Отчёт по лабораторной работе 3

дисциплина: Математическое моделирование

Савченков Д.А., НПИбд-02-18

Содержание

[Цель работы 1](#_Toc65323391)

[Задание 1](#_Toc65323392)

[Выполнение лабораторной работы 2](#_Toc65323393)

[Выводы 5](#_Toc65323394)

# Цель работы

Построить упрощенную модель боевых действий с помощью Python.

# Задание

**Вариант 38** Между страной и страной идет война. Численности состава войск исчисляются от начала войны и являются временными функциями и . В начальный момент времени страна имеет армию численностью 882 0000 человек, а в распоряжении страны армия численностью в 747 000 человек. Для упрощения модели считаем, что коэффициенты постоянны. Также считаем и непрерывными функциями.

Постройте графики изменения численности войск армии и армии для следующих случаев:

1. Модель боевых действий между регулярными войсками
2. Модель ведение боевых действий с участием регулярных войск и партизанских отрядов

# Выполнение лабораторной работы

**1. Боевые действия между регулярными войсками**

1.1. Изучил начальные условия. Коэффициент смертности, не связанный с боевыми действиями, у первой армии 0,4, а у второй – 0,14. Коэффициент эффективности первой и второй армии 0,77 и 0,67 соответственно. Функция, описывающая подход подкрепление первой армии, , подкрепление второй армии описывается функцией . – численность 1-ой армии, – численность 2-ой армии.

1.2. Оформил начальные условия в код на Python:

x0 = 882000  
y0 = 747000  
  
a1 = 0.4  
b1 = 0.67  
c1 = 0.77  
h1 = 0.14  
  
def P1(t):  
 p1 = np.sin(3\*t)+1  
 return p1  
def Q1(t):  
 q1 = np.cos(2\*t)+2  
 return q1

1.3. Для времени задал следующие условия: – начальный момент времени, – предельный момент времени, – шаг изменения времени.

1.4. Добавил в программу условия, описывающие время:

t0 = 0  
tmax = 1  
dt = 0.05  
t = np.arange(t0, tmax, dt)

1.5. Запрограммировал заданную систему дифференциальных уравнений, описывающих изменение численности армий:

def S1(f, t):  
 s11 = -a1\*f[0] - b1\*f[1] + P1(t)  
 s12 = -c1\*f[0] - h1\*f[1] + Q1(t)  
 return s11, s12

1.6. Создал вектор начальной численности армий:

v = np.array([x0, y0])

1.7. Запрограммировал решение системы уравнений:

f1 = odeint(S1, v, t)

1.8. Описал построение графика изменения численности армий:

plt.plot(t, f1)

**2. Боевые действия с участием регулярных войск и партизанских отрядов**

2.1. Изучил начальные условия. Коэффициент смертности, не связанный с боевыми действиями, у первой армии 0,24, а у второй – 0,14. Коэффициент эффективности первой и второй армии 0,47 и 0,67 соответственно. Функция, описывающая подход подкрепление первой армии, , подкрепление второй армии описывается функцией . Изначальная численность армий такая же, как и в п. 1.1.

2.2. Дополнил начальные условия в коде на Python:

a2 = 0.24  
b2 = 0.67  
c2 = 0.47  
h2 = 0.14  
  
def P2(t):  
 p2 = np.fabs(np.sin(2\*t))  
 return p2  
def Q2(t):  
 q2 = np.fabs(np.cos(2\*t))  
 return q2

2.3. Для времени задал следующие условия: – начальный момент времени, – предельный момент времени, – шаг изменения времени. Далее смотреть пункт 1.4 .

2.4. Запрограммировал заданную систему дифференциальных уравнений, описывающих изменение численности армий:

def S2(f, t):  
 s21 = -a2\*f[0] - b2\*f[1] + P2(t)  
 s22 = -c2\*f[0]\*f[1] - h2\*f[1] + Q2(t)  
 return s21, s22

2.5. Т. к. начальная численность армий не изменилась, вектор начальных условий не менял.

2.6. Запрограммировал решение системы уравнений:

f2 = odeint(S2, v, t)

2.7. Описал построение графика изменения численности армий:

plt.plot(t, f2)

**3. Сборка программы**

3.1. Собрал код программы воедино и получил следующий код:

import math  
import numpy as np  
from scipy.integrate import odeint  
import matplotlib.pyplot as plt  
  
x0 = 882000  
y0 = 747000  
  
a1 = 0.4  
b1 = 0.67  
c1 = 0.77  
h1 = 0.14  
  
a2 = 0.24  
b2 = 0.67  
c2 = 0.47  
h2 = 0.14  
  
t0 = 0  
tmax = 1  
dt = 0.05  
t = np.arange(t0, tmax, dt)  
  
def P1(t):  
 p1 = np.sin(3\*t)+1  
 return p1  
def Q1(t):  
 q1 = np.cos(2\*t)+2  
 return q1  
  
def P2(t):  
 p2 = np.fabs(np.sin(2\*t))  
 return p2  
def Q2(t):  
 q2 = np.fabs(np.cos(2\*t))  
 return q2  
  
def S1(f, t):  
 s11 = -a1\*f[0] - b1\*f[1] + P1(t)  
 s12 = -c1\*f[0] - h1\*f[1] + Q1(t)  
 return s11, s12  
  
def S2(f, t):  
 s21 = -a2\*f[0] - b2\*f[1] + P2(t)  
 s22 = -c2\*f[0]\*f[1] - h2\*f[1] + Q2(t)  
 return s21, s22  
  
v = np.array([x0, y0])  
  
f1 = odeint(S1, v, t)  
f2 = odeint(S2, v, t)  
  
plt.plot(t, f1)  
plt.ylabel('Численность армии')  
plt.xlabel('Время')  
plt.legend(['Армия X', 'Армия Y'])  
  
plt.plot(t, f2)  
plt.ylabel('Численность армии')  
plt.xlabel('Время')  
plt.legend(['Армия X', 'Армия Y'])

3.2. Получил графики изменения численностей армий (см. рис. 1 и 2):

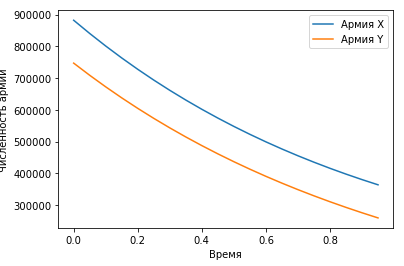


Figure 1: Боевые действия между регулярными войсками

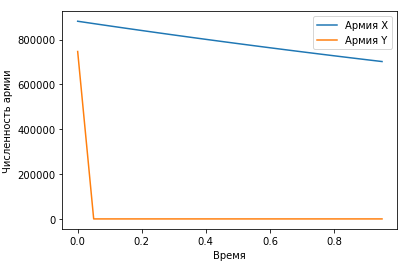


Figure 2: Боевые действия с участием регулярных войск и партизанских отрядов

# Выводы

Построил упрощенную модель боевых действий с помощью Python.

В боевых действиях между регулярными войсками победит армия X за долгий промежуток времени, как показывает график(видим по графику, что численность армии Y будет на исходе под конец войны).

В боевых действиях с участием регулярных войск и партизанских отрядов также победит армия Х, но уже намного быстрее, чем в 1-ом случае (видим по графику, что армия Y потеряла всех бойцов, практически, в самом начале войны).