ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ «НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО»

Факультет информационных технологий и программирования

Лабораторная работа №1

Разностная схема. Погрешность разностной схемы. Оценка погрешности.

Выполнила:
студентка группы M34021
Кумирова Екатерина Александровна

Принял:
кандидат физико-математических наук
Штумпф Святослав Алексеевич

Санкт-Петербург
2024 год

Система уравнений Лотки-Вольтерра «хищник-жертва»:

$$\begin{cases} \hat{x} = \alpha x - xy \\ \hat{y} = \beta xy - cy \end{cases}$$

$x(0) = x_0 > 0,\ y(0) = y_0 > 0$

x – безразмерная численность жертв, y – безразмерная численность хищников

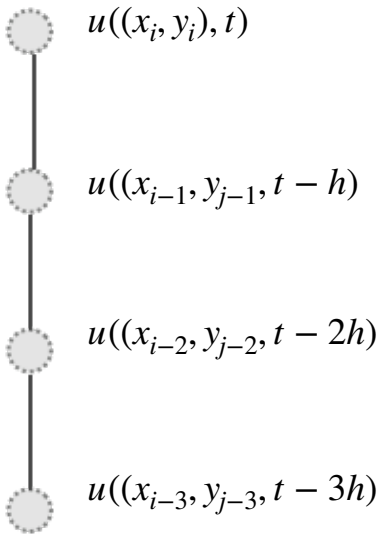
β, c – положительные константы ($\beta < 1$)

Скорость размножения жертв: $\alpha(t) = \alpha_0(1 + \sin(\omega t))$.

Явная численная схема, порядок точности 3:

Нам необходима точность $O(h^N) = O(h^3)$, следовательно, в шаблоне должно быть $N + 1 = 4$ точки.

Шаблон явной численной схемы для записи уравнения в координатах «время-численность»:



Разработаем явную численную схему 3-его порядка точности с помощью метода неопределённых коэффициентов. Существует единственный набор коэффициентов α, который позволяет найти на шаблоне из $(1 + l + m) = 4$ точек значение первой производной с точностью $O(h^{l+m=3})$. Численная схема - явная, так что положим $l = 3$ (количество точек назад), $m = 0$ (количество точек вперёд).

Мы ожидаем, что первая производная будет выражена через линейную комбинацию сеточных точек:

$$f'(x_j) \approx \frac{1}{h} \sum_{k=-l}^{m} \alpha_k f(x_j + kh)$$

Воспользуемся широко известным фактом: в окрестности точки функцию можно разложить в ряд Тейлора и получить некоторое приближение с помощью построенного ряда. Каждый $f(x_j + kh)$ мы раскладываем в ряд Тейлора в окрестности x_j и то, что получилось, суммируем, пока не достигнем 3-его порядка точности.

$$\frac{\partial x}{\partial t} \approx \frac{1}{h}\sum_{k=-3}^{0} \alpha_k f(x_j + kh)$$

$$\frac{\partial x}{\partial t} \approx \frac{1}{h}\Big(\alpha_{-3}x(t-3h) + \alpha_{-2}x(t-2h) + \alpha_{-1}x(t-h) + \alpha_0 x(t)\Big)$$

В упрощенное представление СЛАУ для матрицы неопределённых коэффициентов подставим наши значения $l = 3$ и $m = 0$:

$$\begin{pmatrix} 1 & 1 & \dots & 1 \\ -l & -l+1 & \dots & m \\ l^2 & (l-1)^2 & \dots & m^2 \\ (-l)^3 & (-l+1)^3 & \dots & m^3 \\ \dots & \dots & \dots & \dots \end{pmatrix} \begin{pmatrix} \alpha_{-3} \\ \alpha_{-2} \\ \alpha_{-1} \\ \alpha_0 \\ \dots \end{pmatrix} = (0,1,0,\dots,0)^T$$

$$\begin{pmatrix} 1 & 1 & 1 & 1 \\ -3 & -2 & -1 & 0 \\ 9 & 4 & 1 & 0 \\ -27 & -8 & -1 & 0 \end{pmatrix} \begin{pmatrix} \alpha_{-3} \\ \alpha_{-2} \\ \alpha_{-1} \\ \alpha_0 \end{pmatrix} = (0,1,0,0)^T$$

