

КАЗАНСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ  
ИНСТИТУТ ЭКОЛОГИИ И ПРИРОДОПОЛЬЗОВАНИЯ  
*Кафедра моделирования экологических систем*

МАТЕМАТИЧЕСКИЕ МОДЕЛИ ДИНАМИКИ  
ПОПУЛЯЦИЙ:  
РЕАЛИЗАЦИЯ НА ЯЗЫКЕ PYTHON

Учебное пособие

Казань – 2023

*Принято на заседании учебно-методической комиссии  
Института экологии и природопользования  
Протокол № 2 от 23 марта 2023 г.*

**Авторы-составители:**

доктор физико-математических наук, профессор **Зарипов Ш.Х.**,  
кандидат физико-математических наук, доцент **Никоненкова Т.В.**,  
ассистент **Ложкин Г.И.**,  
кандидат физико-математических наук, доцент **Гильфанов А.К.**,  
кандидат физико-математических наук, доцент **Костерина Е.А.**

**Рецензент:**

доктор физико-математических наук, профессор **Обносков Ю.В.**

**Математические модели динамики популяций: реализация на языке Python:** учебное пособие / Ш.Х. Зарипов, Т.В. Никоненкова, Г.И. Ложкин, А.К. Гильфанов, Е.А. Костерина. – Казань: Изд-во Казанского федерального университета, 2023. – 44 с.

Учебное пособие предназначено для студентов бакалавриата и магистратуры, обучающихся по направлениям "Экология и природопользование", "Биотехнология". Может представлять интерес для обучающихся смежных специальностей.

В пособии приведены математические модели динамики популяций и описание их реализации на языке Python.

# ОГЛАВЛЕНИЕ

<b>ВВЕДЕНИЕ</b>	<b>4</b>
<b>1. ОБЫКНОВЕННЫЕ ДИФФЕРЕНЦИАЛЬНЫЕ УРАВНЕНИЯ</b>	<b>5</b>
1.1. Сведения из теории обыкновенных дифференциальных уравнений . . . . .	5
1.2. Примеры решения обыкновенных дифференциальных уравнений на языке Python . . . . .	7
<b>2. ДИФФЕРЕНЦИАЛЬНЫЕ УРАВНЕНИЯ В ТЕОРИИ ЭПИДЕМИИ (МОДЕЛЬ БЕЙЛИ)</b>	<b>14</b>
2.1. Модель эпидемии без учета выздоровления больных особей . .	14
2.2. Модель эпидемии с учетом выздоровления больных особей . . .	16
<b>3. ДИНАМИКА ЧИСЛЕННОСТИ ПОПУЛЯЦИИ</b>	<b>19</b>
3.1. Модель Мальтуса (неограниченный рост) . . . . .	19
3.2. Логистическая модель (ограниченный рост) . . . . .	20
<b>4. МОДЕЛИ ВЗАИМОДЕЙСТВИЯ ПОПУЛЯЦИЙ: "ХИЩНИК-ЖЕРТВА"</b>	<b>23</b>
<b>5. МОДЕЛЬ ДИНАМИКИ БИОМАССЫ МИКРООРГАНИЗМОВ С УЧЕТОМ ВЛИЯНИЯ ОСВЕЩЕННОСТИ</b>	<b>27</b>
<b>6. ДИСКРЕТНЫЕ МОДЕЛИ ПОПУЛЯЦИЙ</b>	<b>31</b>
6.1. Дискретная модель с неограниченным ростом численности популяции . . . . .	32
6.2. Дискретная логистическая модель (ограниченный рост численности популяции) . . . . .	35
6.3. Модель Рикера (ограниченный рост численности популяции) . .	37
6.4. Дискретная модель популяции с учетом возрастной структуры .	39
<b>ЛИТЕРАТУРА</b>	<b>44</b>

## ВВЕДЕНИЕ

Целью настоящего учебного пособия является освоение решения типичных задач математической экологии с помощью языка программирования Python.

Для описания экологических проблем привлекают методы из самых разных областей математического знания. Самое широкое распространение получили подходы, основанные на аппарате дифференциальных и разностных уравнений. Дифференциальные и разностные уравнения позволяют описывать динамику численности (биомассы) популяции, динамику растительного покрова, обмен веществ, обмен энергией, информацией или любые другие величины, которые изменяются во времени и/или пространстве.

Python — высокоуровневый язык программирования общего назначения, ориентированный на повышение производительности разработчика и читаемости кода. Python реализован практически во всех операционных системах, и большинство его модулей распространяется бесплатно. Язык программирования Python обладает ясным и понятным синтаксисом и хорош для программирования математических вычислений. Стандартные библиотеки включают большой объем полезных функций. Кроме того, для Python написано большое количество прикладных библиотек, в том числе для научных расчетов, которые позволяют решать ряд математических задач без необходимости самостоятельной разработки алгоритмов.

Программную среду для Python можно установить с сайта разработчиков <https://www.python.org/downloads/>. Необходимо выбрать желаемую версию Python и скачать инсталляционный пакет, соответствующий вашей операционной системе. После запуска скачанного файла и установки Python, вам потребуется также установить дополнительные библиотеки.

Другой способ установить Python состоит в использовании бесплатного дистрибутива Anaconda (<https://www.continuum.io/downloads>). Это самый простой способ установить сразу Python и стандартные библиотеки. Кроме всего прочего, вы получите интегрированные оболочки Jupyter Notebook и Spyder, предназначенные для разработки и выполнения программ.

С основными принципами работы в среде Python и командами можно ознакомиться, например, в пособиях [1], [2] и [3].

# 1. ОБЫКНОВЕННЫЕ ДИФФЕРЕНЦИАЛЬНЫЕ УРАВНЕНИЯ

В данном разделе рассматривается численное решение обыкновенных дифференциальных уравнений (ОДУ) с помощью Python-библиотеки **scipy**, которая предназначена для выполнения сложных инженерных, статистических и научных расчетов, а также для построения графиков. Это библиотека с открытым исходным кодом. Она построена на базе библиотеки NumPy. Scipy является частью дистрибутива Anaconda и может быть установлен вместе с Anaconda.

## 1.1. Сведения из теории обыкновенных дифференциальных уравнений

*Обыкновенное дифференциальное уравнение* (ОДУ) — уравнение, содержащее производную искомой функции одной переменной.

### Дифференциальные уравнения первого порядка

Рассмотрим уравнение первого порядка, разрешенное относительно производной, которое имеет вид:

$$y' = f(t, y, p), \quad (1)$$

где  $t$  — независимая переменная,  $y$  — искомая функция,  $p$  — числовой параметр.

Дифференциальное уравнение первого порядка (1) имеет бесконечно много решений. Для того чтобы выделить единственное решение, нужно задать *дополнительное (начальное) условие*:

$$y(t_0) = y_0 \quad (y = y_0 \text{ при } t = t_0). \quad (2)$$

Задача отыскания решения  $y = y(t)$  уравнения (1), удовлетворяющего условию (2), называется *задачей Коши (или начальной задачей)*.

### Дифференциальные уравнения высших порядков

Помимо дифференциальных уравнений первого порядка в различных прикладных задачах большую роль играют дифференциальные уравнения, содержащие производные высших порядков. Например, уравнение второго порядка

$$y'' = f(t, y, y'),$$

возникает во многих практических ситуациях.

Как известно из теории обыкновенных дифференциальных уравнений, данное уравнение второго порядка мы можем представить в виде *системы двух уравнений первого порядка*. Для этого достаточно ввести новую переменную  $y_1 = y'$ :

$$\begin{cases} y' = y_1, \\ y_1' = f(t, y, y_1). \end{cases}$$

Вводя вектор-функцию  $Y = (y, y_1)$ , видно, что последняя система уравнений имеет вид (1).

Аналогичным образом поступают и в случае, когда дифференциальные уравнения, разрешенные относительно старшей производной, содержат производные третьего, четвертого и более высоких порядков.

### Методы решения дифференциальных уравнений

В классическом анализе разработано немало приемов нахождения решений дифференциальных уравнений через элементарные функции. Между тем при решении практических задач эти методы оказываются, как правило, либо совсем бесполезными, либо их решение связано с недопустимыми затратами усилий и времени. Например, уравнение

$$y' = y^2 + 1 \tag{3}$$

имеет аналитическое решение

$$y = \operatorname{tg}(t + c),$$

где  $c$  – произвольная константа. Данное решение легко получается интегрированием обеих частей уравнения (3). Но если уравнение (3) "слегка" подправить к виду:

$$y' = y^2 + t,$$

то решение последнего уравнения уже имеет очень сложную структуру.

К сожалению, дифференциальные уравнения для многих практических задач не могут быть решены аналитически, а чаще всего численные методы – единственная возможность получить решение.

В целом, методы решения дифференциальных уравнений условно можно разделить на две основные группы:

- 1) аналитические методы, когда решение дифференциального уравнения получается в виде функции, заданной формулой;
- 2) численные методы, когда решение дифференциального уравнения по-

лучается в виде функции, заданной в табличной форме.

Так как цель данного раздела состоит в том, чтобы дать краткий обзор по решению обыкновенных дифференциальных уравнений на языке Python, нас будут интересовать лишь численные методы, реализация которых может быть осуществлена с помощью инструментов Python-библиотеки `scipy`.

## 1.2. Примеры решения обыкновенных дифференциальных уравнений на языке Python

Прежде чем приступить к практической части, следует сказать, что для решения обыкновенного дифференциального уравнения (ОДУ) в Python чаще всего используется функция `odeint` модуля `scipy.integrate` библиотеки `scipy`. Эта функция предназначена для решения ОДУ с начальным условием (задача Коши).

### Описание функции `odeint()`

Функция `odeint()` имеет три обязательных аргумента и много опций.

```
odeint(func, y0, t, ...)
```

**func** – функция двух переменных, описывающая дифференциальное уравнение (1)

**y0** – начальное значение искомой функции при  $t=t_0$

**t** – массив точек, в которых необходимо найти значение искомой функции  $y(t)$ . Первый элемент массива должен совпадать с начальной точкой  $t_0$

**...** – необязательные аргументы, о которых можно посмотреть в справке `help(odeint)`

Для того, чтобы использовать функцию `odeint`, дифференциальное уравнение должно быть приведено к стандартному виду (разрешенному относительно производной):

$$\frac{dy}{dt} = f(t, y),$$

где  $y$  может быть вектором, что позволяет решать системы дифференциальных уравнений и дифференциальные уравнения высших порядков.

## Пример 1. Решение обыкновенного дифференциального уравнения первого порядка

Требуется найти решение обыкновенного дифференциального уравнения

$$y' = y^2 + t \quad (4)$$

с начальным условием  $y(0) = 0.1$  на интервале  $t \in [0, 1]$ .

