Постановка задачи:

1. Написать функцию вычисления аналитического решения системы уравнений движения частицы. Нарисовать графики аналитических траекторий для различных параметров
2. Написать программу для решения уравнений движения частицы неявной схемой (10) (11) Нарисовать графики численных траекторий для различных параметров
3. вычислить и нарисовать графики точности численного решения от времени

Импорт:

In [2]:

```
from numpy import tan, pi, exp, array, arange, degrees, append
import matplotlib.pyplot as plt
```

Входные данные:

In [3]:

```
x0, z0 = 0, 0
vx0 = 1
```

Построение графика:

In [4]:

```
def plot(x, y, z, style1, style2, xlabel, ylabel, legend, title, alpha, beta):
    plt.plot(x, y, style1)
    plt.plot(x, z, style2)
    plt.xlabel(xlabel)
    plt.ylabel(ylabel)
    plt.legend(legend)
    plt.title(label=title + '\n' + str(f'alpha = {int(degrees(alpha))}') + str(f'    beta
    plt.grid(True)
    plt.show()
```

Рассчет значений:

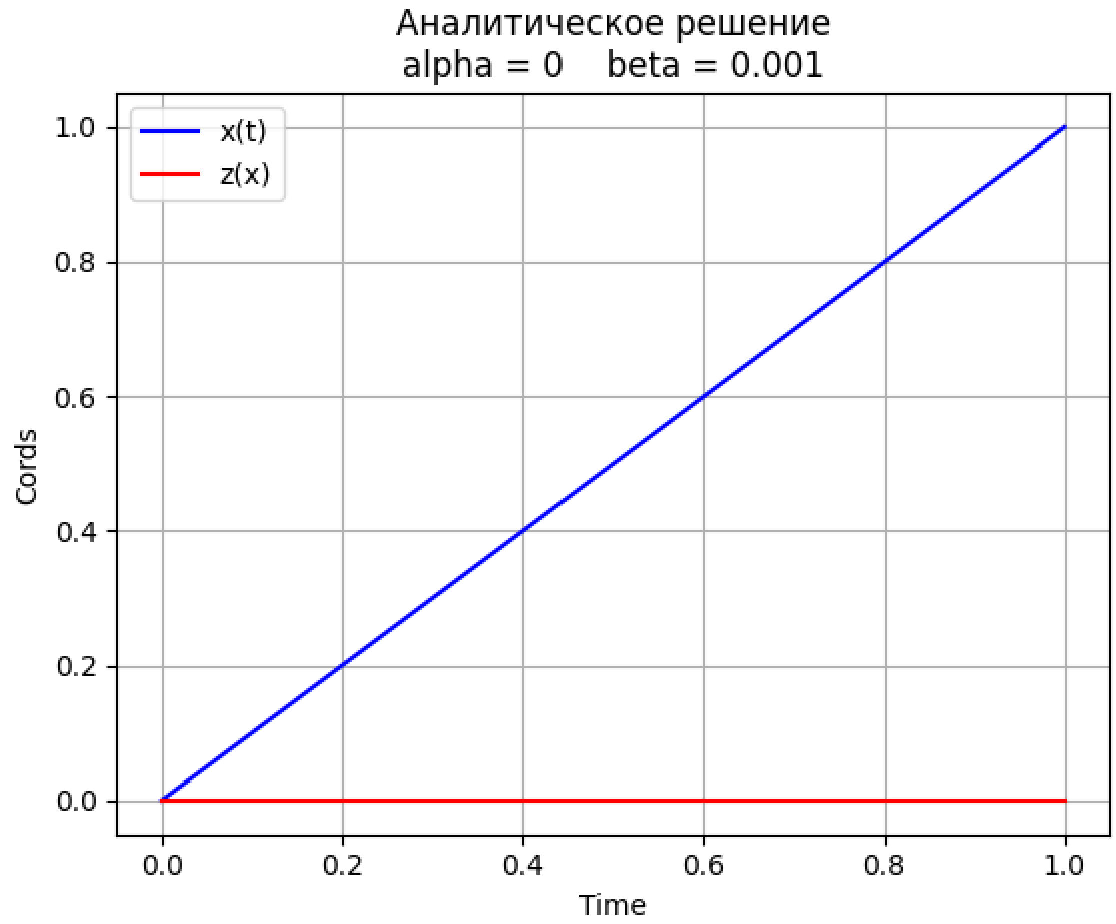
In [5]:

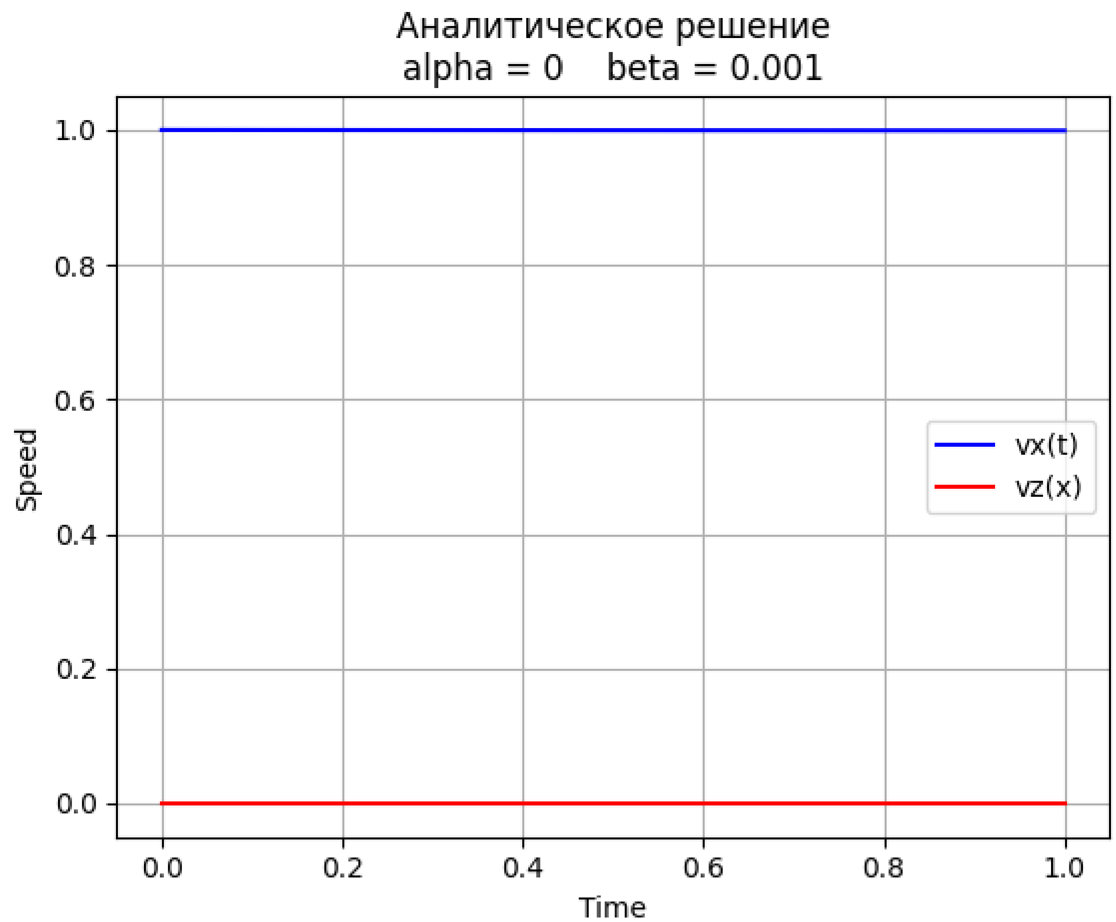
```
def func(alpha, beta):
    assert 0 <= alpha <= pi / 2, 'alpha < 0 or alpha > pi/2'
    assert beta != 0, 'beta = 0'
    vz0 = tan(alpha)
    vx = lambda t: exp(-beta * t)
    vz = lambda t: (vz0 + 2 * vz0 / beta) * exp(-beta * t) - 2 * vz0 / beta
    x = lambda t: (1 - exp(-beta * t)) / beta
    z = lambda t: (vz0 + 2 * vz0 / beta) * (1 - exp(-beta * t)) / beta - 2 * t * vz0 / be
    num_vx = [vx0]
    num_vz = [vz0]
    num_x = [x0]
    num_z = [z0]
    T = arange(x0, x0 + 1, beta)
    delta_t = 1e-3
    for _ in T:
        vxn = num_vx[-1] * (1 - beta * delta_t / 2) * (1 + beta * delta_t / 2)
        num_vx = append(num_vx, [vxn])
        vzn = (num_vz[-1] * (1 - beta * delta_t / 2) - 2 * vz0 * delta_t) / (1 + beta * d
        num_vz = append(num_vz, [vzn])

        xn = num_x[-1] + (num_vx[-2] + num_vx[-1]) * delta_t / 2
        num_x = append(num_x, [xn])
        zn = num_z[-1] + (num_vz[-2] + num_vz[-1]) * delta_t / 2
        num_z = append(num_z, [zn])
    T = append(T, [1])
    an_x = array(list(map(x, T)))
    an_z = array(list(map(z, an_x)))
    an_vx = array(list(map(vx, T)))
    an_vz = array(list(map(vz, T)))
    delta_x = array(list(map(lambda x: abs(x[0] - x[1]), zip(an_x, num_x))))
    delta_z = array(list(map(lambda x: abs(x[0] - x[1]), zip(an_z, num_z))))
    delta_vx = array(list(map(lambda x: abs(x[0] - x[1]), zip(an_vx, num_vx))))
    delta_vz = array(list(map(lambda x: abs(x[0] - x[1]), zip(an_vz, num_vz))))
    plot(T, an_x, an_z, 'b-', 'r-', 'Time', 'Cords', ['x(t)', 'z(x)'], 'Аналитическое реш
    plot(T, an_vx, an_vz, 'b-', 'r-', 'Time', 'Speed', ['vx(t)', 'vz(x)'], 'Аналитическое
    plot(T, num_x, num_z, 'b-', 'r-', 'Time', 'Cords', ['x(t)', 'z(x)'], 'Численное решен
    plot(T, num_vx, num_vz, 'b-', 'r-', 'Time', 'Speed', ['vx(t)', 'vz(x)'], 'Численное р
    plot(T, delta_x, delta_z, 'b-', 'r-', 'Time', 'Cords', ['delta_x(t)', 'delta_z(x)'],
    plot(T, delta_vx, delta_vz, 'b-', 'r-', 'Time', 'Speed', ['delta_vx(t)', 'delta_vz(x)
```