$$\left(\begin{array}{cccc|c} 1 & 1 & 1 & 1 & 0 \\ -3 & -2 & -1 & 0 & 1 \\ 9 & 4 & 1 & 0 & 0 \\ -27 & -8 & -1 & 0 & 0 \end{array}\right)$$

$$\left(\begin{array}{cccc|c} 1 & 0 & 0 & 0 & -\frac{1}{3} \\ 0 & 1 & 0 & 0 & 1{,}5 \\ 0 & 0 & 1 & 0 & -3 \\ 0 & 0 & 0 & 1 & \frac{11}{6} \end{array}\right)$$

$\alpha_{-3} = -\frac{1}{3}$, $\alpha_{-2} = 1{,}5$, $\alpha_{-1} = -3$, $\alpha_0 = \frac{11}{6}$

Обратим внимание на то, что в сумме коэффициенты дают 0 (нам необходимо, чтобы разностная схема не привносила никакого внешнего влияния, которого не заложено в исходной дифференциальной задаче).

$$\frac{\partial x}{\partial t} \approx \frac{1}{h}\Big(-\frac{1}{3}x(t-3h) + 1{,}5x(t-2h) - 3x(t-h) + \frac{11}{6}x(t)\Big)$$

$$\frac{1}{h}\Big(-\frac{1}{3}x(t-3h) + 1{,}5x(t-2h) - 3x(t-h) + \frac{11}{6}x(t)\Big) = \alpha(t-h)x(t-h) - x(t-h)y(t-h)$$

$$x(t) = \frac{6}{11}\Big(h\big(\alpha(t-h)x(t-h) - x(t-h)y(t-h)\big) + \frac{1}{3}x(t-3h) - 1{,}5x(t-2h) + 3x(t-h)\Big)$$

По аналогии для y:

$$\frac{1}{h}\Big(-\frac{1}{3}y(t-3h) + 1{,}5y(t-2h) - 3y(t-h) + \frac{11}{6}y(t)\Big) = \beta x(t-h)y(t-h) - c\,y(t-h)$$

$$y(t) = \frac{6}{11}\Big(h\big(\beta x(t-h)y(t-h) - c\,y(t-h)\big) + \frac{1}{3}x(t-3h) - 1{,}5x(t-2h) + 3x(t-h)\Big)$$

Итоговая численная схема:

$$\begin{cases} x(t) = \frac{6}{11}\Big(h\big(\alpha(t-h)x_{n-1} - x_{n-1}y_{n-1}\big) + \frac{1}{3}x_{n-3} - 1{,}5x_{n-2} + 3x_{n-1}\Big) \\ y(t) = \frac{6}{11}\Big(h\big(\beta x(t-h)y_{n-1} - c\,y_{n-1}\big) + \frac{1}{3}x_{n-3} - 1{,}5x_{n-2} + 3x_{n-1}\Big) \end{cases}$$

Оценка погрешности:

Численная схема 3-го порядка точности, следовательно, погрешность отбрасывания зависит от h^3 (мы отбросили 4-ый компонент разложения, взяли его модуль и тем самым обеспечили точность 3-его порядка).

$$\triangle_{\text{отбрасывания}} = |x_{IV}| = h^3\Big(-\frac{1}{3}\cdot\frac{3^4}{4!} + \frac{3}{2}\cdot\frac{2^4}{4!} - 3\cdot\frac{1^4}{4!}\Big) = h^3\Big(-\frac{1}{3}\cdot\frac{81}{24} + \frac{3}{2}\cdot\frac{16}{24} - 3\cdot\frac{1}{24}\Big) = -h^3\frac{1}{4}$$