*Решение.*

Listing 1: Пример 1

```
from scipy.integrate import odeint # 1 строка
import numpy as np                 # 2 строка
import matplotlib.pyplot as plt   # 3 строка
import os                           # 4 строка

os.chdir("C:\Work")                # 5 строка, путь к папке Work

def func1(y,t):                    # 6 строка
    return y**2+t                  # 7 строка

y0=0.1                             # 8 строка
ti=np.arange(0,1.1,0.1)            # 9 строка
yi=odeint(func1,y0,ti)             # 10 строка
print(yi)                          # 11 строка

# График функции y(t)
plt.plot(ti,yi,"o-r",alpha=0.7,lw=5,mec="g",mew=2,ms=10) # 12 стр
plt.xlabel("t, время",fontsize=20)# 13 строка
plt.ylabel("y",fontsize=20)        # 14 строка
plt.tick_params(axis="both",labelsize=15) # 15 строка
plt.grid(True)                     # 16 строка, добавление сетки
plt.savefig("Fig1.png")            # 17 строка, сохранение png-файла
plt.show()                         # 18 строка
```

Для того, чтобы получить численное решение заданного уравнения (4) в Python, необходимо загрузить модуль **scipy.integrate**, который является частью библиотеки **scipy** (строка 1, Listing 1). Помимо этого нам понадобится библиотека, в которой мы можем выполнять простейшие операции, связанные с матрицами или списками. Для этого загружаем библиотеку **numpy** под псевдонимом **np** (строка 2). А для графической реализации проекта нам потребуется библиотека **matplotlib** и модуль **pyplot** (строка 3). Модуль **matplotlib.pyplot** предоставляет процедурный интерфейс к (объектно-ориентированной) библиотеке **matplotlib**.



Далее определяем функцию, с которой нам предстоит работать. В Python это реализуется через команду **def** (строки 6-7). Задаем начальное значение переменной  $y$  (строка 8). Затем генерируем значения независимой переменной  $t_i$  как последовательность точек из интервала от 0 до 1 с шагом 0.1 (строка 9). Стоит отметить, что функция  $np.arange(start, stop, step)$  возвращает значения в пределах полуоткрытого интервала  $[start, stop)$ .

Решение простого дифференциального уравнения (4) в Python можно найти с помощью функции **odeint** (строка 10), аргументами которой являются: заданная функция `func1`; начальное значение функции  $y$  при  $t = t_0$ ; массив точек  $t_i$ , в которых ищется значение искомой функции  $y(t)$ .

Функция `odeint()` возвращает производные как список (строка 11) при заданных значениях  $t_i = \{0, 0.1, 0.2, \dots, 0.9, 1\}$  независимого аргумента  $t$ :

```

                yi
            [0.10000000]
            [0.10604437]
            [0.12233082]
            [0.14915528]
            [0.18696006]
            [0.23641918]
            [0.29854685]
            [0.37484716]
            [0.46753427]
            [0.57987677]
            [0.71676614]
    
```

Далее визуализируем полученное решение (строки 12-18). Зависимость  $y$  от  $t$  (рис.1) можно легко получить при помощи команды `plt.plot` (строка 12).

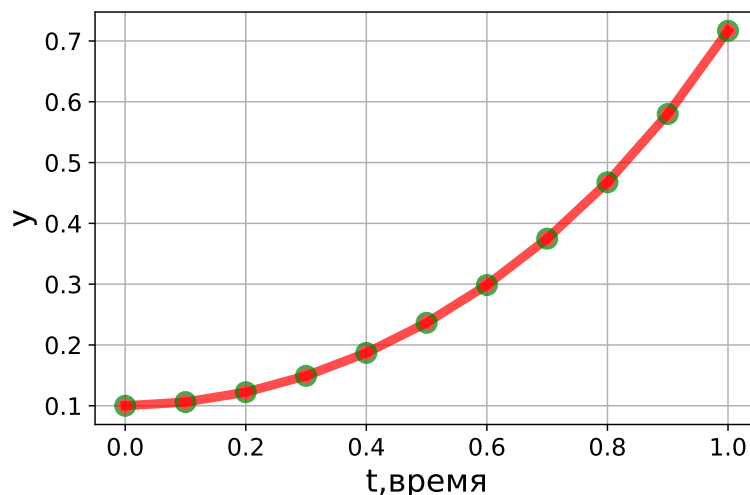


Рис. 1: Зависимость  $y$  от  $t$

Информацию по параметрам функции `plt.plot` можно найти в документации <https://matplotlib.org/stable/api>.

### Задания для самостоятельного решения

Решить следующие задачи Коши:

1.  $y' = y^2 - yt, y(0) = 0, t \in [0, 1]$ ;
2.  $y' = y^2 + 1, y(0) = 0, t \in [0, 1]$ .

### Пример 2. Решение системы ОДУ

Рассмотрим модель, которая была предложена в качестве модели турбулентности в 1963 году американским метеорологом Э.Лоренцем. Модель Лоренца имеет вид системы трех дифференциальных уравнений:

$$\begin{cases} \frac{dx}{dt} = ax + yz, \\ \frac{dy}{dt} = b(y - z), \\ \frac{dz}{dt} = -xy + cy - z, \end{cases} \quad (5)$$

где присутствуют три неизвестных функции  $x(t)$ ,  $y(t)$  и  $z(t)$ , а также параметры  $a, b, c$ . В качестве значений параметров возьмем следующие величины:  $-8/3, -10$  и  $28$ . Начальные условия:  $x(0) = y(0) = z(0) = 1$ .

*Решение.*

Систему дифференциальных уравнений (5) решаем аналогично тому, как решали обыкновенное дифференциальное уравнение (1) (Listing 2). Результат численного решения изображен на рис.2.

Listing 2: Пример 2

```
from scipy.integrate import odeint
import numpy as np
import matplotlib.pyplot as plt
import os

os.chdir("C:\Work")

def func2(eq,t):
    x,y,z=eq
    return [a*x+y*z,b*(y-z),-x*y+c*y-z]

a=-8/3; b=-10; c=28;
```

```

ti=np.arange(0,100,0.01)
y0=[1,1,1]
sol=odeint(func2,y0,ti)
x=sol[:,0]; y=sol[:,1]; z=sol[:,2];

plt.subplot(221) # две строки, два столбца, 1-й рисунок
plt.plot(x,y,lw=1)
plt.title("Фазовый портрет")
plt.xlabel("x")
plt.ylabel("y")
plt.grid(); plt.tight_layout();

plt.subplot(222) # две строки, два столбца, 2-й рисунок
plt.plot(ti,x,lw=1,color="orange")
plt.title("x(t)")
plt.xlabel("t")
plt.ylabel("x")
plt.grid(); plt.tight_layout();

plt.subplot(223) # две строки, два столбца, 3-й рисунок
plt.plot(ti,y,lw=1,color="g")
plt.title("y(t)")
plt.xlabel("t")
plt.ylabel("y")
plt.grid(); plt.tight_layout();

plt.subplot(224) # две строки, два столбца, 4-й рисунок
plt.plot(ti,z,lw=1,color="r")
plt.title("z(t)")
plt.xlabel("t")
plt.ylabel("z")
plt.grid(); plt.tight_layout();

plt.savefig("Fig2.png")
plt.show()

```

---

## Задания для самостоятельного решения

Решить систему обыкновенных дифференциальных уравнений

$$\begin{cases} y_1' = -y_2 - y_3, \\ y_2' = y_1 + ay_2, \\ y_3' = b + y_3(y_1 - c), \end{cases}$$

с начальными значениями  $(1,1,1)$  при  $t \in [0, 100]$ . Для параметров взять следующие значения:  $a = 0.2$ ,  $b = 0.2$ ,  $c = 5$ .

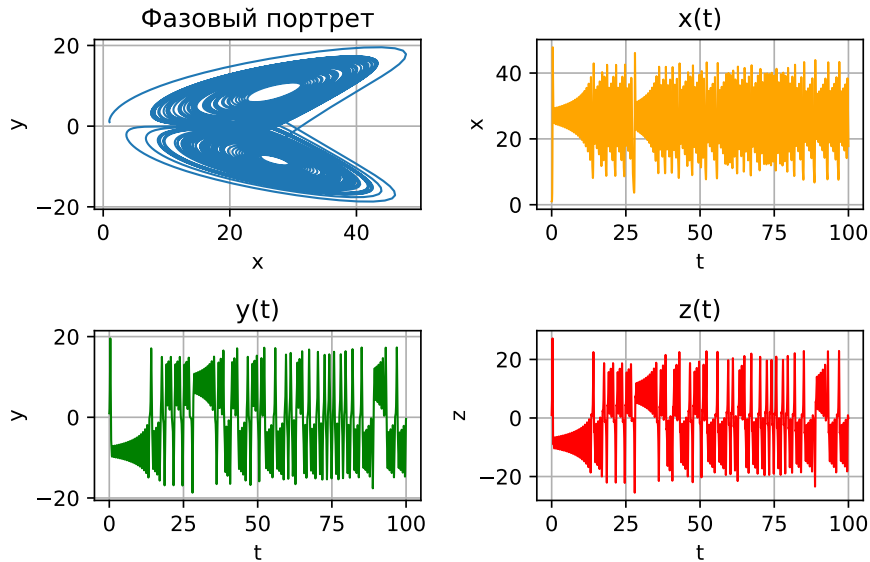


Рис. 2: Зависимость  $x, y, z$  от  $t$  и фазовый портрет в переменных  $x, y$

### Пример 3. Решение обыкновенного дифференциального уравнения второго порядка

Решим уравнение второго порядка для  $t \in [0, 20]$

$$y'' = -0.1y \quad (6)$$

с начальным условием  $y(0) = 1, y'(0) = 0$ .

*Решение.*

Чтобы решить данную задачу, необходимо сначала преобразовать уравнение (6) к системе дифференциальных уравнений первого порядка с помощью замены  $y' = y_1$ :

$$\begin{cases} y' = y_1, \\ y_1' = -0.1y \end{cases} \quad (7)$$

с начальными условиями  $y(0) = 1$  и  $y_1(0) = 0$ .

Реализация данной задачи в Python отображена в Listing 3, а результат решения – на рис. 3.

#### Listing 3: Пример 3

```
from scipy.integrate import odeint
import numpy as np
import matplotlib.pyplot as plt
import os

os.chdir("C:\Work")
```

```

def func3(y,t):
    p,p1=y
    dydt=[p1, -0.1*p]
    return dydt

ti=np.arange(0,20,0.1)
y0=[1,0]
sol=odeint(func3,y0,ti)

plt.plot(ti,sol[:,0],"r",label="y")
plt.plot(ti,sol[:,1],"g",label="y'")
plt.xlabel("t",fontsize=17)
plt.ylabel("y",fontsize=17)
plt.grid();plt.legend();
plt.savefig("Fig3.png")
plt.show()

```

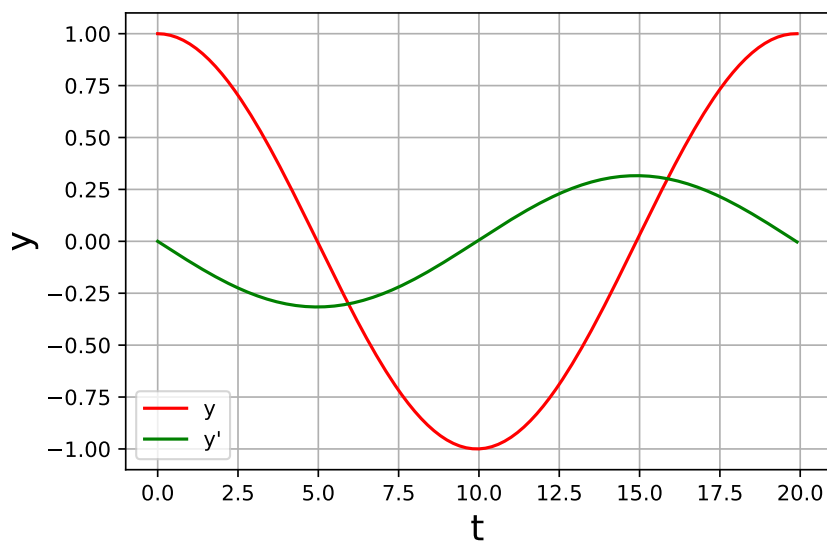


Рис. 3: Зависимость  $y$  от  $t$

### Задания для самостоятельного решения

Решить следующие задачи:

1.  $y'' + 2y' + 3y = \cos(t)$ ,  $y(0) = y'(0) = 0$ ,  $t \in [0, 2\pi]$ ;
2.  $z'' - a(1 - z^2)z' + z = 0$ ,  $z(0) = 2$ ,  $z'(0) = 0$ ,  $a = 1$ ,  $t \in [0, 30]$ .