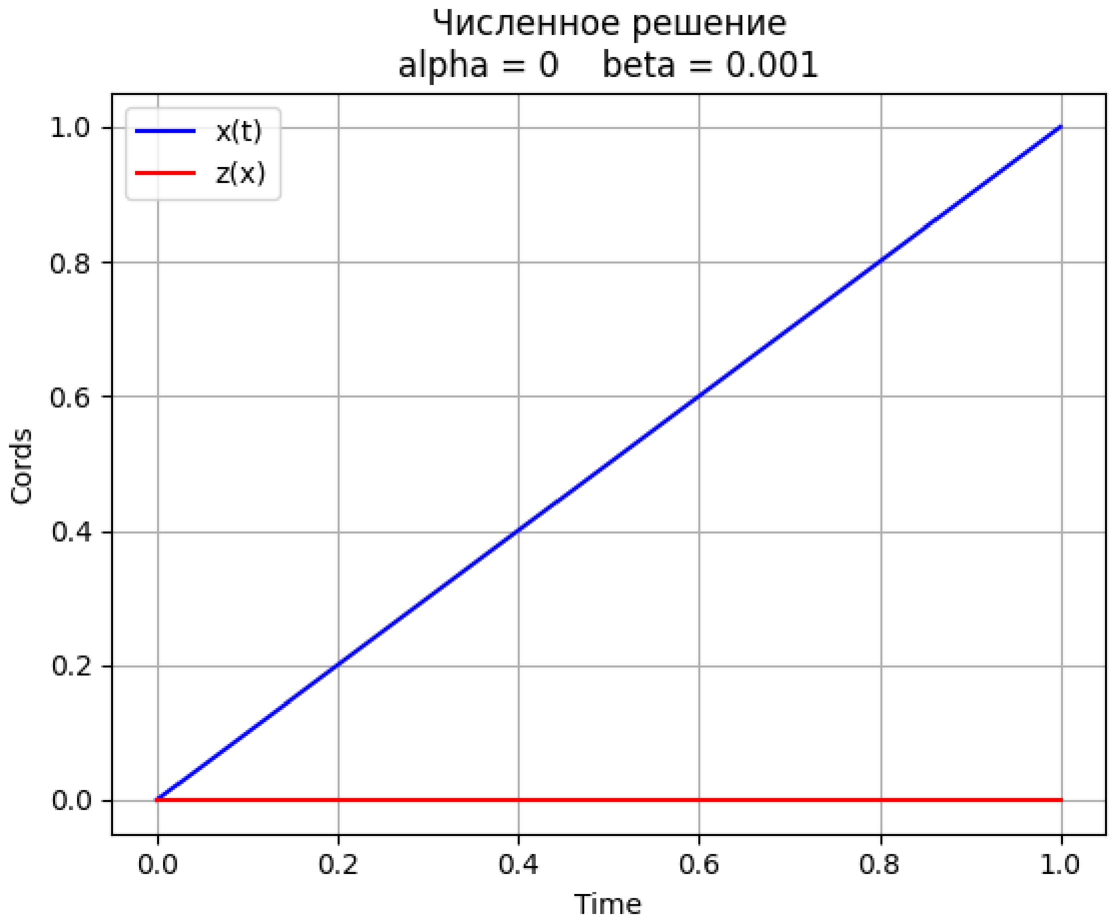
Вызов функции:

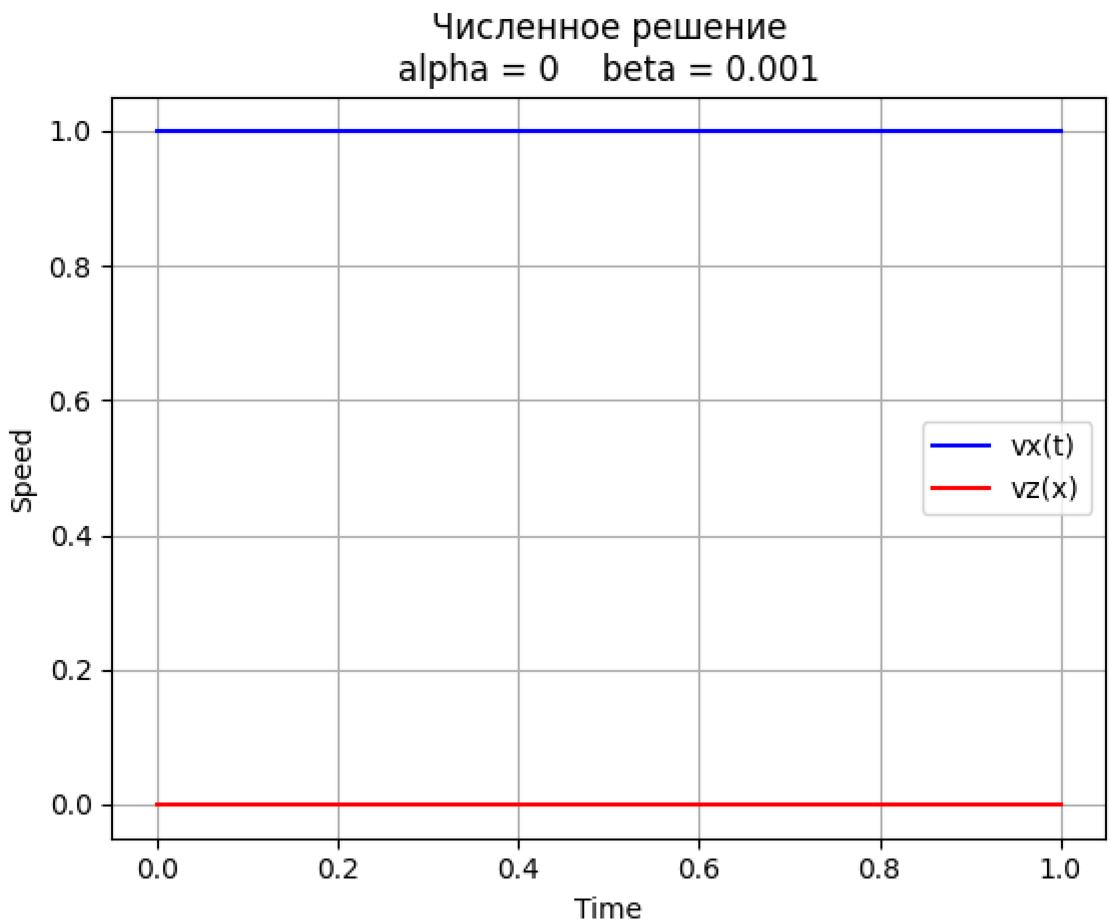
In [6]:

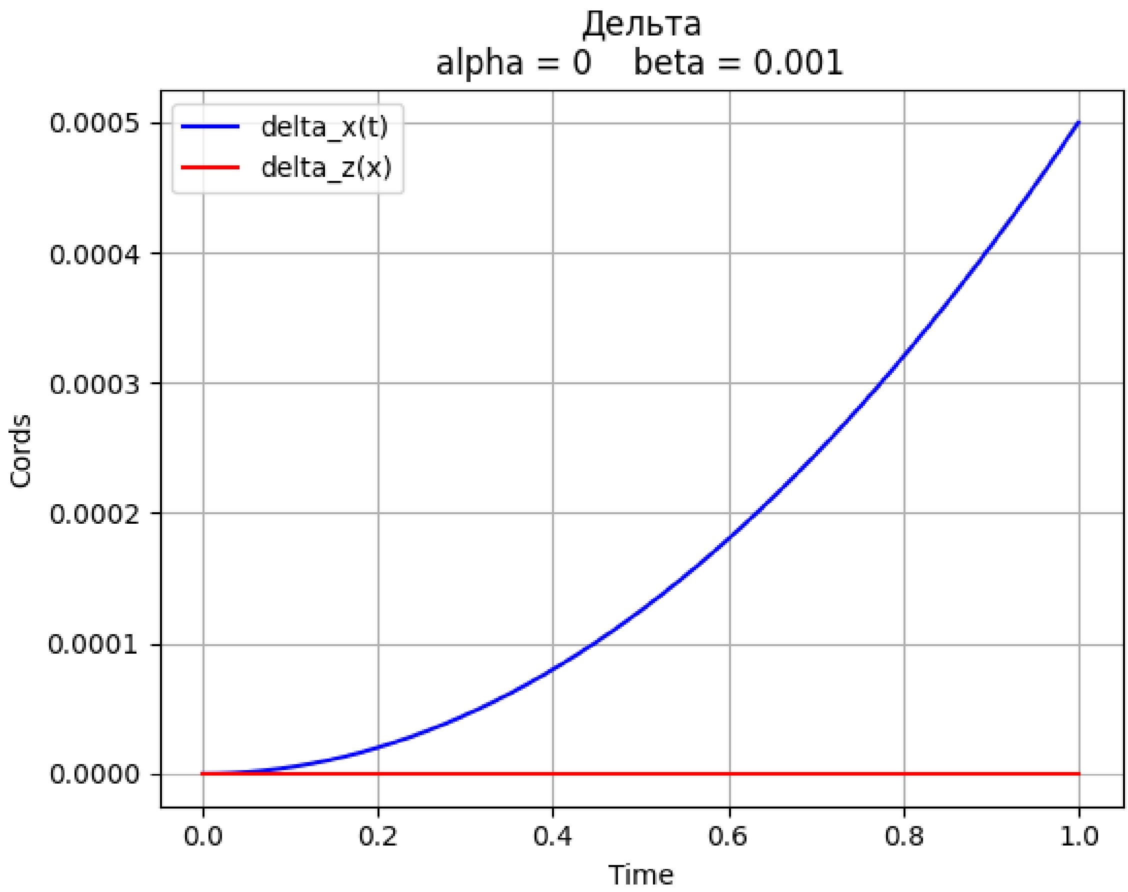
```
func(0, 1e-3)
```

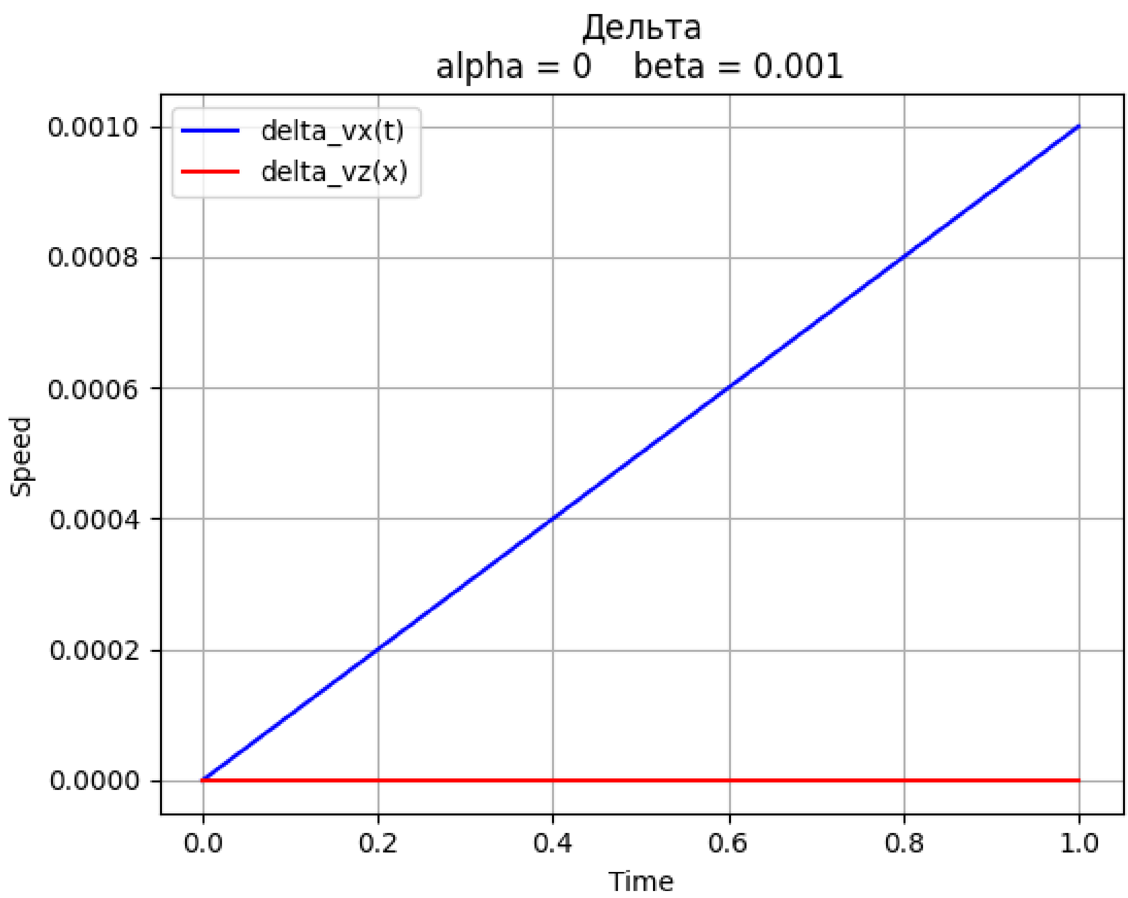





Численное решение
alpha = 0   beta = 0.001
x(t)
z(x)
Cords
Time
0.0
0.2
0.4
0.6
0.8
1.0

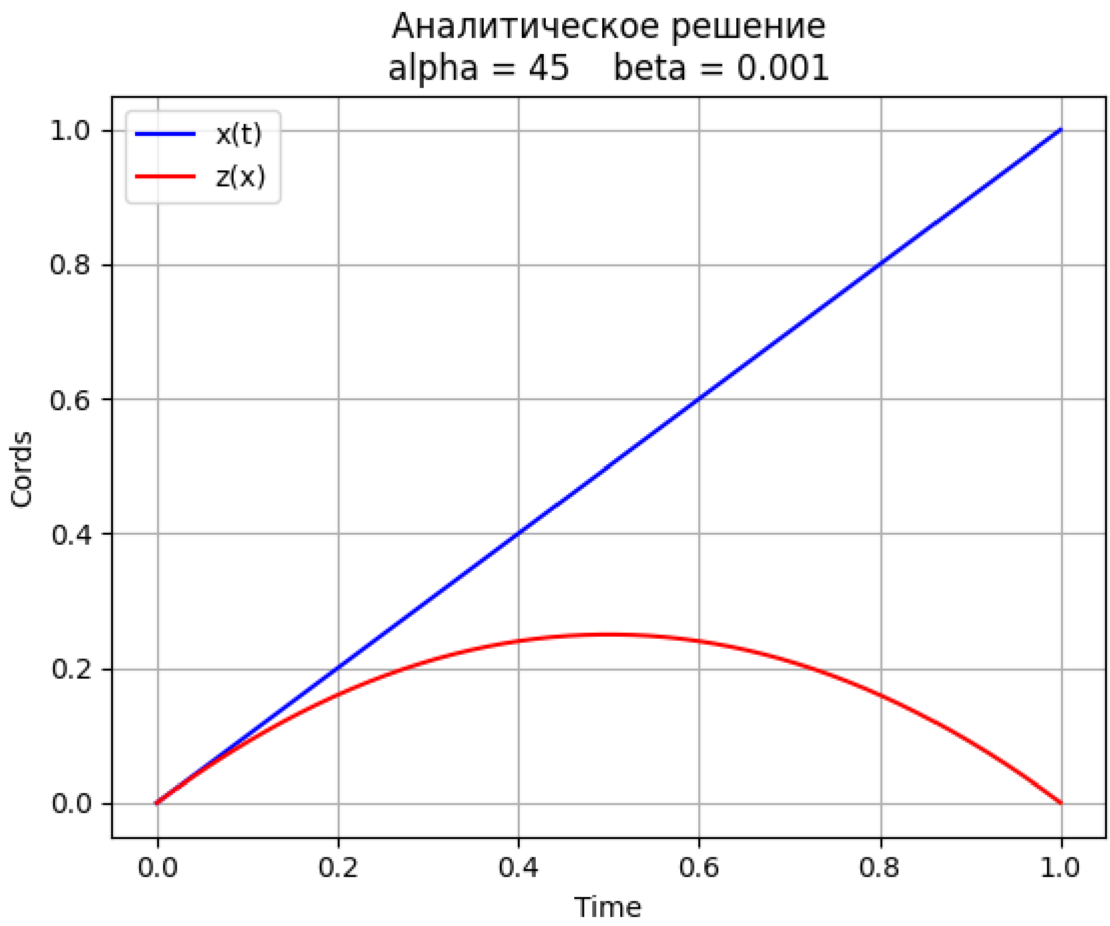

Численное решение
alpha = 0   beta = 0.001
vx(t)
vz(x)
Speed
Time
0.0
0.2
0.4
0.6
0.8
1.0

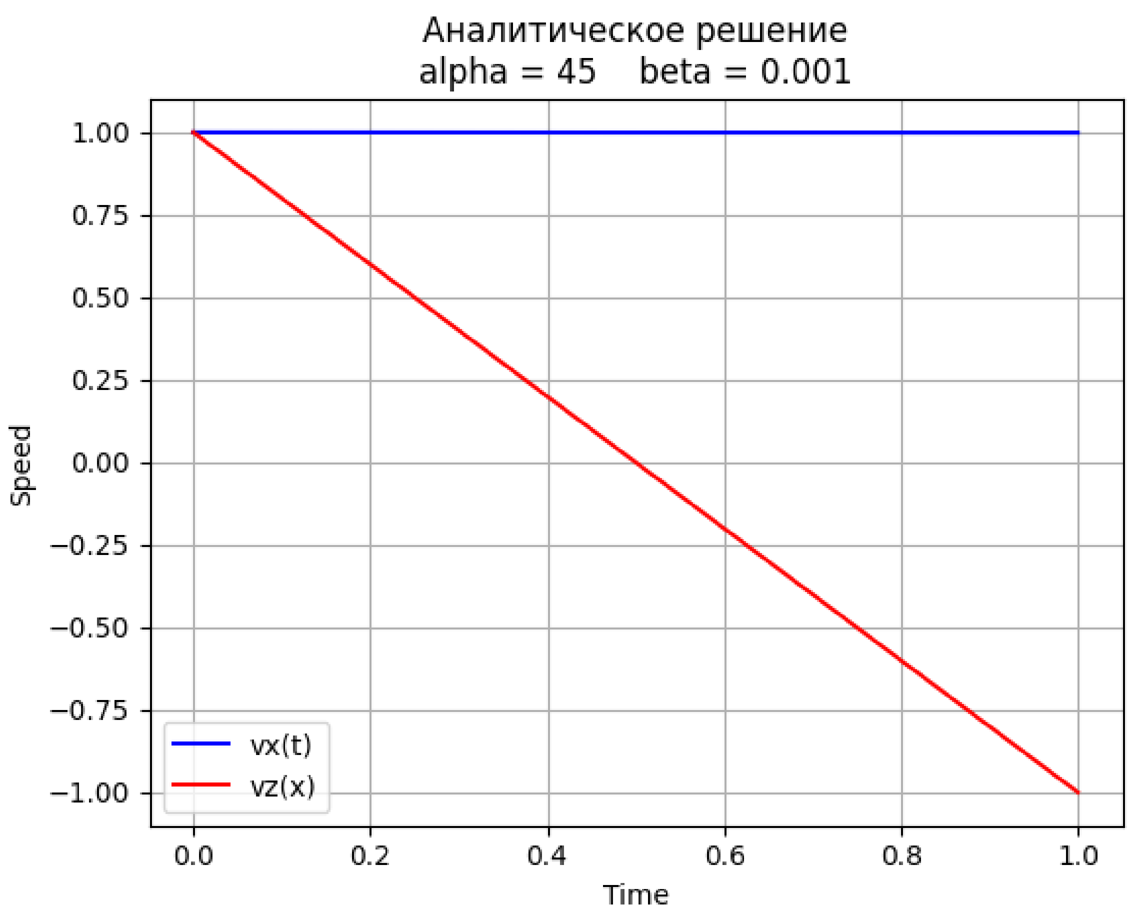

Дельта
alpha = 0 beta = 0.001
delta_x(t)
delta_z(x)
Cords
Time
0.0000
0.0001
0.0002
0.0003
0.0004
0.0005
0.0
0.2
0.4
0.6
0.8
1.0

