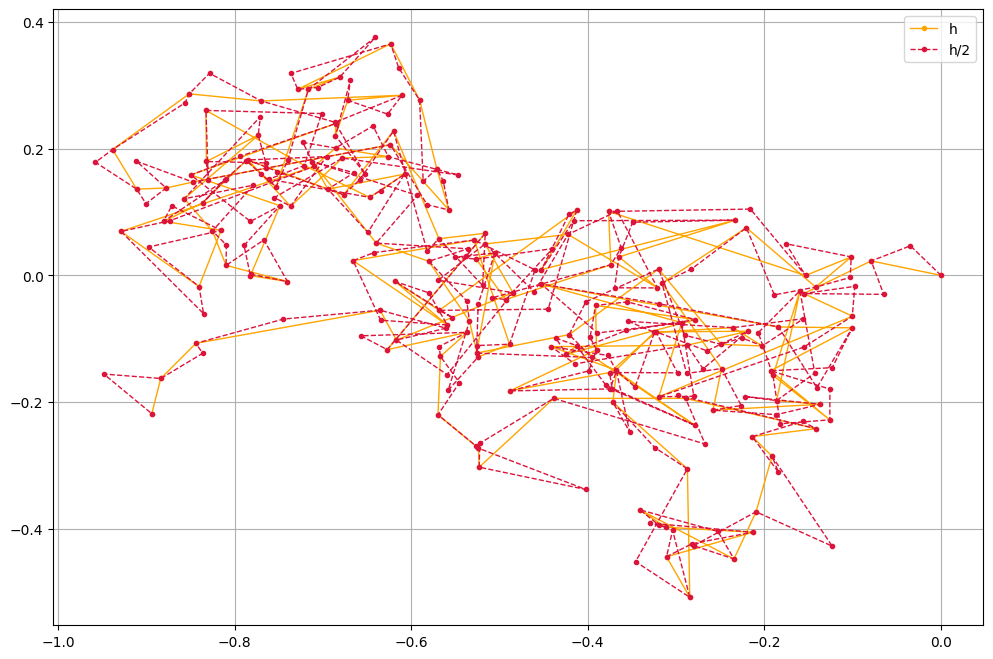
    for j in range(1, 2\*N+1):

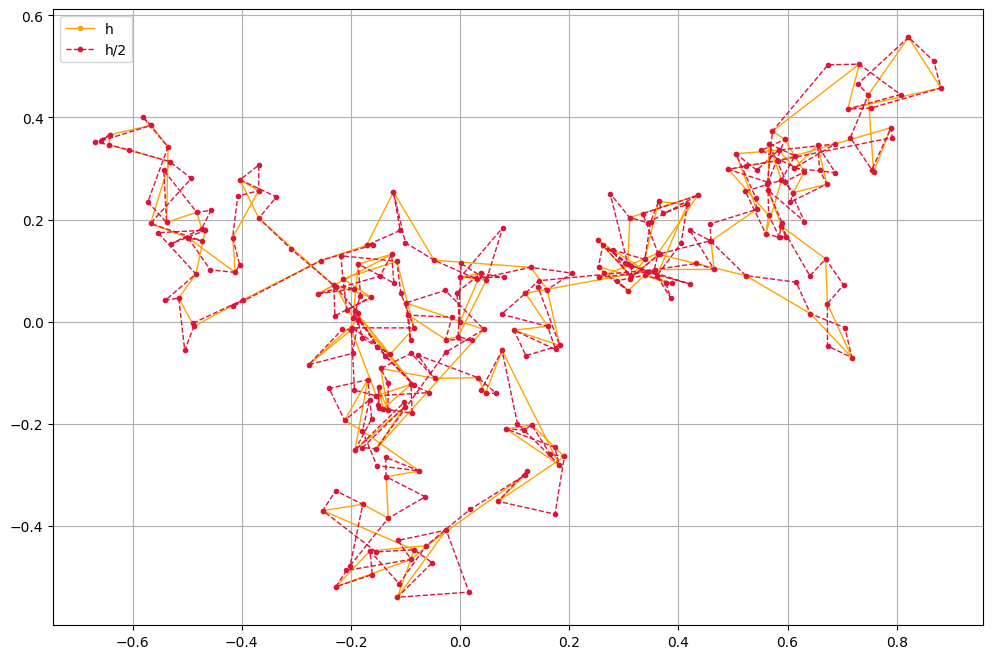
        trajectory\_h2[i][j][0] = trajectory\_h2[i][j - 1][0] + Ksi[i][j - 1][0]