## 2. ДИФФЕРЕНЦИАЛЬНЫЕ УРАВНЕНИЯ В ТЕОРИИ ЭПИДЕМИИ (МОДЕЛЬ БЕЙЛИ)

### 2.1. Модель эпидемии без учета выздоровления больных особей

Рассмотрим задачу о распространении эпидемии инфекционного заболевания в рамках одной популяции [4], [5]. Пренебрегая неоднородностью распределения популяции по пространству, введем две функции  $x(t)$  и  $y(t)$ , характеризующие число незараженных и зараженных особей в момент времени  $t$ . В начальный момент времени  $t = 0$  известны начальные значения  $x(0) = n$  и  $y(0) = a$ .

Для того чтобы построить математическую модель, воспользуемся гипотезой: инфекция передается при встрече зараженных особей с незараженными. Это означает, что число незараженных особей будет убывать с течением времени пропорционально количеству встреч между зараженными и незараженными особями, т.е. пропорционально произведению  $xy$ .

На основании принятого предположения выразим убыль  $\Delta x$  незараженных особей за промежуток времени  $\Delta t$  в виде

$$\Delta x = x(t + \Delta t) - x(t) = -\beta xy \Delta t. \quad (8)$$

Величина  $\beta$  представляет собой коэффициент пропорциональности, который характеризует вероятность передачи инфекции при встречах больных и здоровых особей. В общем случае значение параметра  $\beta$  зависит от вида особи и типа болезни.

Разделим соотношение (8) на  $\Delta t$  и перейдем к пределу при  $\Delta t \rightarrow 0$ :

$$\lim_{\Delta t \rightarrow 0} \frac{\Delta x}{\Delta t} = \frac{dx}{dt} = -\beta xy. \quad (9)$$

Для замыкания модели будем считать, что болезнь не приводит к смертности, следовательно, можно написать условие баланса

$$a + n = x + y = \text{const}. \quad (10)$$

Учитывая (10), перепишем (9) и добавим начальное условие

$$\frac{dx}{dt} = -\beta x(n + a - x), \quad (11)$$

$$x(0) = n. \quad (12)$$

Формулы (11), (12) представляют собой *математическую модель динамики численности незараженных особей*.

При известном  $x(t)$  число  $y(t)$  зараженных особей определяется из условия баланса (10)

$$y = a + n - x. \quad (13)$$

Считая  $\beta$  постоянной величиной, найдем численное решение задачи (11)-(13) с помощью Python.

Listing 4: Численное решение задачи (11)-(13)

```
from scipy.integrate import odeint
import numpy as np
import matplotlib.pyplot as plt
import os

os.chdir("C:\Work")

def func4(x,t):
    y=n+a-t
    return (-b)*x*y

a=100; n=200;
x0=n;
ti=np.arange(0,1,0.1);
b1=[0.01,0.02,0.03]
col=["r","g","b"]
lab=["b=0.01","b=0.02","b=0.03"]

for i in range(len(b1)):
    b=b1[i]
    solx=odeint(func4,x0,ti)
#Построение графика функции x(t) при различных b:
    plt.subplot(121)
    plt.plot(ti,solx,col[i],label=lab[i])
    plt.xlabel("t",fontsize=17)
    plt.ylabel("x",fontsize=17)
    plt.grid(); plt.legend()
    plt.tight_layout()
#Построение графика функции y(t) при различных b:
    soly=a+n-solx
    plt.subplot(122)
    plt.plot(ti,soly,col[i],label=lab[i])
    plt.xlabel("t",fontsize=17)
    plt.ylabel("y",fontsize=17)
    plt.grid(); plt.legend()
```

```
plt.savefig("Fig4.png")
plt.show()
```

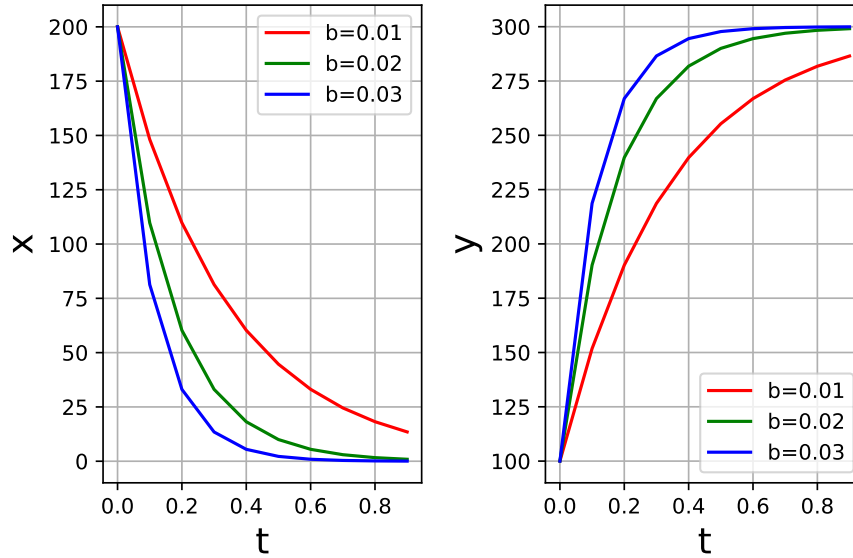


Рис. 4: Динамика численности незараженных  $x(t)$  и зараженных  $y(t)$  особей при  $\beta = \{0.01, 0.02, 0.03\}$ ,  $n = 200$ ,  $a = 100$

Примеры графиков функций  $x(t)$  и  $y(t)$  при нескольких значениях параметра  $\beta$  приведены на рис. 4. Начальные значения числа незараженных и зараженных особей приняты равными  $n = 200$ ,  $a = 100$  (Listing 4). При увеличении  $\beta$  скорость передачи инфекции увеличивается, и численность незараженных особей падает быстрее.

## 2.2. Модель эпидемии с учетом выздоровления больных особей

Изменим приведенную модель (11)-(13), добавим в нее еще один процесс – выздоровление больных особей. Для этого введем новую функцию  $z(t)$ , выражающую число выздоровевших особей. Новая математическая модель может быть представлена системой уравнений

$$\begin{cases} \frac{dx}{dt} = -\beta xy, \\ \frac{dy}{dt} = \beta xy - \gamma y, \\ \frac{dz}{dt} = \gamma y, \end{cases} \quad (14)$$



где параметр  $\gamma$  характеризует степень выздоровления и определяется видом болезни и типом особи. Число выздоровевших особей в начальный момент времени равно нулю, поэтому начальные условия для системы (14) примут вид

$$x(0) = n, \quad y(0) = a, \quad z(0) = 0. \quad (15)$$

Условие баланса (10) запишем в следующем виде:

$$x + y + z = n + a. \quad (16)$$

Найдем численное решение задачи (14)-(16) с помощью Python (Listing 5).

Listing 5: Численное решение задачи (14)-(16)

```
from scipy.integrate import odeint
import numpy as np
import matplotlib.pyplot as plt
import os

os.chdir("C:\Work\...") # путь к папке

def func5(eq, t):
    x,y,z = eq
    return [-b*x*y, b*x*y-g*y, g*y+x+y+z-n-a]

a=100; n=200; b=0.01; g=0.5;
y0=[n,a,0]
ti=np.arange(0,10,0.1)
sol=odeint(func5,y0,ti)
x=sol[:,0]; y=sol[:,1]; z=sol[:,2]

lab=["x","y","z"]
col=["r","g","b"]

for i in range(sol.shape[1]):
    plt.plot(ti,sol[:,i],col[i],label=lab[i])
    plt.xlabel("t",fontsize=17)
    plt.ylabel("x,y,z",fontsize=17)
    plt.grid()
    plt.legend()

plt.savefig("Fig5.png")
plt.show()
```

Графики показывают (рис. 5), что с ростом  $t$  все особи успевают заболеть,

т.е. величина  $x$  падает до нуля. Численность зараженных особей  $y$  сначала растет, но дальше уменьшается в связи с их выздоровлением. При  $t \rightarrow \infty$  модель предсказывает полное выздоровление всех особей.

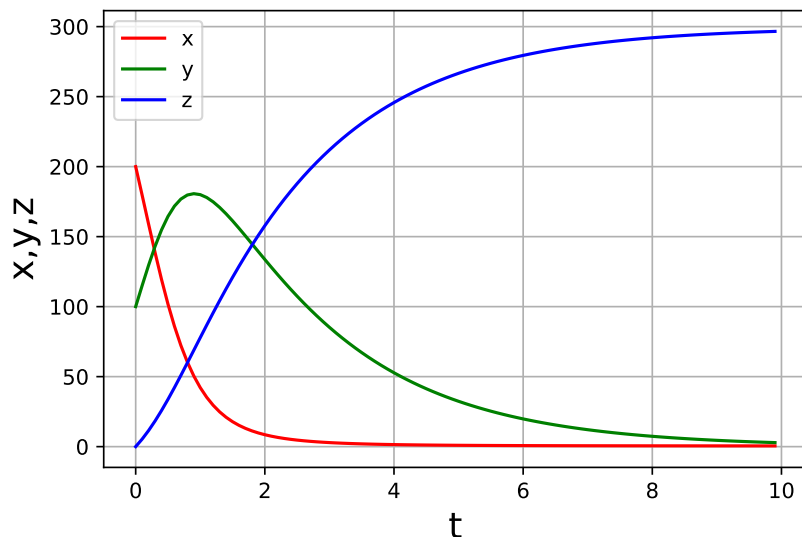


Рис. 5: Динамика численности незараженных  $x(t)$ , зараженных  $y(t)$ , выздоровевших  $z(t)$  особей

### Задания для самостоятельного решения

1. Провести исследование динамики развития эпидемии без учета выздоровления больных особей для  $\beta = 0.01$  и  $\gamma = 0.01, 0.1, 0.2$ .

2. Провести исследование динамики развития эпидемии с учетом выздоровления больных особей для  $\beta = 0.03$  и  $\gamma = 0.2$ .