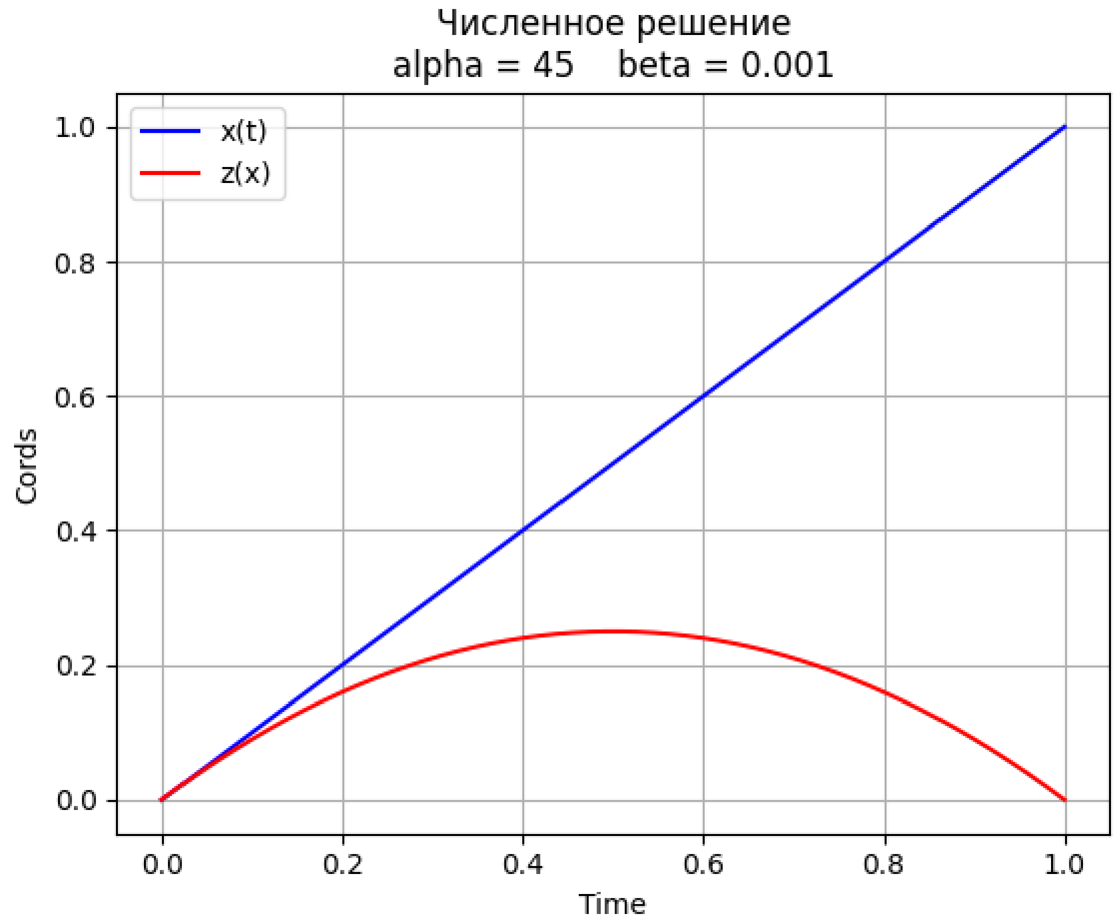

Дельта
alpha = 0 beta = 0.001
delta_vx(t)
delta_vz(x)
Speed
Time
0.0000
0.0002
0.0004
0.0006
0.0008
0.0010
0.0
0.2
0.4
0.6
0.8
1.0

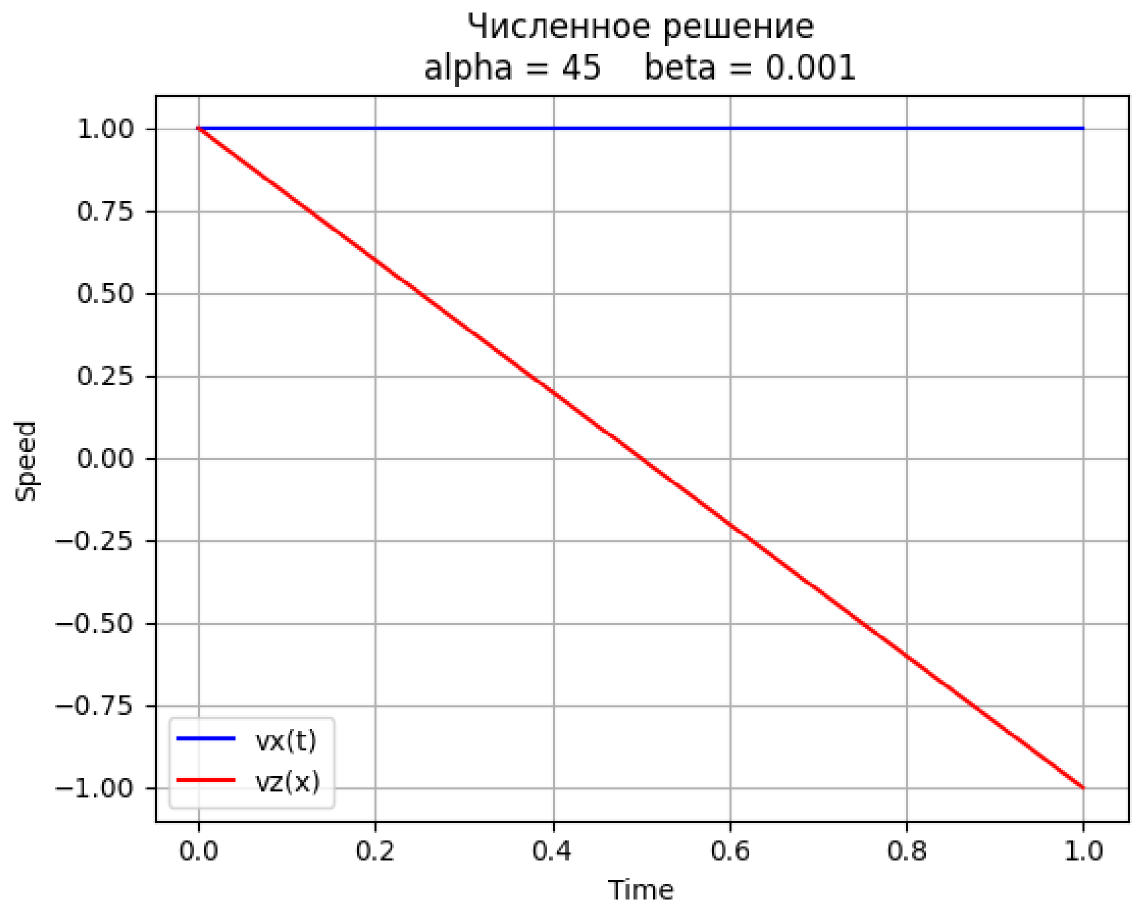
In [7]:

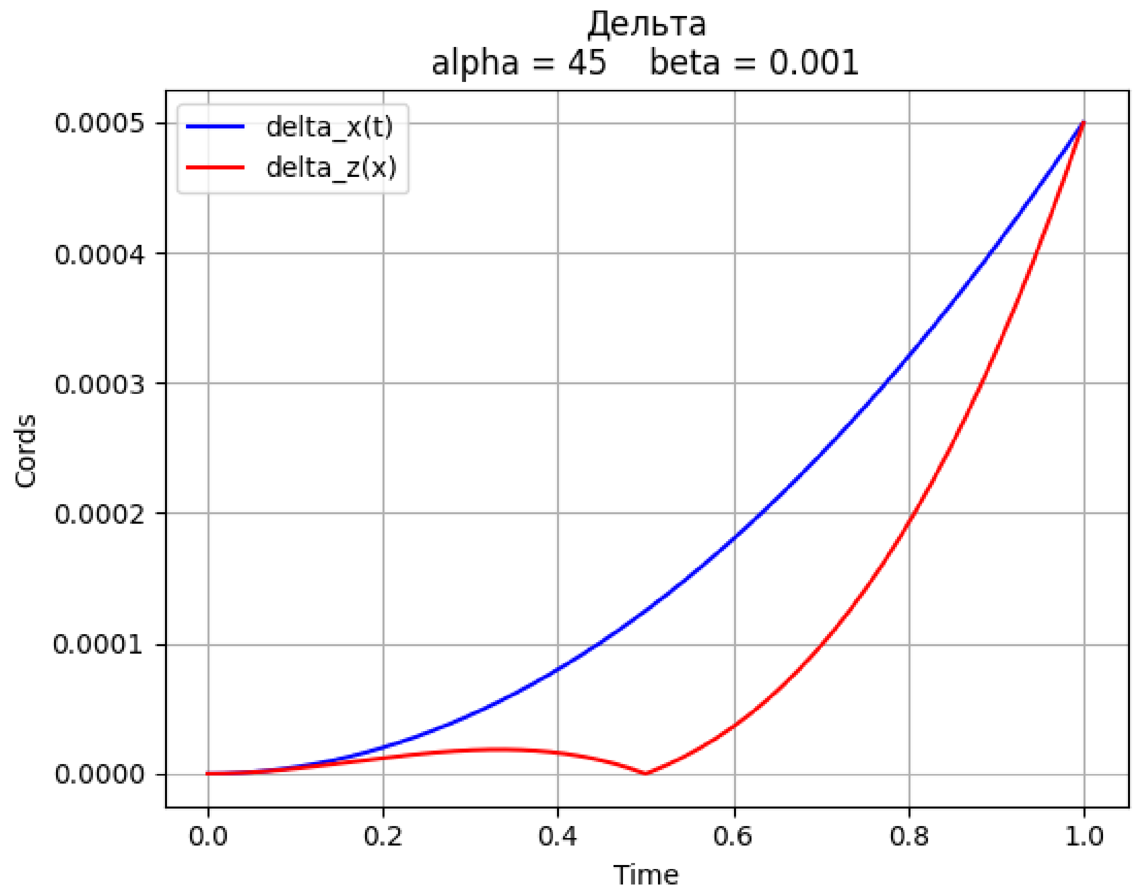
```
func(pi/4, 1e-3)
```

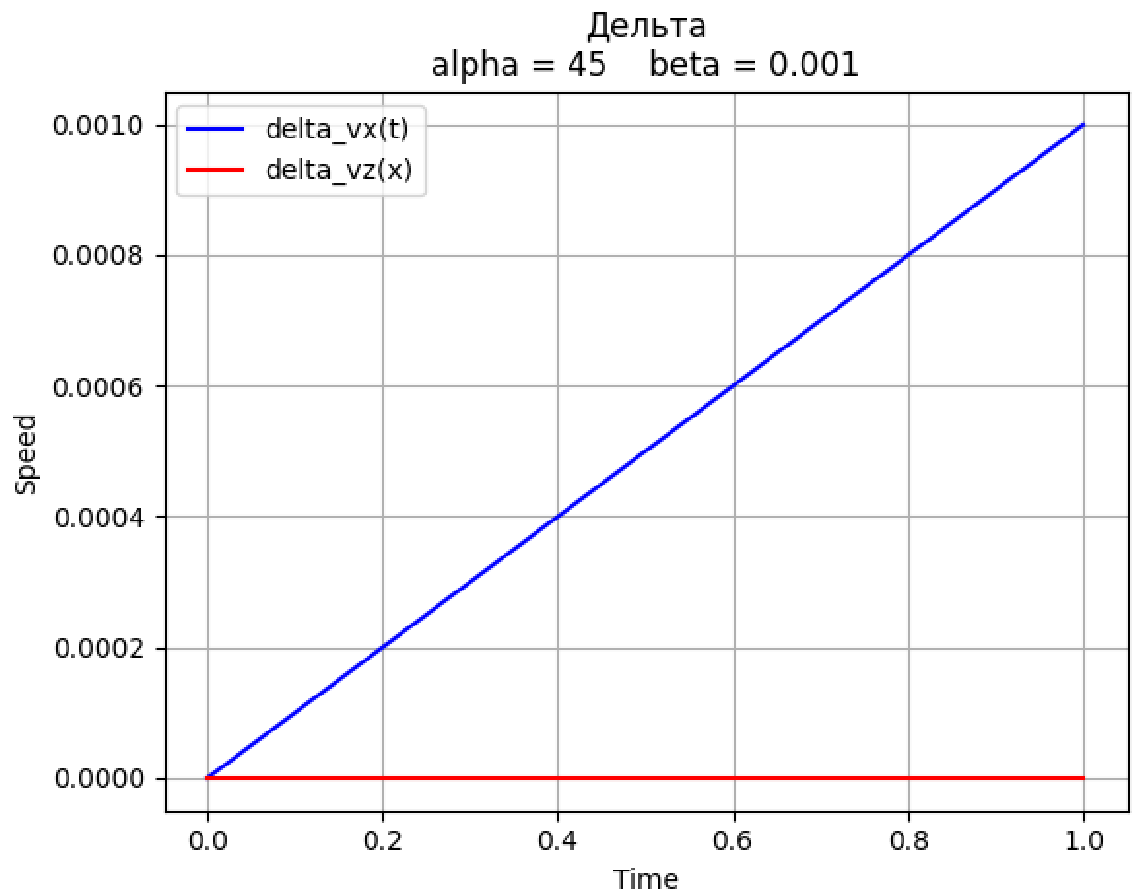





Численное решение
alpha = 45 beta = 0.001
x(t)
z(x)
Cords
Time
0.0
0.2
0.4
0.6
0.8
1.0