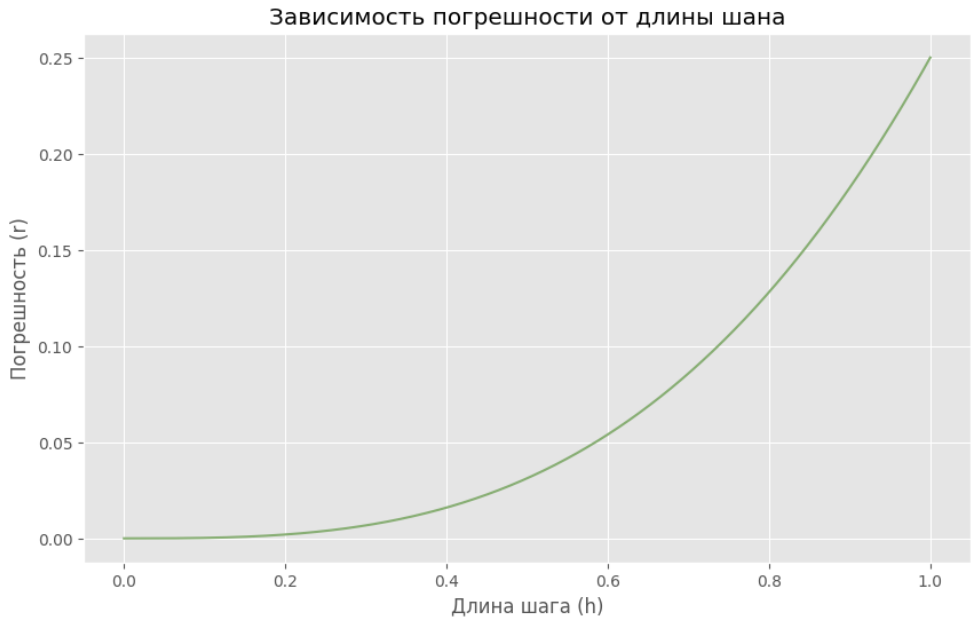
$$\triangle_{\text{округления}} = \frac{2\varepsilon}{h},\ \varepsilon - \text{машинный эпсилон.}$$

$$\triangle = \triangle_{\text{отбрасывания}} + \triangle_{\text{округления}} = h^3\frac{1}{4} + \frac{2\varepsilon}{h}.$$

In [ ]:
```
import matplotlib as mpl
from matplotlib import pyplot as plt
mpl.style.use(['ggplot'])
```

In [ ]:
```
import sys
def get_epsilon(h):
    return h**3/4 + 2 * sys.float_info.epsilon/h
```

In [ ]:
```
import numpy as np
import matplotlib.pyplot as plt

h_values = np.linspace(0.0001, 1.0, 1000)
eps_values = get_epsilon(h_values)

plt.figure(figsize=(10, 6))
plt.plot(tau_values, eps_values, label='Погрешность', color='xkcd:sage')
plt.title('Зависимость погрешности от длины шана')
plt.xlabel('Длина шага (h)')
plt.ylabel('Погрешность (r)')
plt.show()
```



In [ ]:
```
from prettytable import PrettyTable
table = PrettyTable()
table.title = "Зависимость погрешности от длины шага"
table.field_names = ["Длина шага (h)", "Погрешность (r)"]

h_values = [0.0001, 0.0005, 0.001, 0.005, 0.01, 0.05, 0.1, 0.5, 1]

for h in h_values:
    r = get_eps(h)
    table.add_row([h, r])

print(table)
```

```
+-----------------------------------------+
|  Зависимость погрешности от длины шага  |
+----------------+------------------------+
| Длина шага (h) |    Погрешность (r)     |
+----------------+------------------------+
|     0.0001     | 4.690892098500626e-12  |
|     0.0005     | 3.213817841970013e-11  |
|     0.001      | 2.504440892098501e-10  |
|     0.005      | 3.1250088817841975e-08 |
|      0.01      | 2.5000004440892103e-07 |
|      0.05      | 3.125000000888179e-05  |
|      0.1       | 0.00025000000000444095 |
|      0.5       |  0.03125000000000089   |
|       1        |  0.25000000000000044   |
+----------------+------------------------+
```

In [ ]:
```
import math
import numpy as np
import matplotlib as mpl
from matplotlib import pyplot as plt
mpl.style.use(['ggplot'])
```