## 3. ДИНАМИКА ЧИСЛЕННОСТИ ПОПУЛЯЦИИ

### 3.1. Модель Мальтуса (неограниченный рост)

Для построения математических моделей динамики численности популяций, как правило, используются различные гипотезы. Одна из простейших гипотез: скорость изменения численности популяции пропорциональна самой численности. На ее основе **Мальтусом** в 1798 г. была сформулирована модель неограниченной одиночной популяции:

$$\frac{dx}{dt} = rx, \quad (17)$$

где  $x$  – численность популяции,  $t$  – время,  $r = \alpha - \beta$ ,  $\alpha$  – коэффициент рождаемости,  $\beta$  – коэффициент смертности. Коэффициент  $r$  называют мальтузианским параметром, он характеризует репродуктивный потенциал вида. Найдем решение задачи Коши для уравнения (17) с начальным условием

$$x = x_0 \quad \text{при} \quad t_0 = 0, \quad (18)$$

используя язык Python (Listing 6).

Listing 6: Численное решение задачи (17)-(18)

```
from scipy.integrate import odeint
import numpy as np
import matplotlib.pyplot as plt
import os

os.chdir("C:\Work")

def func6(y,t):
    return r[i]*t

y0=0.2
ti=np.arange(0,1,0.1)
r=[-1,0,2]
lab=["r=-1","r=0","r=2"]
col=["r","g","b"]

for i in range(len(r)):
    sol=odeint(func6,y0,ti)
    plt.plot(ti,sol,col[i],label=lab[i])
    plt.xlabel("t",fontsize=17)
    plt.ylabel("y",fontsize=17)
    plt.grid()
```

```
plt.legend()

plt.savefig("Fig6.png")
plt.show()
```

---

Зависимость  $x(t)$  для различных коэффициентов прироста приведена на (рис. 6). Для положительных значений  $r$  модель Мальтуса предсказывает неограниченный рост популяции.

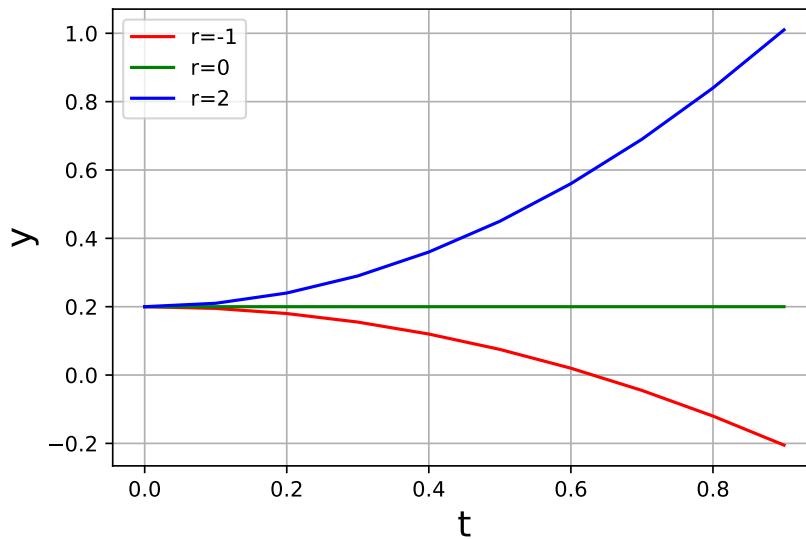


Рис. 6: Зависимость  $x$  от времени при различных  $r$

### 3.2. Логистическая модель (ограниченный рост)

Модель неограниченной популяции может описывать динамику популяции в начальный период развития, когда популяция не испытывает нехватки пищевых ресурсов. В реальных условиях любой биологический вид существует в ограниченной пищевой среде. В 1838 г. бельгийским математиком **Ферхюльстом** модель Мальтуса была впервые обобщена для случая ограниченной популяции при изучении изменений численности народонаселения, а затем в 1928 г. американским биологом **Пирлом** она была применена к различным биологическим системам.

Пусть коэффициент прироста  $r$  с увеличением численности популяции уменьшается по линейному закону

$$r \approx r_m - \gamma x,$$

где  $r_m$  – коэффициент прироста при неограниченном количестве пищи и минимальной смертности,  $\gamma$  – коэффициент, показывающий насколько сильно уменьшается величина  $r$  с ростом численности популяции. Подставляем приведенное выражение в уравнение (17). Тогда получаем:

$$\frac{dx}{dt} = x(r_m - \gamma x). \quad (19)$$

Введя параметр  $k = r_m/\gamma$  (емкость среды или равновесная численность популяции), перепишем (19) в виде

$$\frac{dx}{dt} = r_m x \left(1 - \frac{x}{k}\right). \quad (20)$$

Уравнение (20) называется логистическим уравнением Ферхюльста-Пирла.

Получим численное решение задачи Коши для уравнения (20) с начальным условием

$$x = x_0 \quad \text{при} \quad t_0 = 0 \quad (21)$$

в среде Python (Listing 7).

Listing 7: Численное решение задачи (20)-(21)

```
from scipy.integrate import odeint
import numpy as np
import matplotlib.pyplot as plt
import os
os.chdir("C:\Work")

def func7(y, t):
    return rm*y*(1-y/k[i])

rm=0.1; y0=1;
ti=np.arange(0,100,2)
k=[5,10,20]
lab=["k=5", "k=10", "k=20"]
col=["r", "g", "b"]

for i in range(len(k)):
    sol=odeint(func7,y0,ti)
    plt.plot(ti,sol,col[i],label=lab[i])
    plt.xlabel("t",fontsize=17)
    plt.ylabel("x",fontsize=17)
    plt.grid(); plt.legend();

plt.savefig("Fig7.pdf"); plt.show()
```

Графики зависимостей  $x(t)$  для различных  $k$  изображены на рис. 7. В отличие от модели Мальтуса, численность популяции растет со временем до равновесного значения  $x = k$ .

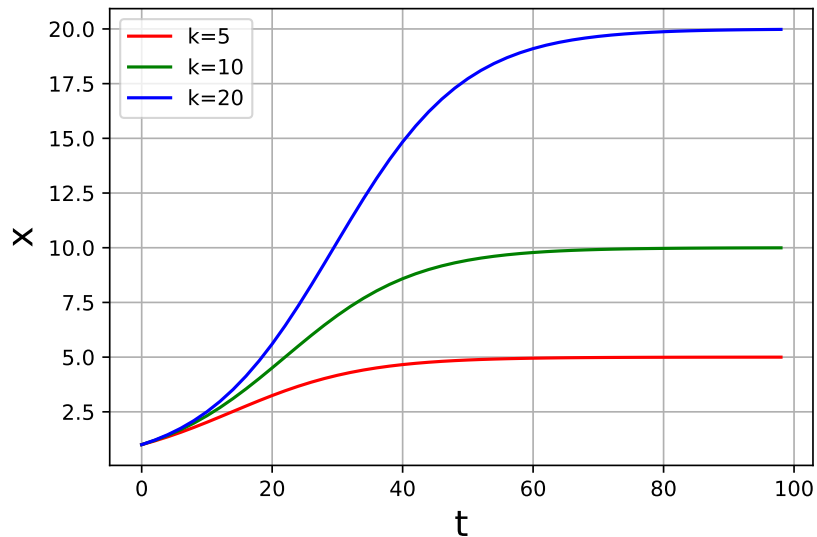


Рис. 7: Зависимость  $x$  от  $t$  при различных  $k$

### Задания для самостоятельного решения

1. Получить численное решение задачи (17)-(18), если  $x_0 = 500$ ,  $r = -0.5$  ( $r = 0.25, 0.5$ ). Построить график функции  $x = x(t)$ .
2. Получить численное решение задачи (20)-(21), если  $x_0 = 50$ ,  $k = 100$ ,  $r_m = 0.02$ . Построить график функции  $x = x(t)$ .

## 4. МОДЕЛИ ВЗАИМОДЕЙСТВИЯ ПОПУЛЯЦИЙ: "ХИЩНИК-ЖЕРТВА"

Одной из классических задач математической экологии является модель популяционной системы "хищник-жертва", описываемая **уравнениями Лотки и Вольтерры**:

$$\begin{cases} \frac{dx}{dt} = r_1x - \lambda_1xy, \\ \frac{dy}{dt} = \lambda_2xy - \beta_2y, \end{cases} \quad (22)$$

где  $x$  и  $y$  – плотности популяций жертвы и хищника,  $r_1$  – коэффициент естественного прироста жертвы (без учета поедания ее хищником),  $\beta_2$  – коэффициент смертности хищника,  $\lambda_1$ ,  $\lambda_2$  – коэффициенты, характеризующие скорость поедания жертвы хищником и обусловленную этим скорость изменения плотности хищника,  $\lambda_2 = \gamma\lambda_1$ ,  $\gamma$  – коэффициент, показывающий насколько увеличивается плотность популяции хищника при увеличении потребления пищи на единицу массы или численности. Уравнения (22), дополненные начальными условиями

$$x(0) = x_0, \quad y(0) = y_0 \quad (23)$$

представляют собой задачу Коши для системы нелинейных обыкновенных дифференциальных уравнений (Listing 8).

Listing 8: Численное решение задачи (22)-(23)

```
from scipy.integrate import odeint
import numpy as np
import matplotlib.pyplot as plt
import os

os.chdir("C:\Work")

r1=0.5; l1=0.01; l2=0.01; b2=0.2;
ti=np.arange(0,100,0.1)
y0=[25,5]
sol=odeint(func8,y0,ti)
x=sol[:,0]
y=sol[:,1]
```

```

plt.subplot(121)
plt.plot(ti,x,"r",label="x")
plt.plot(ti,y,"b",label="y")
plt.xlabel("t",fontsize=17)
plt.ylabel("x,y",fontsize=17)
plt.grid();plt.legend();
plt.title("a)")

plt.subplot(122)
plt.plot(x,y,color="g")
plt.xlabel("x",fontsize=17)
plt.ylabel("y",fontsize=17)
plt.grid()
plt.title("b)")
plt.tight_layout()
plt.savefig("Fig8.png")
plt.show()

```

Результатом выполнения программы будут зависимости плотностей популяций жертвы и хищника от времени (рис. 8a). Наблюдается характерная периодическая динамика экологической системы "хищник-жертва", выражаемая в фазовой плоскости замкнутой кривой (рис. 8b).