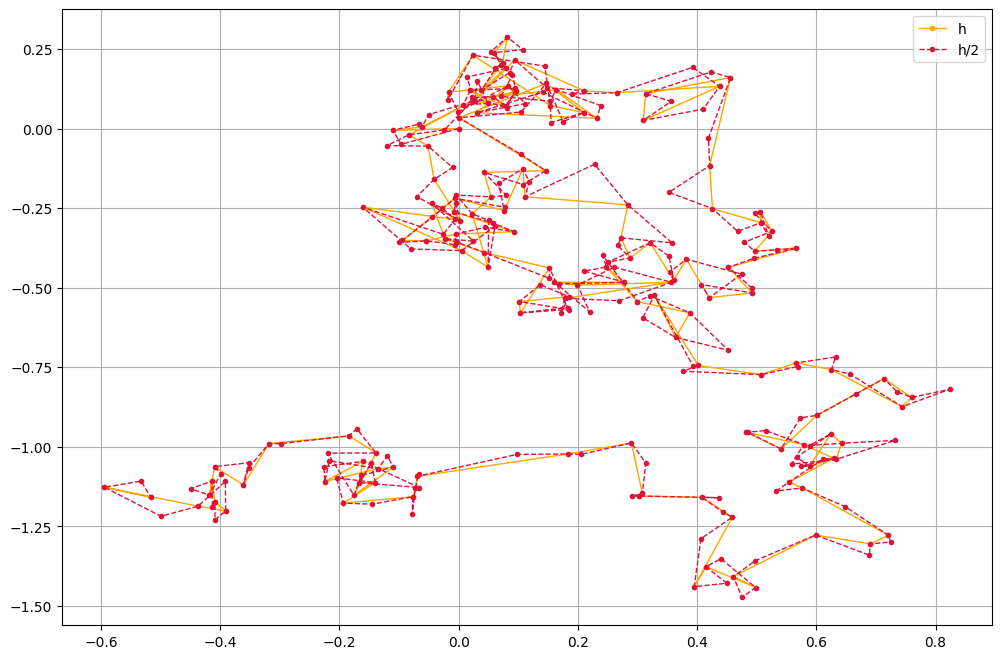
        trajectory\_h2[i][j][1] = trajectory\_h2[i][j - 1][1] + Ksi[i][j - 1][1]

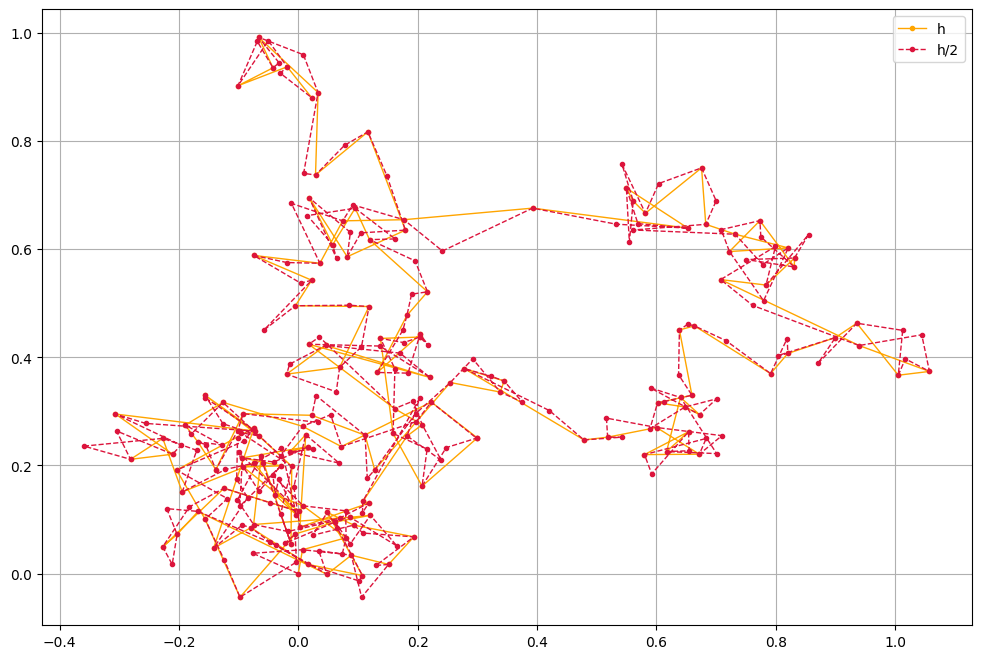
1. Результат – последовательность точек и последовательность точек . Соединив эти точки для наглядности отрезками прямых, получим смоделированные траектории.
2. Выведем на печать 5 траекторий (1-ю, 11-ю, 21-ю, 31-ю и 41-ю) c шагом и шагом :