Моделирование

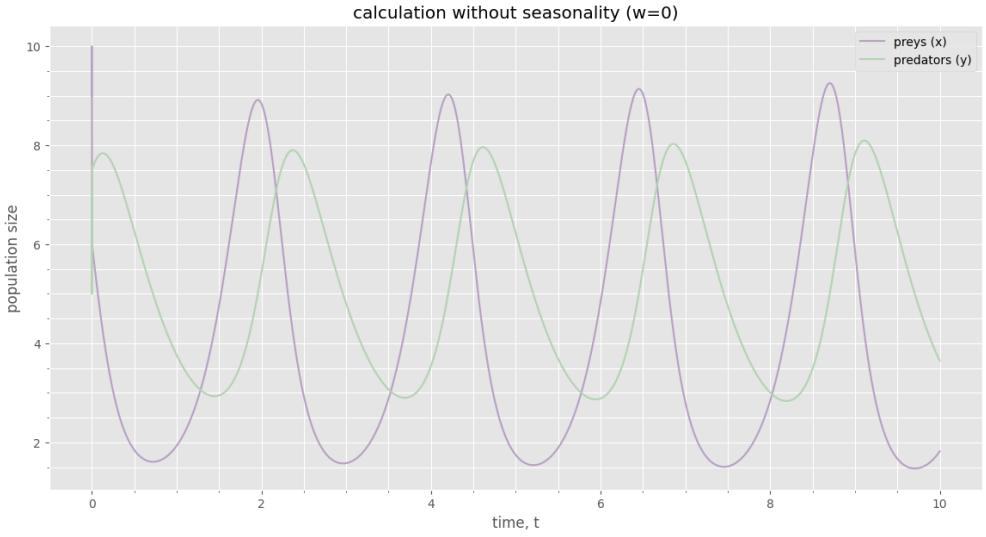
In [ ]:
```
def simulate_population(x0, y0, w, a_0=5, b=0.4, c=1.7, h=0.001, iterations=10000):
    def a(t):
        return a_0 * (1 + math.sin(w * t))

    x = x0.copy()
    y = y0.copy()

    time_points = [0, h, 2 * h]

    delta = -1/3    # a_{-3}
    gamma = 1.5     # a_{-2}
    beta = -3       # a_{-1}
    alpha = 11/6    # a_0

    for i in range(3, iterations):
        current_time = time_points[-1] + h
        time_points.append(current_time)

        x_value = (h * (a(current_time - h) * x[i-1] - x[i-1] * y[i-1]) - delta * x[i-3] - gamma * x[i-2] - beta * x[i-1]) / alpha
        y_value = (h * (b * x[i-1] * y[i-1] - c * y[i-1]) - delta * y[i-3] - gamma * y[i-2] - beta * y[i-1]) / alpha

        x.append(x_value)
        y.append(y_value)

    return time_points, x, y
```

Начальные условия

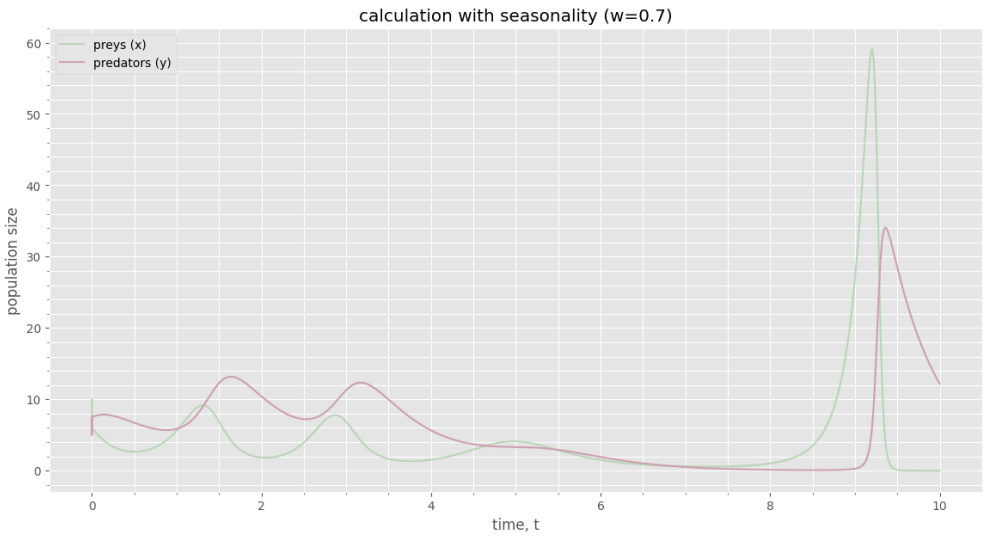
In [ ]:
```
x0 = [9, 10, 8] # preys
y0 = [7, 5, 6] # predators
```