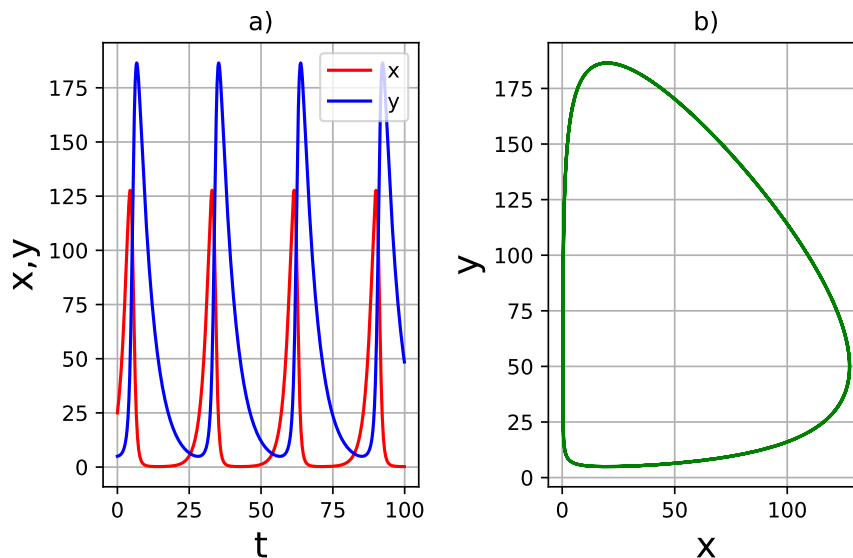


Рис. 8: а) Зависимость плотности популяций хищника и жертвы от времени при  $r_1 = 0.5; \lambda_1 = 0.01; \lambda_2 = 0.01; \beta_2 = 0.2; x_0 = 25; y_0 = 5$ ; б) Фазовый портрет системы "хищник-жертва"

Система уравнений (22) может быть дополнена новыми членами, учитывающими другие популяционные процессы, такие, например, как внутривидовая конкуренция. В случае учета внутривидовой конкуренции жертвы первое



уравнение системы (22) примет вид

$$\frac{dx}{dt} = r_1x - \lambda_1xy - g_1x^2, \quad (24)$$

где  $g_1$  – коэффициент внутривидовой конкуренции жертвы. Кривые  $x(t)$  и  $y(t)$  и фазовый портрет системы "хищник-жертва" при наличии внутривидовой конкуренции ( $g_1 = 0.0005$ ) приведены на рис. 9, а код программы на языке Python представлен на Listing 9.

Listing 9: Численное решение задачи (22)-(24)

```
from scipy.integrate import odeint
import numpy as np
import matplotlib.pyplot as plt
import os
os.chdir("C:\Work")

def func9(eq1,t):
    x1,y1=eq1
    return [r1*x1-l1*x1*y1-g1*x1**2,l2*x1*y1-b2*y1]

r1=0.5; l1=0.01; l2=0.01; b2=0.2; g1=0.0005;
ti=np.arange(0,700,0.1)
y0=[25,5]
sol=odeint(func9,y0,ti)
x=sol[:,0]
y=sol[:,1]

plt.subplot(121)
plt.plot(ti,x,"r",label="x",lw=1)
plt.plot(ti,y,"b",label="y",lw=1)
plt.xlabel("t",fontsize=17)
plt.ylabel("x,y",fontsize=17)
plt.grid();plt.legend();
plt.title("a")
plt.subplot(122)
plt.plot(x,y,color="g",lw=1)
plt.xlabel("x",fontsize=17)
plt.ylabel("y",fontsize=17)
plt.grid()
plt.title("b")
plt.tight_layout()
plt.savefig("Fig9.png")
plt.show()
```

---

Учет внутривидовой конкуренции меняет периодический характер пове-

дения плотности популяций жертвы и хищника. Наблюдаются затухающие колебания, отражением которых в фазовой плоскости является спиралевидная кривая.

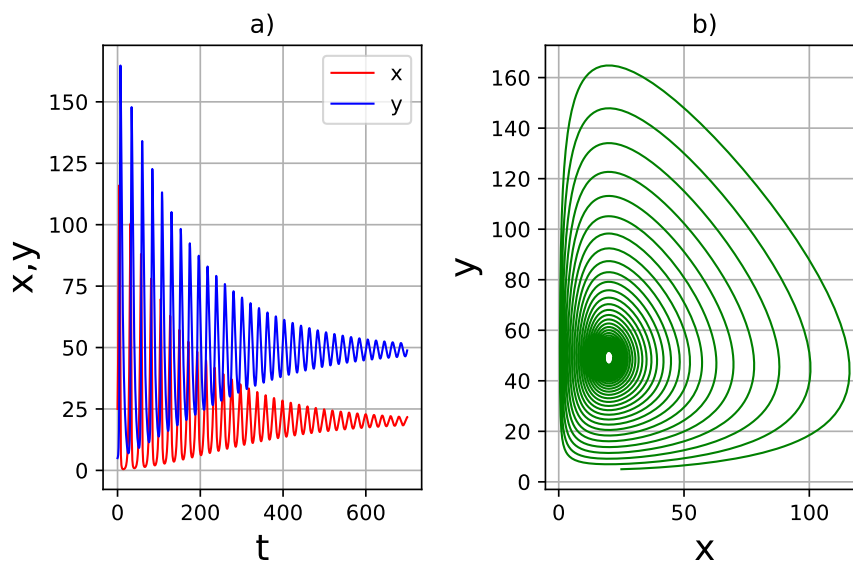


Рис. 9: а) Зависимость  $x$  и  $y$  от  $t$  с учетом внутривидовой конкуренции; б) Фазовый портрет системы "хищник-жертва" с учетом внутривидовой конкуренции

### Задания для самостоятельного решения

Используя Listing 8, написать программу для задачи Коши (22)-(23) с учетом замены первого уравнения системы на (24) при  $t \in [0, 1000]$ ,  $g_1 = 0.0005$ ,  $r_1 = 0.5$ ,  $\lambda_1 = 0.01$ ,  $\lambda_2 = 0.01$ ,  $\beta_2 = 0.2$ ,  $x_0 = 25$ ,  $y_0 = 5$ .

## 5. МОДЕЛЬ ДИНАМИКИ БИОМАССЫ МИКРООРГАНИЗМОВ С УЧЕТОМ ВЛИЯНИЯ ОСВЕЩЕННОСТИ

Суточные колебания освещенности и температуры воздушной или водной среды обитания относятся к первичным периодическим факторам. Обеспечение биоты солнечной энергией осуществляется в циклическом режиме (день – ночь), когда освещенность меняется от нуля до максимума. Изменение температуры происходит с меньшей амплитудой из-за влияния тепловой инерции земной поверхности или водной среды. Продолжительность дня и ночи измеряется часами, сезонные изменения протекают более медленно – в течение десятков суток. Амплитуда сезонного изменения среднесуточной освещенности меньше, чем суточного. В свою очередь температура в течение сезона варьируется в более широком диапазоне, так как тепловой инерции земной поверхности недостаточно для длительного сглаживания температурных колебаний, обусловленных изменяющимся притоком солнечной энергии. В течение года могут проявиться экстремальные изменения факторов окружающей среды: сильные засухи, холодные зимы. Таким образом, биота развивается в условиях циклического изменения факторов внешней среды различных временных масштабов, поэтому у нее сформировались сложные внутренние механизмы реагирования на суточные, сезонные и годовые циклы изменения условий окружающей среды.

Рассмотрим суточную динамику биомассы фототрофных микроорганизмов на основе модели Мальтуса неограниченной одиночной популяции [12]:

$$\frac{dx}{dt} = (\mu - \varepsilon)x, \quad (25)$$

где  $x$  – концентрация биомассы [г сух. вещества/л],  $\mu$  – удельная скорость роста биомассы [ч<sup>-1</sup>];  $\varepsilon$  – удельная скорость расходования биомассы [ч<sup>-1</sup>] за счет поедания растительноядными организмами и дыхания растений.

В общем случае величина  $\mu$  зависит от концентрации субстратов, температуры среды, освещенности. Будем считать, что концентрация субстратов и температура воды в водоеме в течение суток изменяются несущественно, основным внешним фактором примем освещенность  $E$  [Вт/м<sup>2</sup>].

Пусть продолжительность дня и ночи равны между собой (день равноденствия). Тогда освещенность  $E$  от момента восхода Солнца  $t = 0$  до его захода

$t = 12$  ч меняется по закону:

$$E = E_n \sin(2\pi t/24), \quad (26)$$

где  $E_n$  – освещенность в полдень (максимальная),  $t$  – время (в часах). В период от 12 до 24 ч освещенность принята равной нулю  $E = 0$ .

Полагая, что скорость процесса фотосинтеза пропорциональна освещенности  $E$ , зависимость  $\mu$  от  $E$  представим в виде гиперболической зависимости:

$$\mu = \mu_{max} \frac{E}{K_E + E}, \quad (27)$$

где  $K_E$  – коэффициент, равный освещенности, при которой  $\mu = 0.5\mu_{max}$ . Величину  $\varepsilon$  будем считать не изменяющейся в течение суток. В начальный момент времени  $x(0) = x_0$ .

Примем следующие значения параметров математической модели [12]:  $x_0 = 5$  г/л,  $E_n = 400$  Вт/м<sup>2</sup>,  $\mu_{max} = 0.3$  ч<sup>-1</sup>,  $K_E = 100$  Вт/м<sup>2</sup>,  $\varepsilon = 0.1$  ч<sup>-1</sup>. Пусть период наблюдения – трое суток. На рис. 10 приведены результаты расчетов по описанной модели (Listing 10) для зависимостей  $E(t)$  и  $\mu(t)$ . Периодический характер изменения освещенности вызывает периодическое изменение удельной скорости роста биомассы. В результате динамика роста биомассы также носит циклический характер (рис. 11). В дневное время, когда удельная скорость роста биомассы оказывается выше удельной скорости расходования биомассы, мальтузианский параметр модели положителен и наблюдается рост биомассы. В ночное время при нулевой скорости роста  $\mu(t)$  правая часть уравнения (25) будет отрицательна, что приводит к уменьшению биомассы в результате ее расходования. С течением времени биомасса в среднем уменьшается. Это связано с принятым предположением о постоянстве скорости расходования биомассы  $\varepsilon$ .

Listing 10: Численное решение задачи (25)-(27)

```
from scipy.integrate import odeint
import numpy as np
import matplotlib.pyplot as plt
import os
os.chdir("C:\Work")

En=400; mmax=0.3; Ke=100; e=0.1;

def func10(x,t):
    if t/24 - (t//24) < 0.5:
        Et=En*np.sin(2*np.pi*t/24)
```

```

        else:
            Et=0
            mu=mmax*Et/(Ke+Et)
            dxdt=(mu-e)*x
            return dxdt

ti=np.arange(0,72,0.1);
x0=5;
sol=odeint(func10,x0,ti)

plt.plot(ti,sol,lw=2)
plt.xlabel("t",fontsize=17)
plt.ylabel("x",fontsize=17)
plt.grid();plt.tight_layout();
plt.savefig("Fig10.png")
plt.show()
#-----
# Графики функций E(t), mu(t)
def func11(t):
    if t/24 - (t//24)< 0.5:
        Et=En*np.sin(2*np.pi*t/24)
    else:
        Et=0
    return Et

Eti=np.zeros(len(ti))
for i in range(len(ti)):
    Eti[i]=func11(ti[i])

mui=mmax*Eti/(Ke+Eti)
#график 1
plt.subplot(121)
plt.plot(ti,Eti,color="r",lw=2)
plt.xlabel("t",fontsize=15)
plt.ylabel("E(t)",fontsize=15)
plt.grid(); plt.title("b")
#график 2
plt.subplot(122)
plt.plot(ti,mui,color="g",lw=2)
plt.xlabel("t",fontsize=15)
plt.ylabel("mu(t)",fontsize=15)
plt.grid(); plt.title("b")
plt.tight_layout()
plt.savefig("Fig11.png")
plt.show()