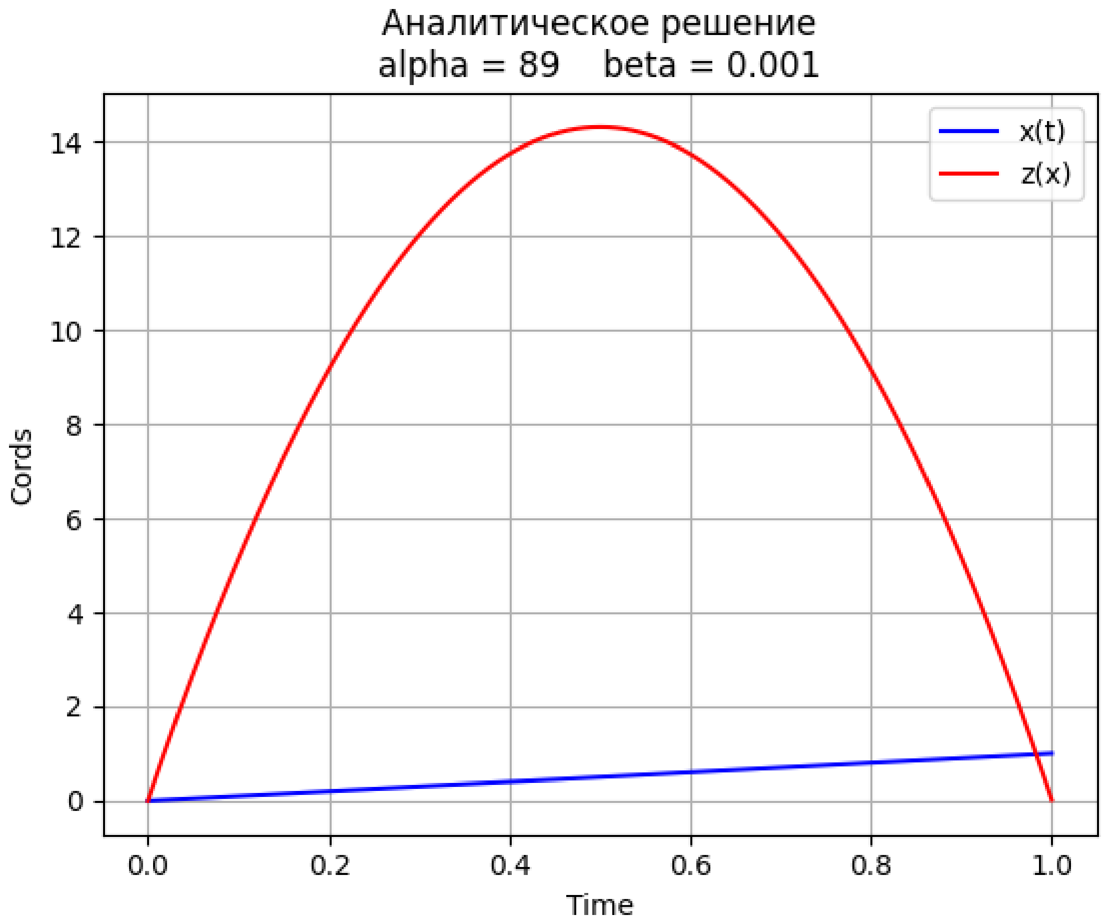

Численное решение
alpha = 45 beta = 0.001
vx(t)
vz(x)
Speed
Time
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75
−1.00
0.0
0.2
0.4
0.6
0.8
1.0

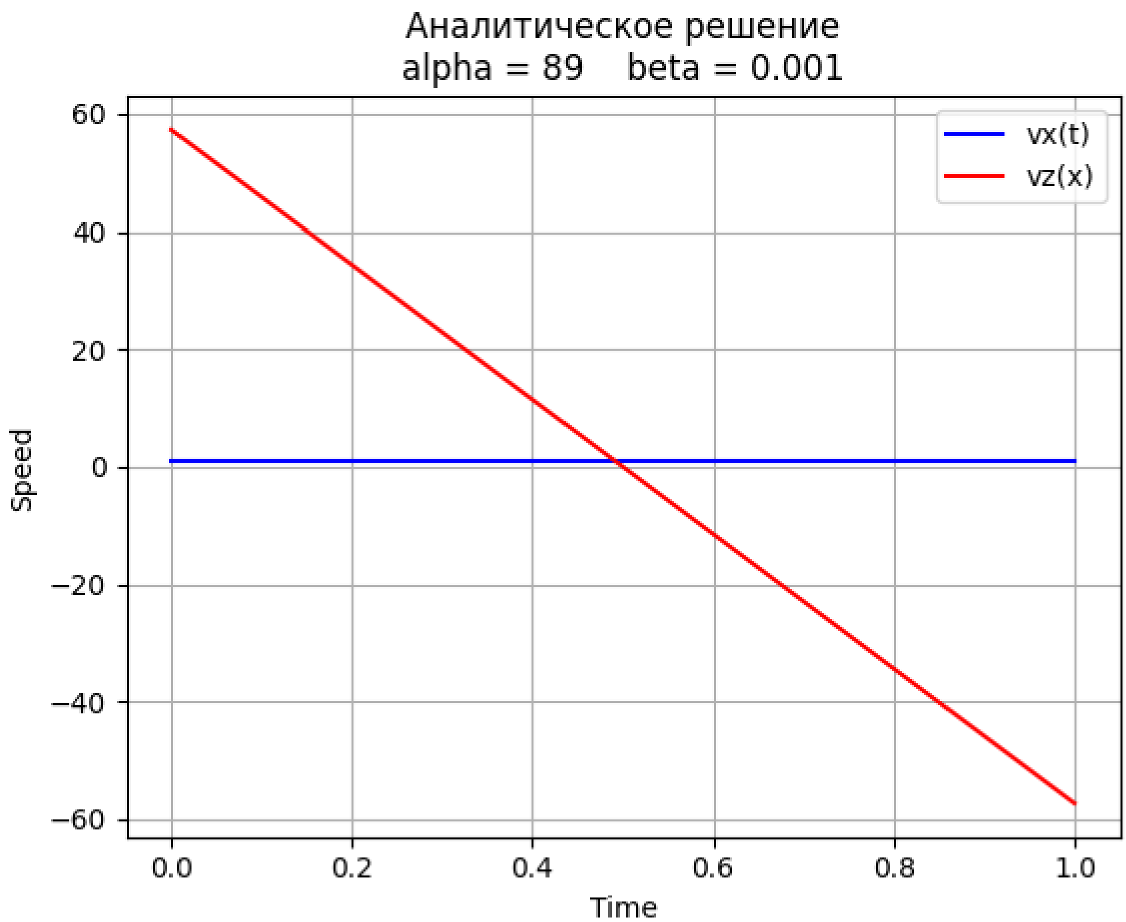

Дельта
alpha = 45 beta = 0.001
delta_x(t)
delta_z(x)
Cords
Time
0.0000
0.0001
0.0002
0.0003
0.0004
0.0005
0.0
0.2
0.4
0.6
0.8
1.0

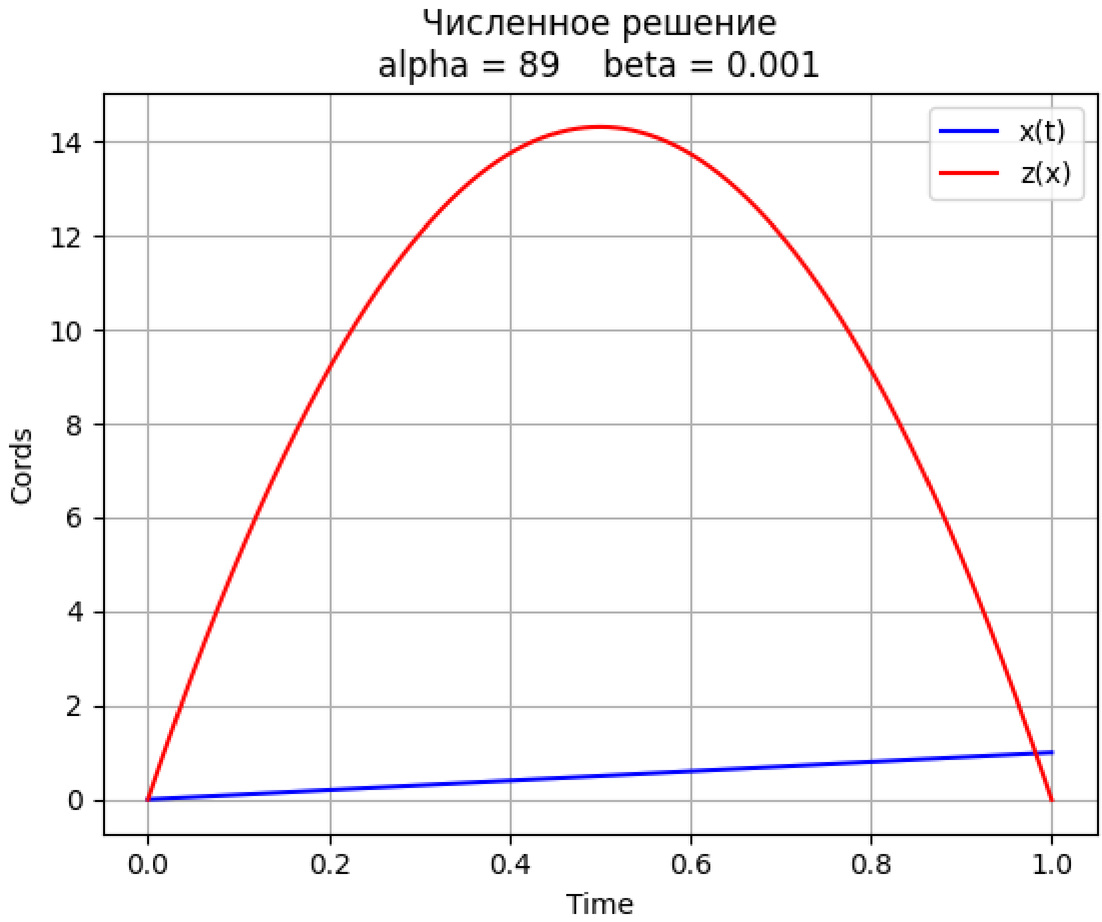

Дельта
alpha = 45 beta = 0.001
delta_vx(t)
delta_vz(x)
Speed
Time
0.0000
0.0002
0.0004
0.0006
0.0008
0.0010
0.0
0.2
0.4
0.6
0.8
1.0