Визуализация

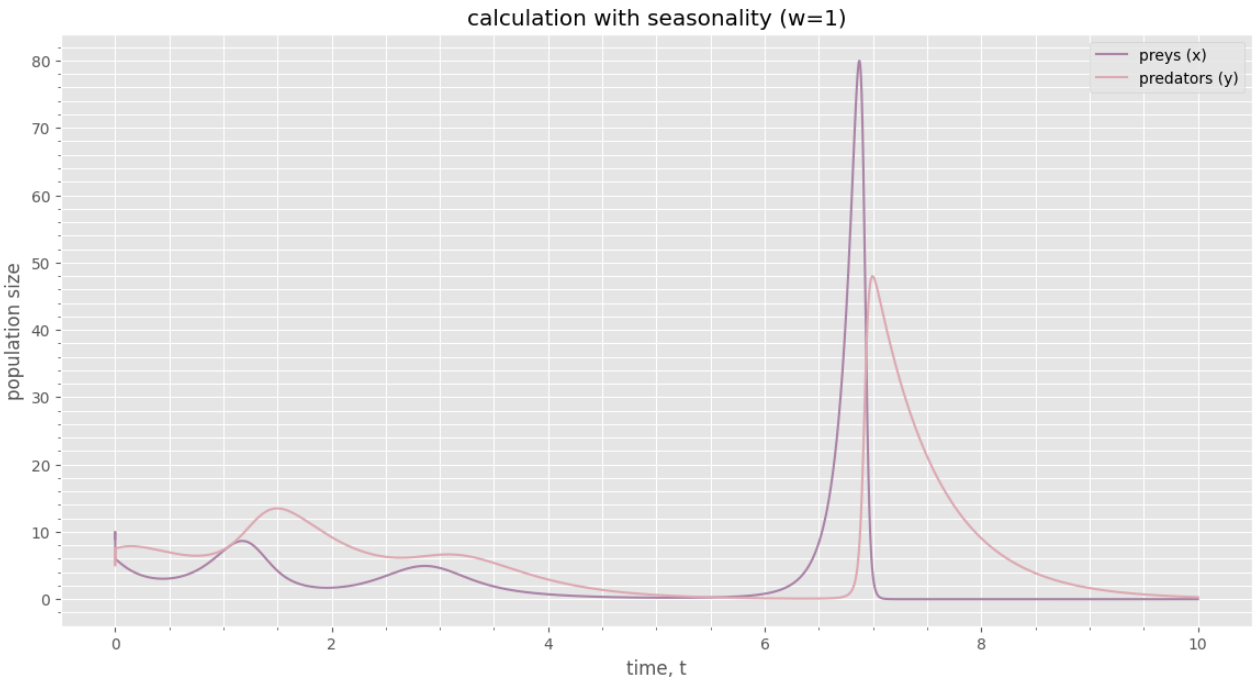
In [ ]:
```
w = 0
h_arr, x, y = simulate_population(x0, y0, w)

fig = plt.figure(figsize=(14, 7))
plt.grid(True, which='both')
plt.minorticks_on()
plt.plot(h_arr, x, c="#B7A0C5", label="preys (x)")
plt.plot(h_arr, y, c='#b4d3b2', label="predators (y)")

plt.legend()
plt.xlabel("time, t")
plt.ylabel("population size")
plt.title("calculation without seasonality (w=0)")
plt.show()
```