```

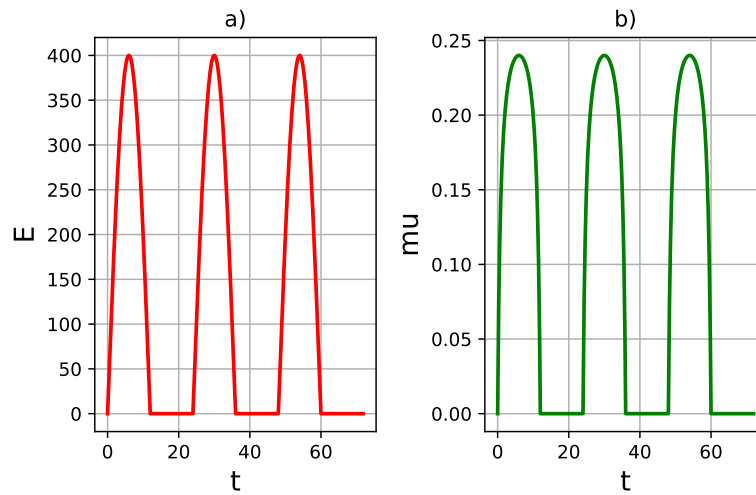


Рис. 10: а) Зависимость освещенности от времени; б) Зависимость удельной скорости роста биомассы от времени

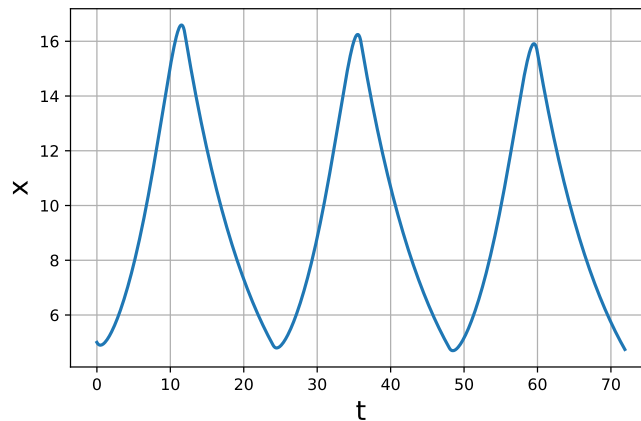


Рис. 11: Зависимость биомассы от времени

### Задания для самостоятельного решения

Провести исследование динамики биомассы фототрофных организмов в течение двух суток при следующих значениях параметров: освещенность в полдень  $E_n = 300$ , максимальная удельная скорость роста биомассы  $\mu_{max} = 0.05$ , коэффициент  $K_E = 200$ , удельная скорость расходования биомассы  $\varepsilon = 0$ .

## 6. ДИСКРЕТНЫЕ МОДЕЛИ ПОПУЛЯЦИЙ

В непрерывных моделях популяций численность или плотность популяции считается непрерывной функцией времени и/или пространственных координат. В реальности численность популяции представляет собой дискретную величину, которая принимает определенные значения в фиксированные моменты времени. Дискретные значения численности популяции могут быть получены из экспериментальных данных по изучению реальных популяций (лабораторных или полевых) в дискретные моменты времени. Если при этом предположить, что численность популяции  $x_n$  в момент времени  $t$  зависит от численностей в некоторые предшествующие моменты времени, то для описания динамики численности популяций можно применять аппарат разностных уравнений.

Исторически первой дискретной моделью в математической экологии принято считать модель динамики популяции в виде ряда чисел

$$1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89, 144, 233, 377, \dots, \quad (28)$$

приведенную в книге "Трактат о счете" (1202 год) итальянского ученого Леонардо Фибоначчи. Числовая последовательность (28) вошла в историю как ряд чисел Фибоначчи, а его члены – числа Фибоначчи. Числа в последовательности (28) выражают динамику роста численности кроликов от поколения к поколению согласно простому правилу: каждый последующий член последовательности равен сумме двух предыдущих. Приведенная модель выражает гипотезу, что количество воспроизводимых кроликов в данном поколении равно сумме кроликов в двух предыдущих поколениях.

В настоящее время дискретные модели широко применяются для исследования динамики популяций и относятся к важной группе математических моделей в экологии. В той или иной степени дискретные модели популяций описаны в работах [7] - [11], которые были использованы при написании данного пособия. В дискретных моделях время представляется как дискретная переменная, и наблюдения выполняются лишь через определенные фиксированные интервалы времени (ежечасно, ежегодно, через каждые 10 лет и т.п.).

Введем дискретную переменную  $x_n$ , представляющую собой численность популяции к концу  $n$ -го периода времени. Тогда  $x_0, x_1, x_2, \dots, x_n, x_{n+1}, \dots$  – последовательность чисел, описывающая развитие популяции во времени. На практике обычно известна начальная численность популяции и скорость

роста популяции в разные периоды времени. Для определения численности популяции  $x_n$  вводится понятие разностного уравнения.

**Определение.** *Разностное уравнение* – уравнение, которое связывает между собой значения численности популяции  $x_n$  при различных значениях индекса  $n$ . Если  $N_{max}$  и  $N_{min}$  представляют собой наибольший и наименьший из индексов  $n$ , встречающихся в записи уравнения, то порядок разностного уравнения будет равен  $N_{max} - N_{min}$ . Различают линейные и нелинейные разностные уравнения. Разностные уравнения, содержащие  $x_n$ ,  $x_{n+1}$  и т.д. в степени выше первой и/или их произведения, являются *нелинейными*.

Аппарат разностных уравнений широко используется в вычислительной математике при определении дискретных значений функций, получаемых в результате дискретизации задачи для дифференциальных уравнений, описывающих различные физические процессы. Разностные уравнения в задачах математической экологии могут быть записаны как математические выражения различных гипотез, формулируемых относительно динамики численности популяций.

### 6.1. Дискретная модель с неограниченным ростом численности популяции

Рассмотрим модель одиночной неограниченной популяции (17)

$$\frac{dx}{dt} = rx,$$

где  $x$  – численность популяции,  $t$  – время,  $r$  – коэффициент прироста.

Заменяя производную  $dx/dt$  на  $\Delta x/\Delta t = (x_{t_{n+1}} - x_{t_n})/\Delta t$ , получаем

$$\frac{x_{t_{n+1}} - x_{t_n}}{\Delta t} = rx_{t_n}$$

или

$$x_{t_{n+1}} = (1 + r\Delta t)x_{t_n}, \quad (29)$$

где  $x_{t_{n+1}} = x(t_{n+1})$  – численность популяции в момент времени  $t_{n+1}$ , следующий за  $t_n$ . Уравнение (29) – разностное уравнение, которое является *дискретным аналогом модели одиночной неограниченной популяции* (17). В силу того, что  $x(t_n) = x_{t_n} = x_n$ , уравнение (29) часто записывают в виде

$$x_{n+1} = a x_n, \quad (30)$$



где

$$a = 1 + r\Delta t.$$

Запишем уравнение (30) в виде:

$$\begin{aligned}x_1 &= ax_0, \\x_2 &= ax_1 = a^2x_0, \\&\dots \\x_n &= a^n x_0.\end{aligned}$$

При известном начальном значении  $x_0$  можно рассчитать динамику популяции во времени. В зависимости от коэффициента  $a = 1 + r\Delta t$  возможны следующие ситуации:

- 1)  $a > 1$ , т.е.  $r\Delta t > 0 \Rightarrow a^n \rightarrow \infty$  при  $n \rightarrow \infty$  – неограниченный рост;
- 2)  $a = 1$  – численность популяции не меняется;
- 3)  $0 < a < 1$ , т.е.  $-1 < r\Delta t < 0 \Rightarrow a^n \rightarrow 0$  – вымирание популяции;
- 4)  $a = 0$  – вымирание за один период времени;
- 5)  $a < 0$ , т.е.  $r\Delta t < -1$  – отрицательные численности (нереальная ситуация).

Задавая различные значения параметра  $r$  (коэффициента роста популяции) и начальной численности  $x_0$  популяции, можно получить качественно различные типы поведения переменной, удовлетворяющей разностному уравнению (29).

Результаты численных расчетов по формуле (29) с начальным условием

$$x(t_0) = x_0 \tag{31}$$

получены по программе Listing 11 и представлены на рис. 12.

Listing 11: Численное решение задачи (29), (31)

```
import numpy as np
import matplotlib.pyplot as plt
import os
os.chdir("C:\Work")

r=1/5 # значения параметра модели: r=1/5; -1/5
Tstop=30; N=int(Tstop/dT)

x=np.zeros(N) # создание вектора x=(0,...,0)
x[0]=2 # начальное условие
ti=np.arange(0, Tstop, dT)

for k in range(N-1):
```

```
x[k+1]=(1+r*dT)*x[k] # разностное уравнение
```

```
plt.plot(ti,x,lw=2,color="r",label="r=1/5")  
plt.xlabel("t",fontsize=15)  
plt.ylabel("x(t)",fontsize=15)  
plt.title("a)")  
plt.grid(); plt.legend()  
plt.tight_layout()  
plt.savefig("Fig12.png")  
plt.show()
```

---

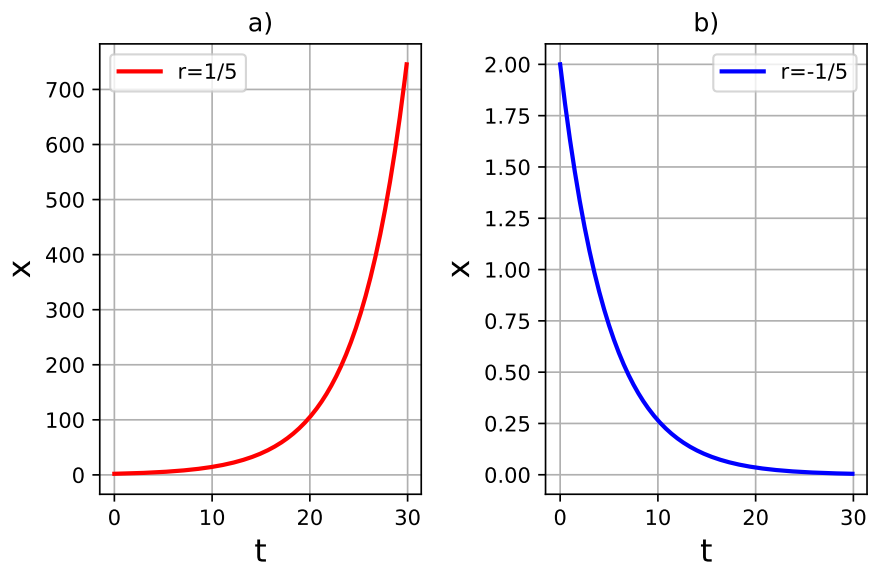


Рис. 12: Зависимость  $x$  от времени  $t$  при начальном условии  $x_0 = 2$  и различных значениях параметра модели а)  $r = 1/5$ ; б)  $r = -1/5$

## 6.2. Дискретная логистическая модель (ограниченный рост численности популяции)

Как было отмечено ранее, дискретная модель (30) при коэффициенте прироста  $a = (1 + r\Delta t) > 1$  предсказывает неограниченный рост численности популяции. В реальности ни одна популяция не может увеличиваться бесконечно вследствие ограниченности пищевых ресурсов и других ограничивающих внешних факторов. Для учета этого обстоятельства введем *условие ограничения роста*.