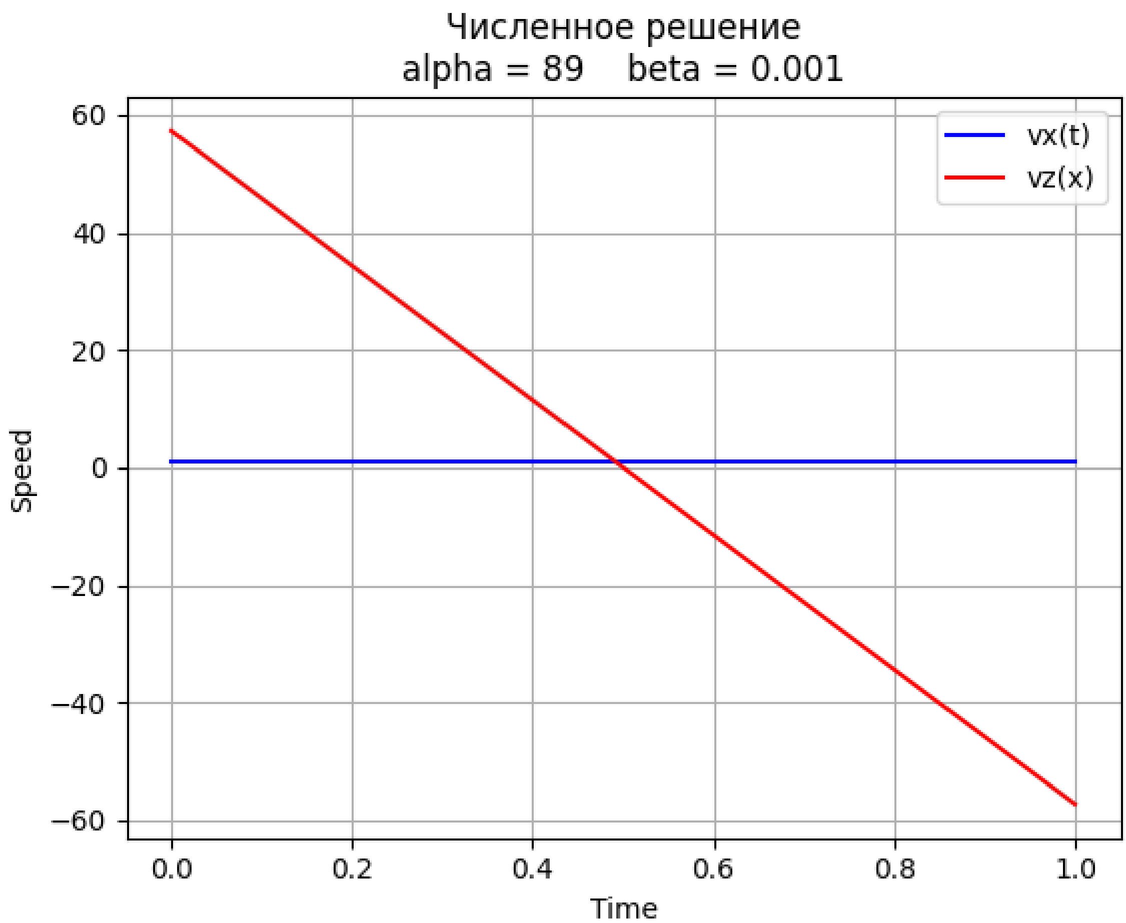
In [10]:

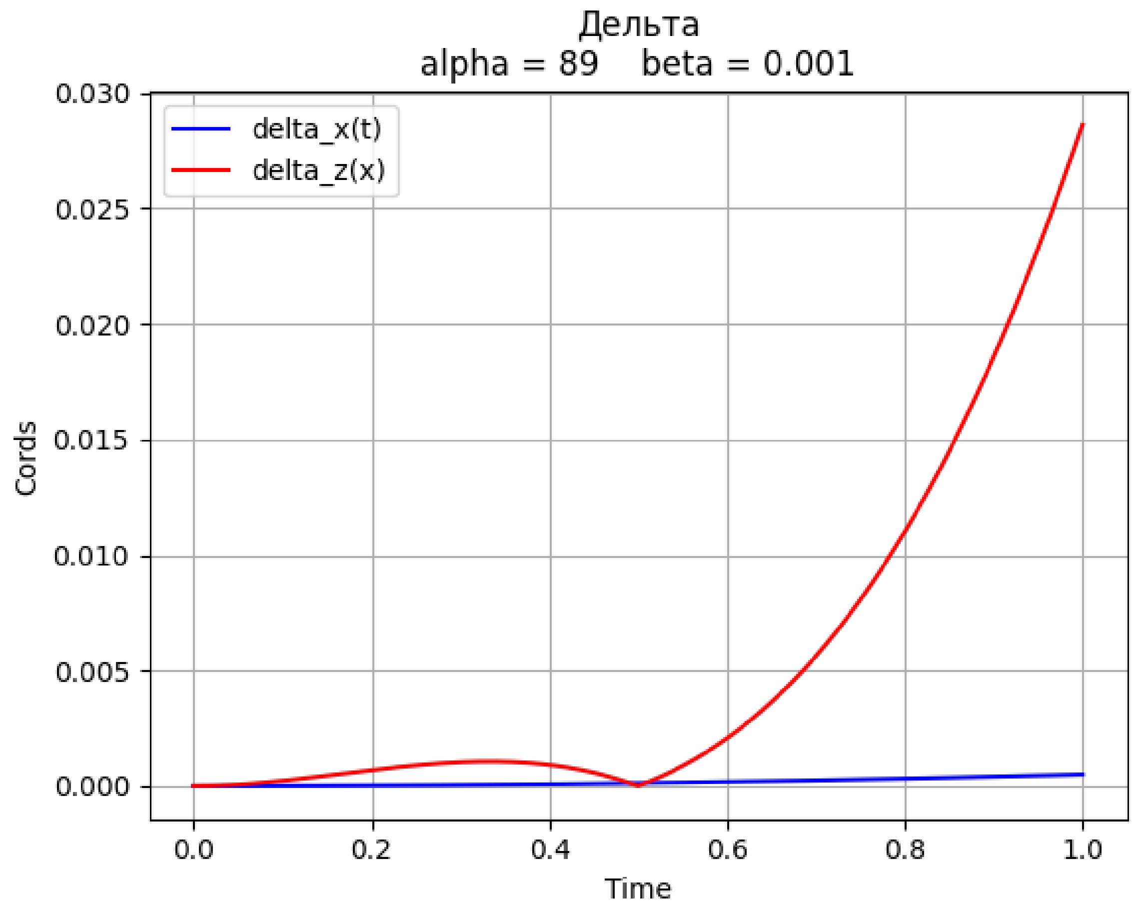
```
func(pi/(180/89), 1e-3)
```

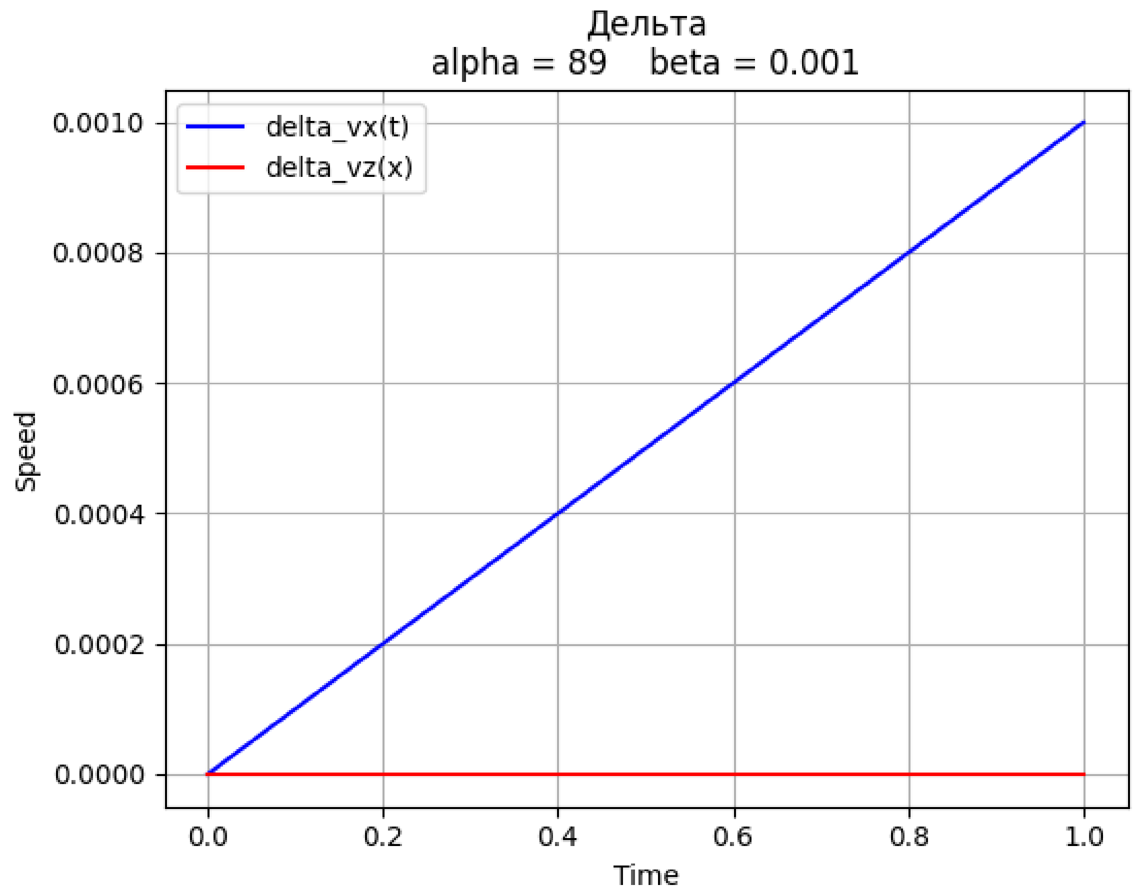





Численное решение
alpha = 89 beta = 0.001
x(t)
z(x)
Cords
Time
14
12
10
8
6
4
2
0
0.0
0.2
0.4
0.6
0.8
1.0