****

****

****

****

1. Для каждой траектории вычислим:
2. вариации компонент

for i in range(0, n):

    for j in range(1, N+1):

      sum\_h[0] +=abs(trajectory\_h[i][j][0] -  trajectory\_h[i][j-1][0])

      sum\_h[1] +=abs(trajectory\_h[i][j][1] -  trajectory\_h[i][j-1][1])

(883.3189870605808, 883.2678183992446)

* Найдём среднее значение вариации по всем траекториям:

Var\_h[0] = sum\_h[0]/n

Var\_h[1] = sum\_h[1]/n

(8.833189870605807, 8.832678183992446)

1. суммы квадратов приращений компонент

for i in range(0, n):

    for j in range(1, N+1):

      sqSum\_h[0] +=abs(trajectory\_h[i][j][0] -  trajectory\_h[i][j-1][0])\*\* 2

      sqSum\_h[1] +=abs(trajectory\_h[i][j][1] -  trajectory\_h[i][j-1][1])\*\* 2

(87.14332678957055, 87.81468442735537)

* Найдём среднее значение этих сумм

SqVar\_h[0] = sqSum\_h[0]/n

SqVar\_h[1] = sqSum\_h[1]/n

(0.8714332678957055, 0.8781468442735537)

1. Уменьшим значение в два раза и вычислим и .

for i in range(0, n):

    for j in range(1, 2\*N+1):

      sum\_h2[0] +=abs(trajectory\_h2[i][j][0] -  trajectory\_h2[i][j-1][0])

      sum\_h2[1] +=abs(trajectory\_h2[i][j][1] -  trajectory\_h2[i][j-1][1])

      sqSum\_h2[0] +=abs(trajectory\_h2[i][j][0] -  trajectory\_h2[i][j-1][0])\*\* 2

      sqSum\_h2[1] +=abs(trajectory\_h2[i][j][1] -  trajectory\_h2[i][j-1][1])\*\* 2

Var\_h2 = np.zeros(2);

SqVar\_h2 = np.zeros(2);

Var\_h2[0] = sum\_h2[0]/n

Var\_h2[1] = sum\_h2[1]/n

SqVar\_h2[0] = sqSum\_h2[0]/n

SqVar\_h2[1] = sqSum\_h2[1]/n

(12.431052166915595, 12.524596093039133)

(0.8664176908643563, 0.8815478067731155)

* Сравним полученные значения для исходного и уменьшенного шага:

Таблица 1 – сравнение

|  |  |
| --- | --- |
|  |  |
| (8.833189870605807, 8.832678183992446) | (12.431052166915595, 12.524596093039133) |

Таблица 2 – сравнение

|  |  |
| --- | --- |
|  |  |
| (0.8714332678957055, 0.8781468442735537) | (0.8664176908643563, 0.8815478067731155) |

Полученные результаты соответствуют свойствам траекторий скалярного винеровского процесса:

1. Сумма квадратов приращений компонент стремится в средне квадратичном к (Таблица 2)

|  |  |
| --- | --- |
| 1. Сумма приращений компонент стремится в средне квадратичном к (Таблица 1) |  |

1. Вычислим теоретическую вероятность и сравним ее с эмпирической вероятностью достижения указанного уровня в момент .

* Эмпирическая вероятность:

amount\_z = 0

for i in range(0, n):

    if np.linalg.norm(trajectory\_h[i, N, :]) >= z:

        amount\_z += 1

amount\_z /= n

|  |
| --- |
|  |

* Теоретическая вероятность достижения уровня :

amount\_z\_T = 1 - sps.chi2(2).cdf((z \* z)/(sigma \* sigma \* T))

|  |  |
| --- | --- |
|  |  |

**Выводы:**

Мы научились моделировать и визуализировать траектории двумерного винеровского процесса интенсивности с шагом и шагом . Кроме того, мы вычислили вариации компонент, нашли среднее значение вариации по всем траекториям с шагом *h*, а также вычислили суммы квадратов приращений компонент и их среднее значение. То же самое мы проделали для траекторий с шагом *h/2*. В результате, сравнив полученные значения для траекторий с шагами *h* и *h/2*, получили значения, которые соответствуют двум свойствам траекторий скалярного винеровского процесса: сумма квадратов приращений компонент стремится в средне квадратичном к , а сумма приращений компонент – к бесконечности. Были вычислены теоретическая и эмпирическая вероятность достижения указанного уровня в момент . Они оказались близки по своим значениям.