Пусть коэффициент прироста  $a$  будет зависеть от численности популяции, а именно, будет убывать по мере роста численности популяции по закону  $a \sim a(1 - x_n)$ . Тогда уравнение (30) примет вид

$$x_{n+1} = a x_n(1 - x_n). \quad (32)$$

Уравнение (32) называется *логистическим уравнением* и является *дискретным аналогом модели Ферхюльста–Пирла* (20). Это уравнение может описывать не только динамику популяций, но и многие другие явления в природе, экономике и обществе.

Отметим, что величина  $x$  в уравнении (32) меняется от 0 до 1, а параметр  $a$  должен быть больше 0 и меньше 4. При других значениях  $x$  и  $a$  логистическое уравнение дает отрицательные значения численности популяции.

Задавая различные значения параметра модели  $a$  и начальной численности популяции  $x_0$ , получим различные типы поведения переменной  $x$ , удовлетворяющей разностному уравнению (32) (Listing 12).

Listing 12: Численное решение уравнения (32)

```
import numpy as np
import matplotlib.pyplot as plt
import os
os.chdir("C:\WORK")

a=0.5; # значения параметра модели: a=0.5;2;3.3;3.9
N=20
x=np.zeros(N+1)
x[0]=0.2 # начальное условие

for k in range(N):
    x[k+1]=a*x[k]*(1-x[k]) # разностное уравнение

ti=np.arange(0,N+1,1)
```

```
plt.plot(ti,x,color="r",label="a=0.5")
plt.xlabel("n")
plt.ylabel("x")
plt.grid(); plt.legend()
plt.savefig("Fig13.png")
plt.show()
```

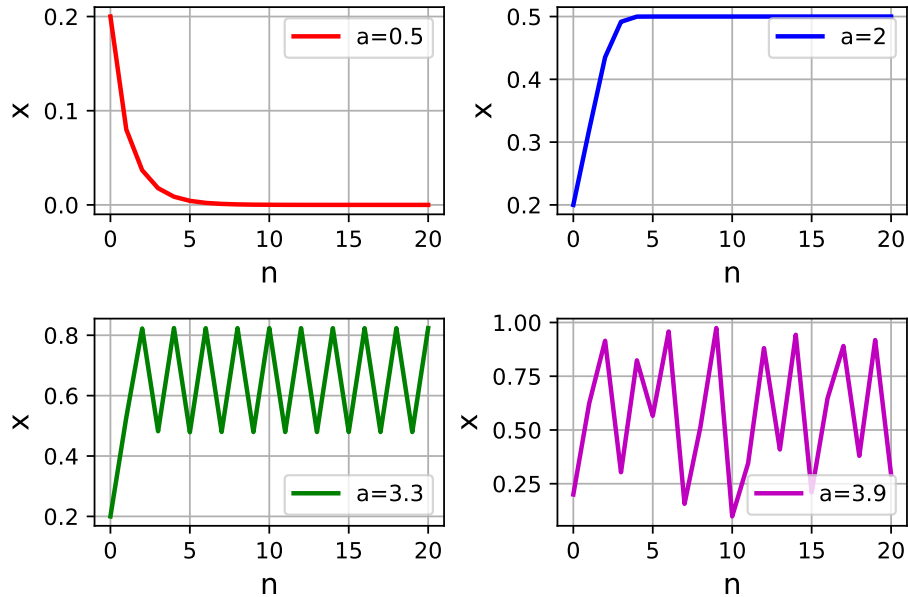


Рис. 13: Зависимость  $x$  от номера  $n$  момента времени  $t_n$  при различных значениях  $a$

На рис. 13 показаны графики зависимости  $x$  от номера  $n$  момента времени  $t_n$  для разных значений параметра  $a$ . Начальное значение численности популяции во всех расчетах было принято равным  $x_0 = 0.2$ . Для значения  $a = 0.5$  популяция за несколько периодов времени приходит к вымиранию (нулевое значение  $x_n$  при  $n \rightarrow \infty$ ). При  $a = 2$  получаем единственное равновесное значение численности  $x_n = 0.5$ . Из рис.13 видно, что при  $a = 3.3$  конечное значение численности популяции начинает осциллировать между двумя уровнями, которые соответствуют значениям 0.8296 и 0.471, то есть реализуется периодический режим.

С ростом коэффициента  $a$  динамика системы усложняется. Так, для  $a = 3.9$  можно наблюдать, что процесс перестает быть периодическим, численность популяции принимает новые неповторяющиеся значения. Такое поведение называется *нерегулярным или хаотическим*. Природа столь сложного поведения решения логистического уравнения заключена в нелинейности правой части уравнения, которая является квадратичной функцией.

### 6.3. Модель Рикера (ограниченный рост численности популяции)

В теории популяций часто используют исправленную модель ограниченной одиночной популяции, а именно модель Рикера:

$$x_{n+1} = x_n e^{a(1-x_n/k)}, \quad (33)$$

где параметр  $a$  интерпретируется как внутренняя скорость роста популяции, а  $k$  – как биологическая емкость среды. Модель Рикера описывает количество особей  $x$  в дискретный момент времени  $t_{n+1}$  в зависимости от количества особей предыдущего поколения в момент времени  $t_n$ .

Опишем результаты численных исследований динамики популяции согласно модели Рикера (33) (Listing 13).

Listing 13: Численное решение уравнения (33) с  $x_0 = 0.1$

```
import numpy as np
import matplotlib.pyplot as plt
import os
os.chdir("C:\WORK")

a=0.5; # значения параметра модели: a=0.5;1.8;2.5;3.5
N=20
k=1

x=np.zeros(N+1)
x[0]=0.1 # начальное условие
ti=np.arange(0,N+1,1)

for i in range(N):
    x[i+1]=x[i]*np.exp(a*(1-x[i]/k)) # разностное уравнение

plt.plot(ti,x,lw=2,color="r",label="a=0.5")
plt.xlabel("n",fontsize=13)
plt.ylabel("x",fontsize=13)
plt.grid();plt.legend()
plt.savefig("Fig14.png")
plt.show()
```

Начальное значение численности популяции во всех расчетах принято равным  $x_0 = 0.1$  и  $k = 1$ . Приведенные на рис. 14 зависимости численности популяции от номера  $n$  момента времени ( $t_n$ ) дают представление о динамике системы в рамках модели Рикера при варьировании коэффициента  $a$ .

При  $a = 0.5$  численность популяции монотонно возрастает до миниму-

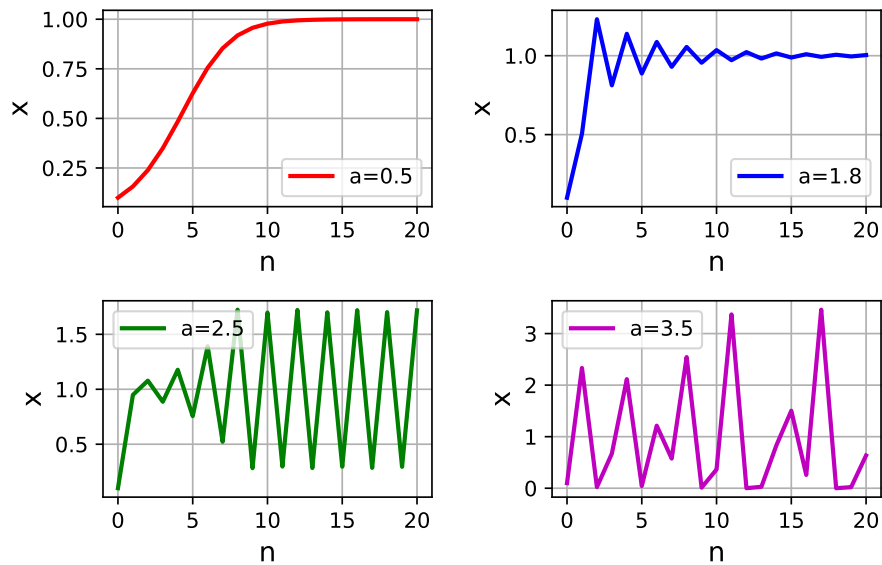


Рис. 14: Зависимость  $x$  от номера  $n$  момента времени  $t_n$  при различных значениях  $a$

ма своей численности и остается дальше неизменной на этом уровне. При  $a = 1.8$  изменения численности представляют собой затухающие колебания. При  $a = 2.5$  наблюдается двухточечный цикл. При  $a > 3$  можно наблюдать, что процесс перестает быть периодическим (нерегулярное или хаотическое поведение).

Итак, расчеты показывают, что при

- $0 < a \leq 2$  численность популяции будет стремиться к одному определенному значению;
- $2 < a \leq 2.692$  численность популяции будет подчиняться колебательному режиму, который останется периодическим и устойчивым бесконечно долго;
- $a > 2.692$  изменение численности популяции будет иметь хаотический характер.

Таким образом, численные расчеты по модели Рикера выявляют спектр типов поведения численности популяции: выход на стабильный уровень, периодические колебания и хаотическое поведение. Многообразие решений для модели Рикера обусловлено нелинейностью модели, также как и для логистической модели.

#### 6.4. Дискретная модель популяции с учетом возрастной структуры

Для учета в модели популяции выживаемости особей разных возрастных групп их представляют как дискретные возрастные классы, численность которых зависит от численности предшествующих (а в отдельных случаях, и всех остальных) возрастных классов. Задача описания динамики возрастных классов популяций приводит к дискретным матричным моделям или к системе разностных уравнений.

В работе [13] была предложена и исследована модель динамики численности популяции с возрастной структурой, которая может быть представлена совокупностью двух возрастных классов: младшего, включающего неполовозрелых особей, и старшего, состоящего из особей, участвующих в размножении. Рассмотрим подробнее данную модель.

Обозначим численность младшего возраста в  $n$ -й сезон размножения через  $x_n$ , а численность репродуктивного поколения через  $y_n$ . Период размножения заканчивается появлением новорожденных особей нового поколения. Предполагается, что времени между двумя последовательными периодами размножения достаточно для полного развития младенцев до взрослого состояния, а новорожденных – до состояния младшего возраста. Коэффициенты выживаемости и плодовитости зрелых особей считаются постоянными. Принятое предположение характерно для организмов с небольшим периодом жизни, включающим два-три периода размножения: насекомые, рыбы, мелкие млекопитающие, двух- или трехлетние растения и др.

Дискретная модель двухвозрастной популяции представляется в виде двух разностных уравнений

$$\begin{aligned}x_{n+1} &= by_n, \\y_{n+1} &= x_n(1 - x_n) + cy_n,\end{aligned}\tag{34}$$

где  $b$  – произведение коэффициентов рождаемости и выживаемости приплода на первом году жизни, а  $c$  – выживаемость половозрелых особей. В работе [13] показано, что все множество допустимых значений параметров  $b$  и  $c$  ( $b > 0$ ,  $0 < c < 1$ ) системы (34) можно разбить на три области:

- 1)  $b + c < 1$  – в этой области для (34) существует только устойчивое положение нулевого равновесия  $x = 0$ ,  $y = 0$ ;
- 2)  $b + c > 1$ ,  $b + 2c < 3$  – существует устойчивое ненулевое положение равновесия;
- 3)  $b + 2c > 3$  – существуют неустойчивые нулевое и ненулевое положения

равновесия (иначе говоря, стационарные точки) системы (34).

Как и в [13], были проведены численные исследования поведения системы (34) при  $n \rightarrow \infty$  в области значений  $b$  и  $c$ , удовлетворяющих условию  $b + 2c > 3$ .