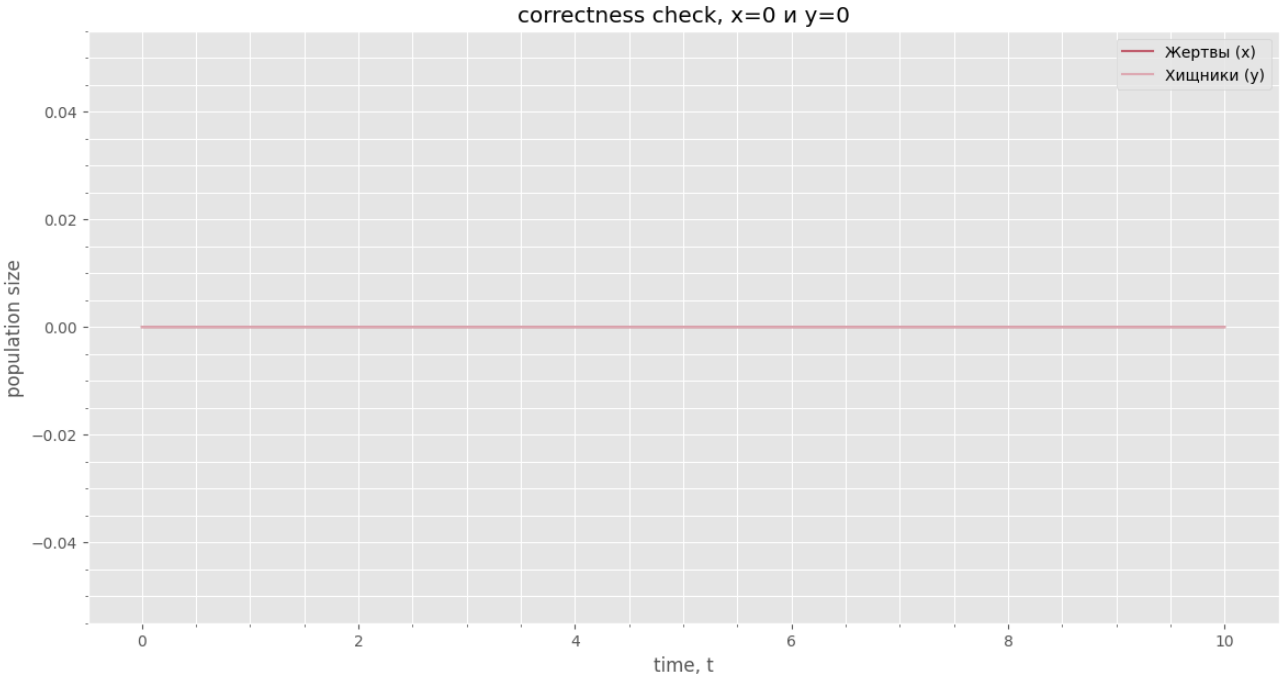
In [ ]:
```
w = 0.7
h_arr, x, y = simulate_population(x0, y0, w)

fig = plt.figure(figsize=(14, 7))
plt.grid(True, which='both')
plt.minorticks_on()
plt.plot(h_arr, x, color='#bad6b8', label="preys (x)")
plt.plot(h_arr, y, color='#cea0af', label="predators (y)")

plt.legend()
plt.xlabel("time, t")
plt.ylabel("population size")
plt.title("calculation with seasonality (w=0.7)")
plt.show()
```



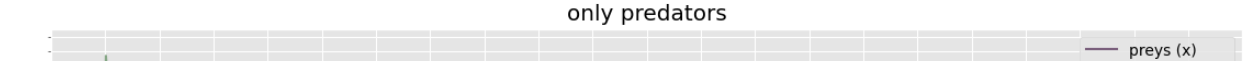
In [ ]:
```
w = 1
h_arr, x, y = simulate_population(x0, y0, w)

fig = plt.figure(figsize=(14, 7))
plt.grid(True, which='both')
plt.minorticks_on()
plt.plot(h_arr, x, color='#aa83a6', label="preys (x)")
plt.plot(h_arr, y, color='#dda9b2', label="predators (y)")

plt.legend()
plt.xlabel("time, t")
plt.ylabel("population size")
plt.title("calculation with seasonality (w=1)")
plt.show()
```