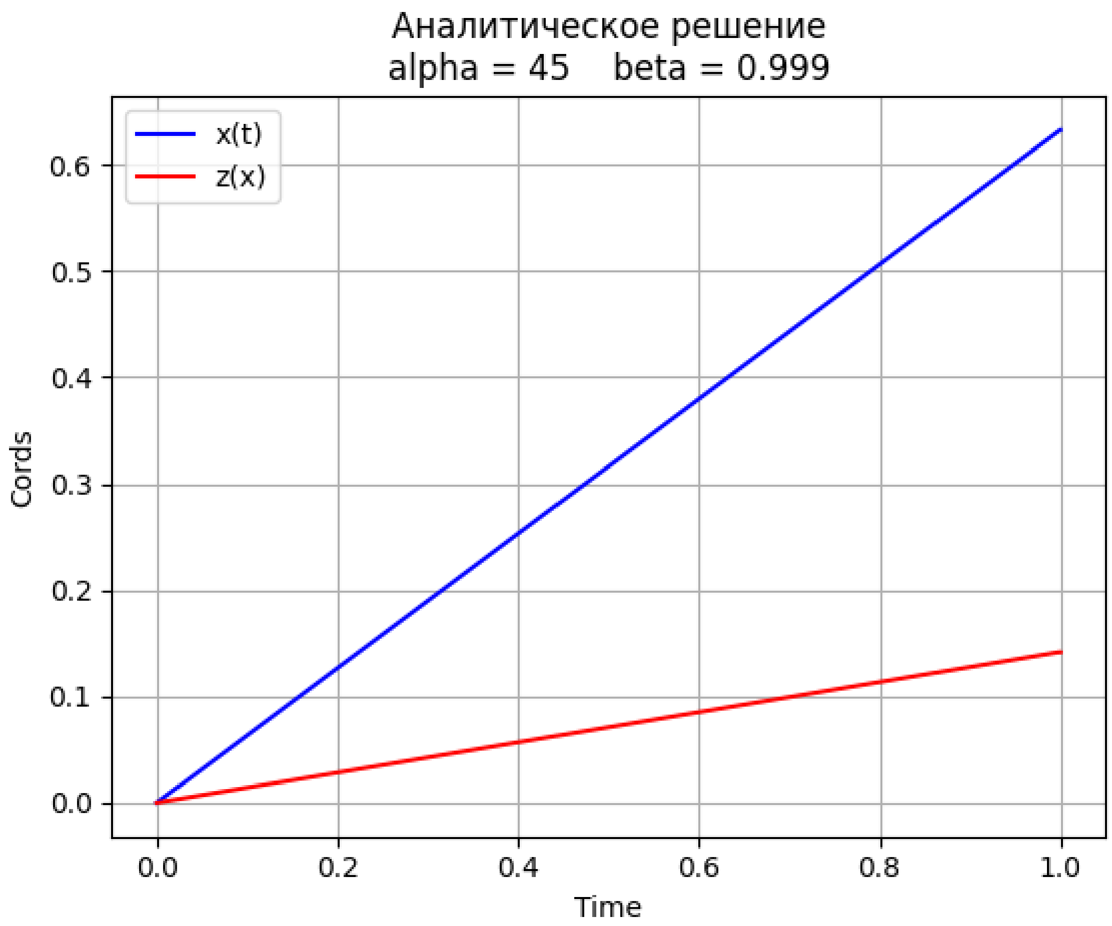

Численное решение
alpha = 89 beta = 0.001
vx(t)
vz(x)
Speed
Time
60
40
20
0
−20
−40
−60
0.0
0.2
0.4
0.6
0.8
1.0

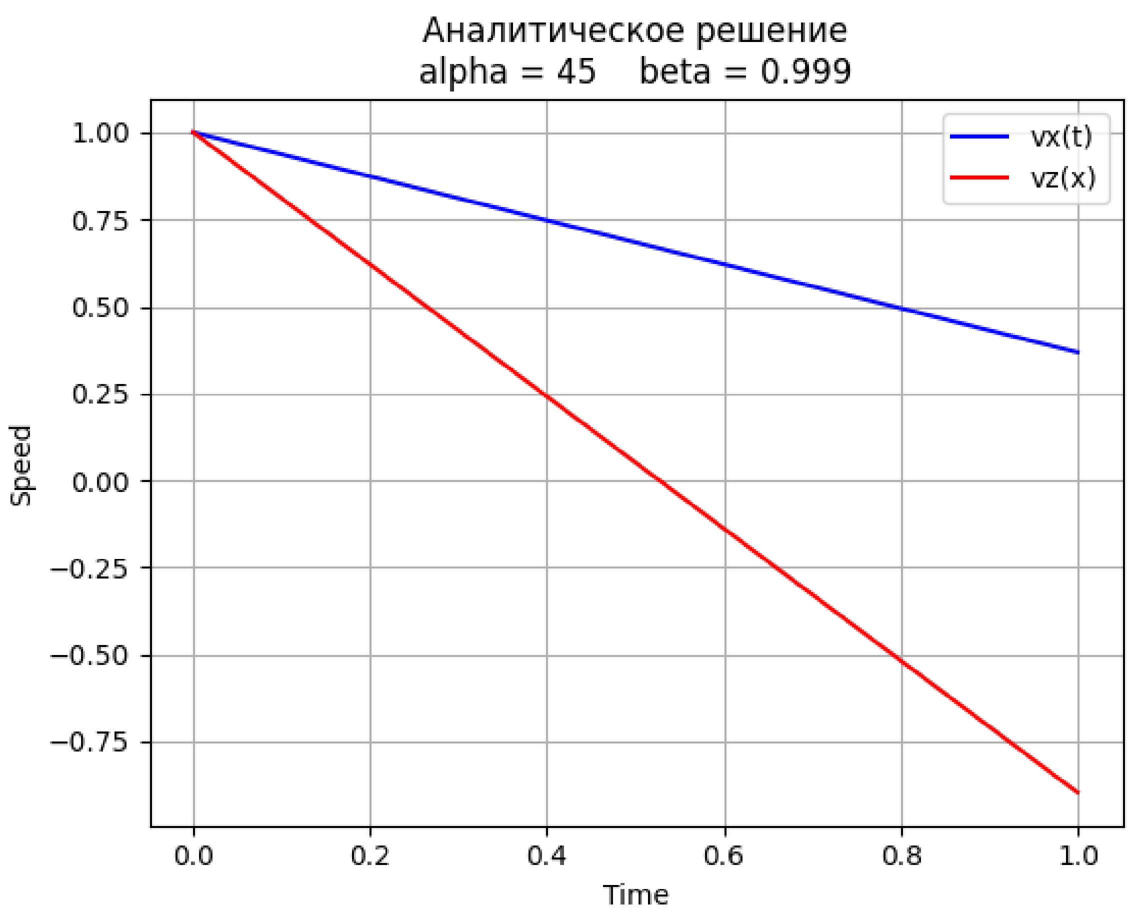

Дельта
alpha = 89 beta = 0.001
delta_x(t)
delta_z(x)
Cords
Time
0.030
0.025
0.020
0.015
0.010
0.005
0.000
0.0
0.2
0.4
0.6
0.8
1.0

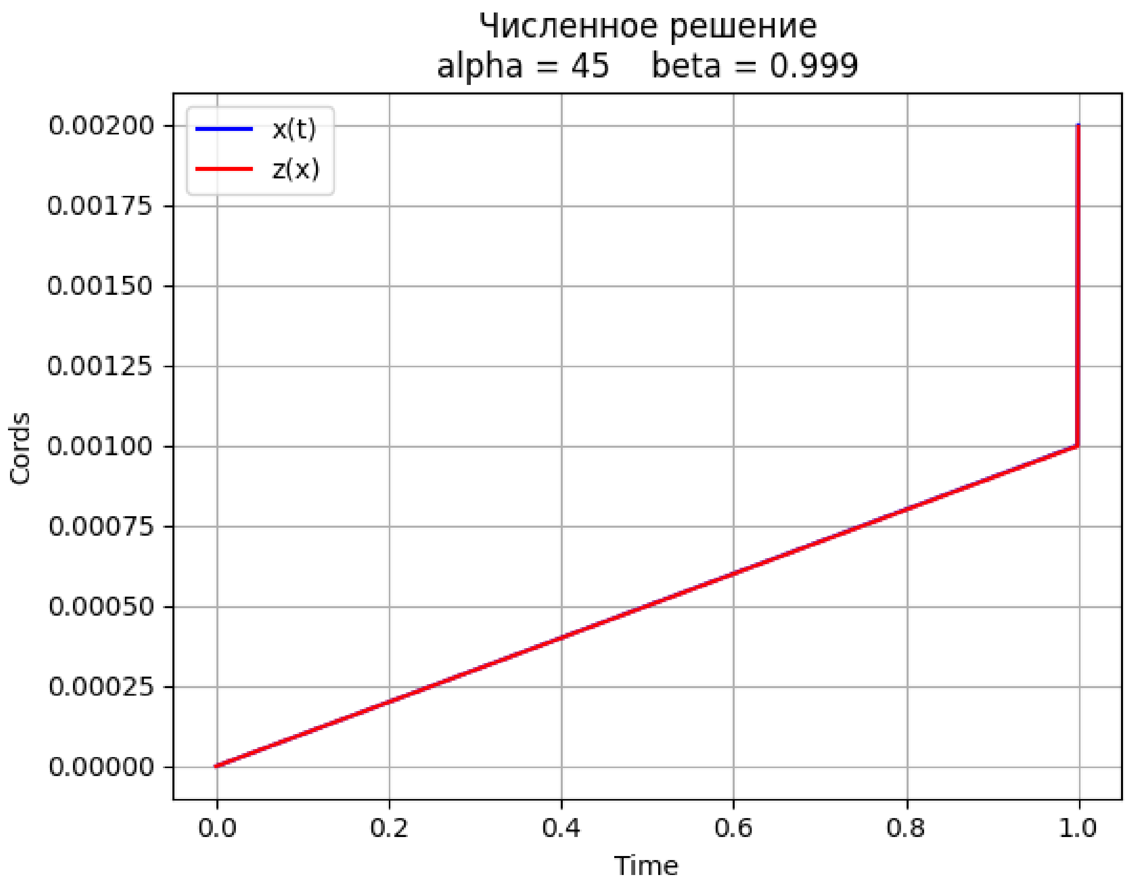

Дельта
alpha = 89 beta = 0.001
delta_vx(t)
delta_vz(x)
Speed
Time
0.0010
0.0008
0.0006
0.0004
0.0002
0.0000
0.0
0.2
0.4
0.6
0.8
1.0