Listing 14: Численное решение задачи (34)

```
import numpy as np
import matplotlib.pyplot as plt
import os
os.chdir("C:\WORK")

b=2.8; # значения параметра модели: b=2.8;3.10;3.22;3.31
c=0.15
N=30
x=np.zeros(N+1)
y=np.zeros(N+1)
x[0]=0.2
y[0]=0.1

for k in range(N):
    x[k+1]=b*y[k]
    y[k+1]=x[k]*(1-x[k])+c*y[k]

ti=np.arange(0,N+1,1)
plt.plot(ti,x,color="lightpink",label="x",lw=2)
plt.plot(ti,y,color="r",label="y",lw=2)
plt.xlabel("n")
plt.ylabel("x, y")
plt.grid();plt.legend();
plt.savefig("Fig15.png")
plt.show()
```

Для расчетов (Listing 14) полагалось  $c = 0.15$ , а значение параметра  $b = \{2.8; 3.10; 3.22; 3.31\}$ , начальные значения численности двух поколений принимались равными  $x_0 = 0.2$ ,  $y_0 = 0.1$ .

Представление о динамике численности неполовозрелых и половозрелых особей дает рис. 15. Наблюдаются нерегулярные колебания численностей с изменением периода времени.

Аттракторы системы (34), т.е. установившееся устойчивое поведение системы для различных значений  $b$ , приводятся на рис. 16 (Listing 15). Расчеты проводились до  $n = 10000$ , для представления на графике сохранялись значения численности популяции при  $n = 6000$ . Рисунки показывают равновесные состояния системы в фазовом пространстве, к которым стремятся



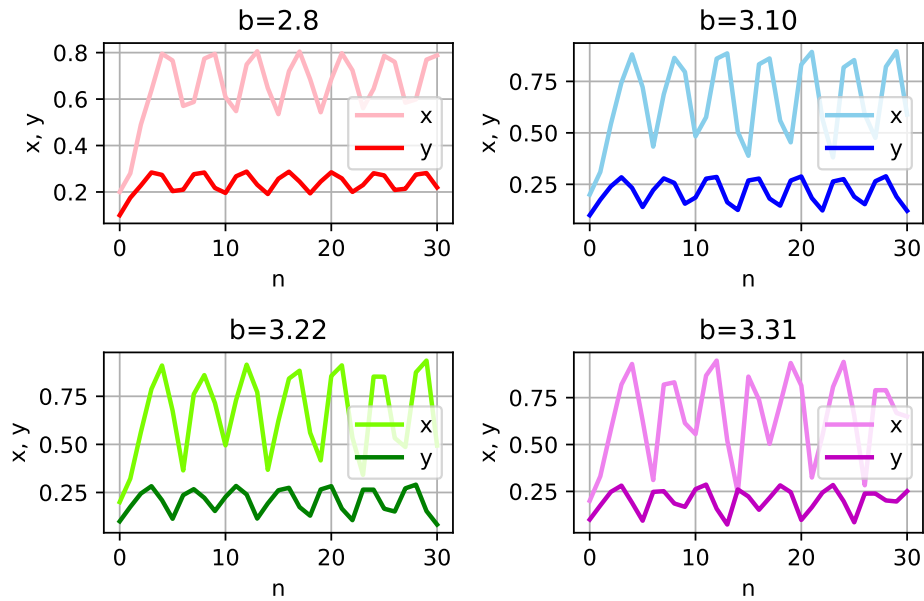


Рис. 15: Динамика численности двух поколений популяции при  $x_0 = 0.2$ ,  $y_0 = 0.1$ ,  $c = 0.15$

переменные  $x$ ,  $y$  при  $n \rightarrow \infty$ . По мере изменения параметра  $b$  характер линий значительно меняется. При  $b = 2.8$  аттрактор системы (34) представляет собой замкнутую кривую в фазовом пространстве – предельный цикл. С увеличением параметра  $b$  замкнутая кривая трансформируется в области со сложной структурой. Все линии утолщаются, и постепенно множество точек траектории более или менее плотно закрывает некоторую область фазового пространства. Такое поведение системы позволяет предположить наличие сложной серии бифуркаций, т.е. скачкообразных изменений аттракторов.

Listing 15: Численное решение задачи (34). Фазовая плоскость

```
import numpy as np
import matplotlib.pyplot as plt
import os
os.chdir("C:\WORK")

b=2.8; # значения параметра модели: b=2.8;3.10;3.22;3.31
c=0.15
N=6000
x=np.zeros(N+1); y=np.zeros(N+1)
x[0]=0.2; y[0]=0.1

for k in range(N):
    x[k+1]=b*y[k]
    y[k+1]=x[k]*(1-x[k])+c*y[k]
```

```

# фазовая плоскость
plt.scatter(x,y,marker="o",color="r",s=2)
plt.xlabel("x")
plt.ylabel("y")
plt.title("b=2.8")

plt.savefig("Fig16.png")
plt.show()

```

Таким образом, модель популяции с двумя возрастными классами (34) описывает разнообразие вариантов динамики численности популяции, также как и модели без выделения возрастных классов (30), (32), (33).

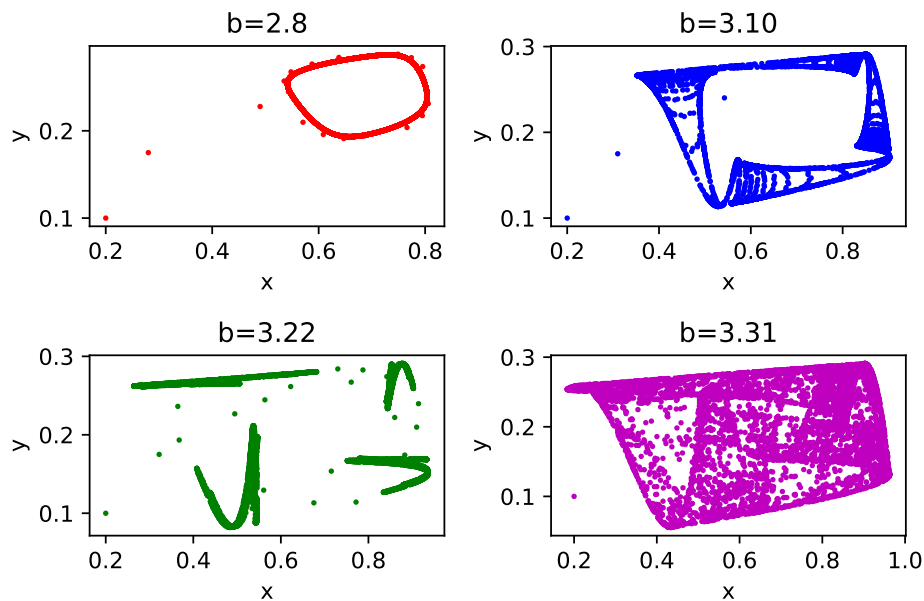


Рис. 16: Аттрактор системы (34) при  $x_0 = 0.2$ ,  $y_0 = 0.1$ ,  $c = 0.15$

Из результатов анализа двух моделей (33) и (34) можно сделать один общий вывод: даже самые простые детерминированные дискретные модели динамики численности одиночных популяций могут приводить к сложному поведению решения, характеризуемому циклическими или нерегулярными хаотическими колебаниями численности популяции. В математическом плане причиной появления периодических или нерегулярных решений является нелинейность моделей. Даже при постоянных параметрах динамика рассматриваемых систем содержит различные колебания, в том числе и нерегулярные. Косвенно параметры модели могут учитывать воздействие различных внешних факторов. Однако, сложное поведение в динамической системе может быть связано с внутренней сущностью системы и может проявиться в отсутствие влияния внешних факторов.

## Задания для самостоятельного решения

### 1. Модель Рикера

$$x_{n+1} = x_n e^{a(1-x_n/k)}$$

Исследовать динамику популяции при следующих параметрах:

$$a = 0.1; 0.5; 1; 1.9; 2; 2.3; 2.6; 3; k = 100; 200.$$

### 2. Модель "хищник-жертва"

$$x_{n+1} = x_n + a_1 x_n - b_1 x_n y_n,$$

$$y_{n+1} = y_n - a_2 y_n + b_2 x_n y_n.$$

Исследовать динамику двух популяций. Для анализа модели принять в качестве базовых следующие значения:

$$x_0 = 26, y_0 = 7, a_1 = 0.01, a_2 = 0.02, b_1 = 0.002, b_2 = 0.001.$$

### 3. Модель эпидемии

$$x_{n+1} = x_n - b x_n y_n,$$

$$y_{n+1} = y_n + b x_n y_n,$$

где  $x$  – численность здоровых особей,  $y$  – численность больных особей, а  $b$  – коэффициент передачи инфекции. Для анализа модели принять в качестве базовых следующие значения:

$$x_0 = 10, y_0 = 1, b = 0.05; 0.03; 0.02; 0.01.$$

## ЛИТЕРАТУРА

- [1] Доля П.Г. Введение в научный Python / П.Г. Доля. – Харьков: Харьковский национальный университет, 2016. – 333 с.
- [2] Доусон М. Программируем на Python / М. Доусон. – СПб.: Питер, 2014. – 416 с.
- [3] Лутц М. Изучаем Python / М. Лутц. – СПб.: Символ–Плюс, 2011. – 1280 с.
- [4] Бэйли Р. Математика в биологии и медицине / Р. Бэйли. – М.: Мир, 1970. – 326 с.
- [5] Гильдерман Ю.И. Лекции по высшей математике для биологов / Ю.И. Гильдерман. – Новосибирск: Наука, 1974. – 409 с.
- [6] Зарипов Ш.Х. Задачи математической экологии и пакет Maxima / Ш.Х. Зарипов, Д.Ф. Абзалилов, Е.А. Костерина. – Казань, 2015. – 120 с.
- [7] Жигулев В.Н. Динамика неустойчивостей / В.Н. Жигулев. – М.: МФТИ, 1996. – 344 с.
- [8] Ризниченко Г.Ю. Математические модели биологических продукционных процессов / Г.Ю. Ризниченко, А.Б. Рубин. – М.: Изд-во МГУ, 1993. – 302 с.
- [9] Ризниченко Г.Ю. Математические модели в биофизике и экологии / Г.Ю. Ризниченко. – Иж.: ИКИ, 2003. – 184 с.
- [10] Скалецкая Е.И. Дискретные модели динамики численности популяций и оптимизация промысла / Е.И. Скалецкая, Е.Я. Фрисман, А.П. Шапиро. – М.: Наука, 1979. – 166 с.
- [11] Семенова Е.Е. Математические методы в экологии: Сборник задач и упражнений / Е.Е. Семенова, Е.В. Кудрявцева. – Петрозаводск: Изд. ПетрГУ, 2005. – 130 с.
- [12] Фролов Ю.П. Введение в математическое моделирование биологических процессов. Часть 2. Организмы и популяции / Ю.П. Фролов. – Самара: Изд-во Самарского ун-та, 1994. – 317 с.
- [13] Фрисман Е.Я. Странные аттракторы в простейших моделях динамики численности популяции с возрастной структурой / Е.Я. Фрисман // Доклады Академии Наук, 1994. – Т.338. – №2. – С.282–286.