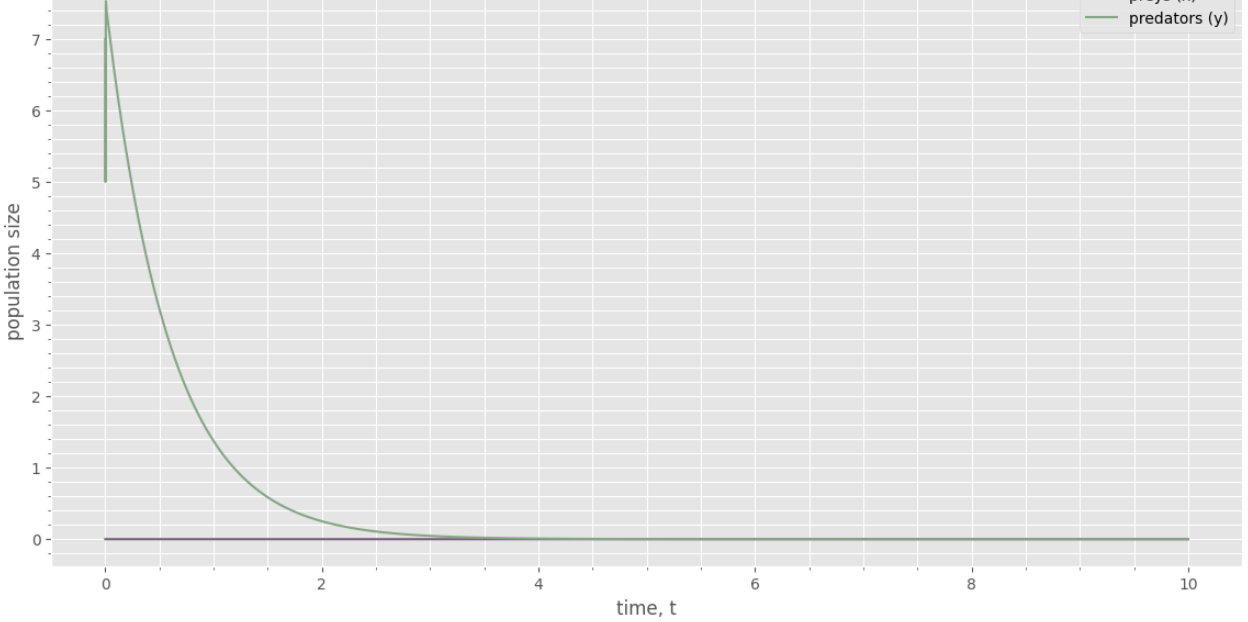


In [ ]:
```
h_arr, x, y = simulate_population([0]*3, [0]*3, w=0)
fig = plt.figure(figsize=(14, 7))
plt.grid(True, which='both')
plt.minorticks_on()
plt.plot(h_arr,x, color='#c05d6d', label="Жертвы (x)")
plt.plot(h_arr,y, color='#dda9b2', label="Хищники (y)")

plt.legend()
plt.xlabel("time, t")
plt.ylabel("population size")
plt.title("correctness check, x=0 и y=0")
plt.show()
```



In [ ]:
```
h_arr, x, y = simulate_population([0,0,0], y0, w=0)
fig = plt.figure(figsize=(14, 7))
plt.grid(True, which='both')
plt.minorticks_on()
plt.plot(h_arr,x, color='#7a5f7c', label="preys (x)")
plt.plot(h_arr,y, color='#86a786', label="predators (y)")

plt.legend()
plt.xlabel("time, t")
plt.ylabel("population size")
plt.title("only predators")
plt.show()
```





In [ ]:
```
h_arr, x, y = simulate_population(x0, [0,0,0], w=0)
fig = plt.figure(figsize=(14, 7))
plt.grid(True, which='both')
plt.minorticks_on()
plt.plot(h_arr,x, color='#aa83a6', label="preys(x)")
plt.plot(h_arr,y,  color='#86a786', label="predators(y)")

plt.legend()
plt.xlabel("time, t")
plt.ylabel("population size")
plt.title("only preys")
plt.show()
```