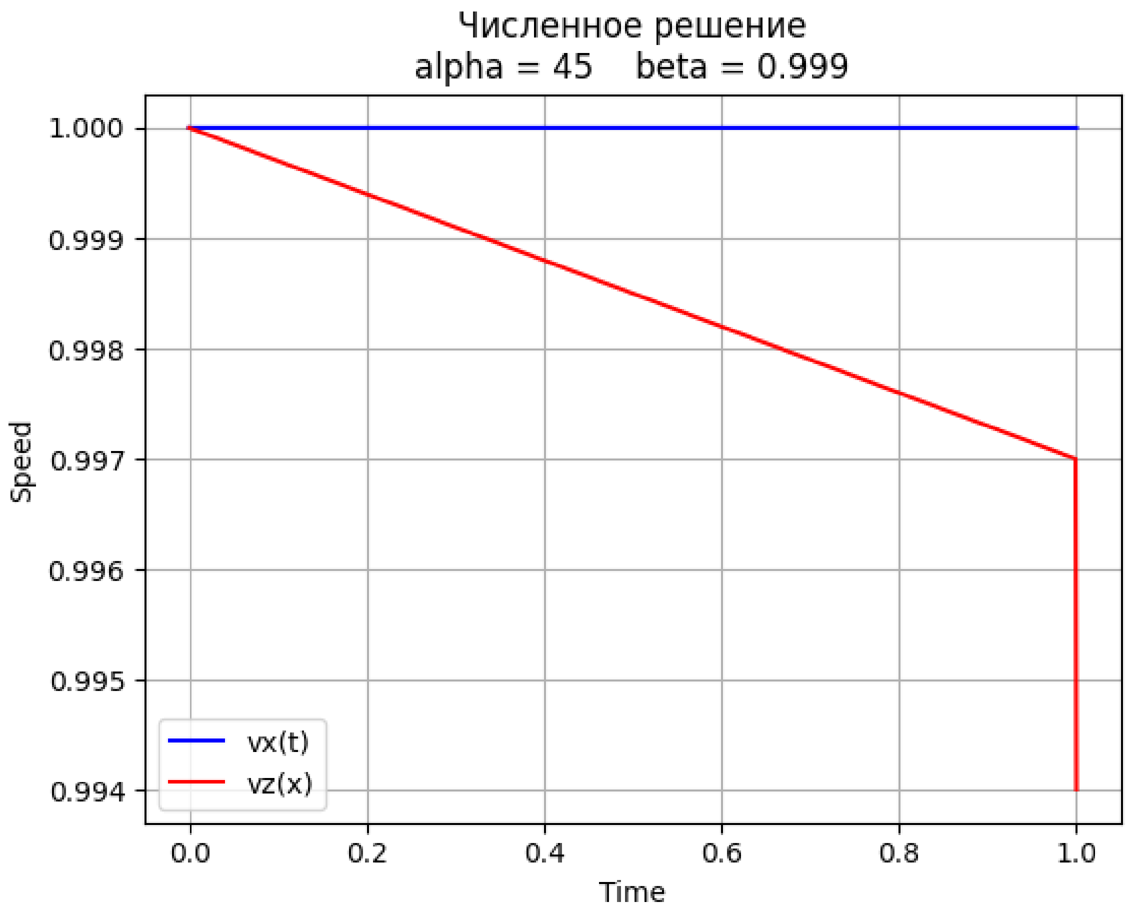
In [9]:

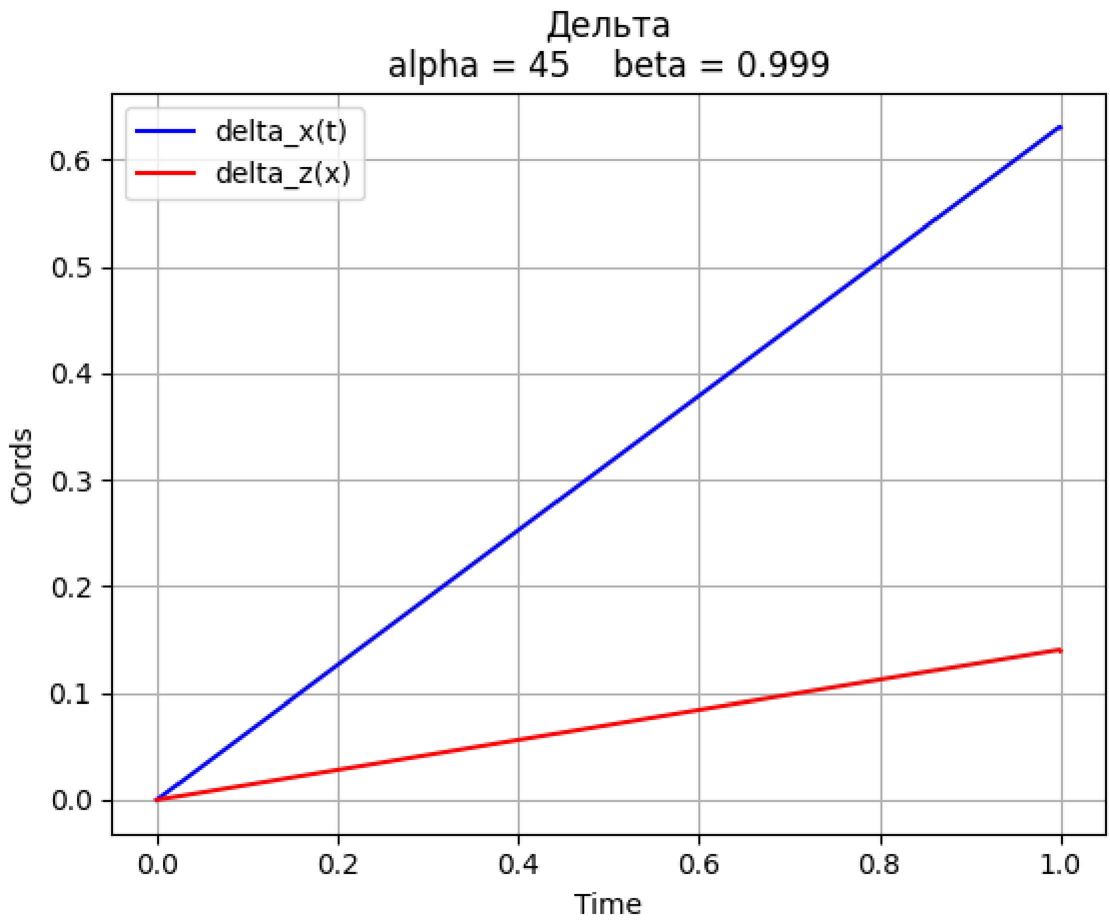
```
func(pi/4, 1-1e-3)
```

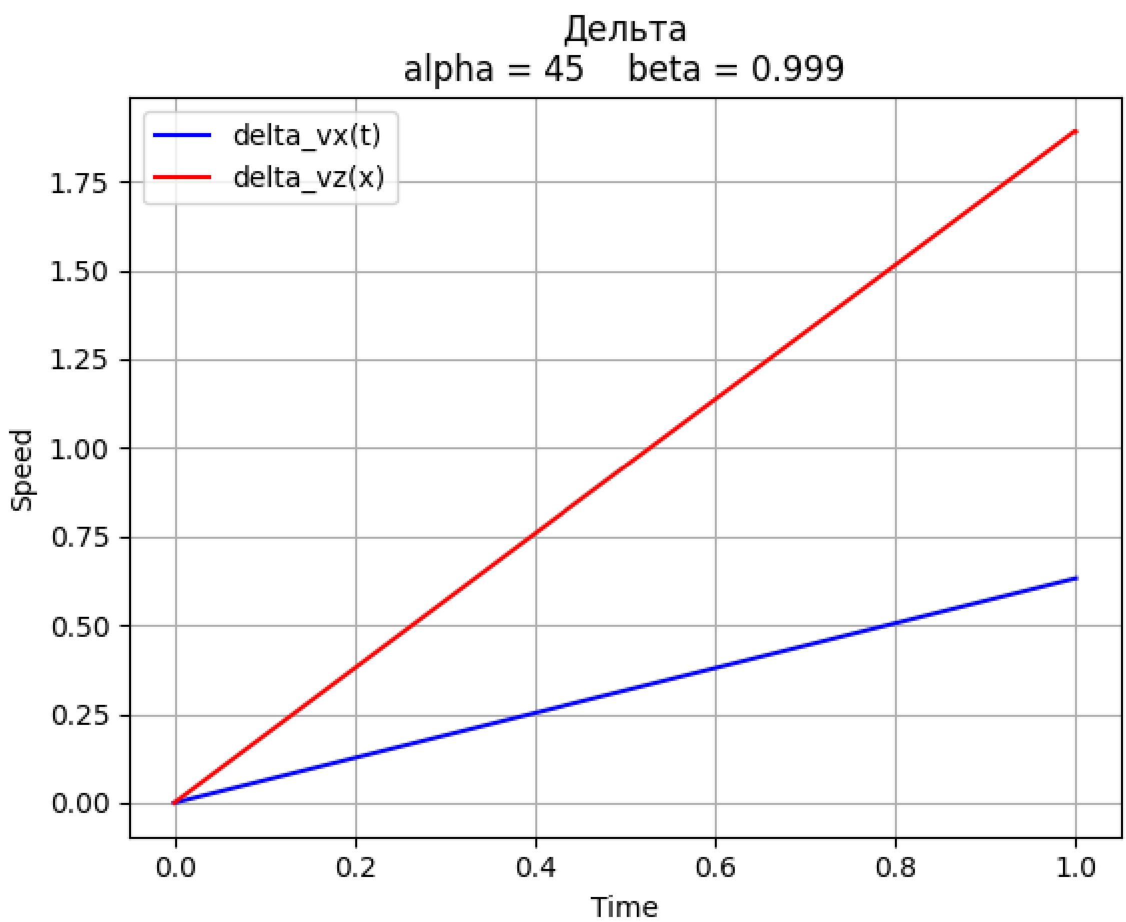





Численное решение
alpha = 45 beta = 0.999
x(t)
z(x)
Cords
Time
0.00200
0.00175
0.00150
0.00125
0.00100
0.00075
0.00050
0.00025
0.00000
0.0
0.2
0.4
0.6
0.8
1.0


Численное решение
alpha = 45 beta = 0.999
vx(t)
vz(x)
Speed
Time
1.000
0.999
0.998
0.997
0.996
0.995
0.994
0.0
0.2
0.4
0.6
0.8
1.0


Дельта
alpha = 45 beta = 0.999
delta_x(t)
delta_z(x)
Cords
Time
0.0
0.1
0.2
0.3
0.4
0.5
0.6
0.0 0.2 0.4 0.6 0.8 1.0


Дельта
alpha = 45 beta = 0.999
delta_vx(t)
delta_vz(x)
Speed
Time
0.00
0.25
0.50
0.75
1.00
1.25
1.50
1.75
0.0 0.2 0.4 0.6 0.8 1.0