

НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
ІМЕНІ ІГОРЯ СІКОРСЬКОГО»
Факультет прикладної математики
Кафедра прикладної математики

Звіт
із лабораторної роботи №2
з дисципліни «Теорія керування»
на тему
«Математичне моделювання епідемій»

Виконала:

студентка групи КМ-03
Ковальчук А. С.

Перевірили:

Професор ПМА ФПМ
Норкін В.І.
Асистент ПМА ФПМ
Жук І.С.

Зміст

Постановка задачі.....	3
Теоретичні відомості.....	4
Порядок виконання роботи.....	5
Основна частина.....	6
Дослідження параметру N (Загальне число індивідів)	8
Дослідження параметру r (Інтенсивність контактів індивіда).....	9
Дослідження параметру s (Ймовірність зараження при контакті з хворим)	10
Дослідження параметру u (Відсоток вакцинованих за час).....	11
Висновок.....	12
Використана література	13
Додаток А – Код програми	14

Постановка задачі

Проста неперервна динамічна модель епідемії в популяції з N індивідів SIR-модель (Susceptible, Infected, Recovered)

Введемо наступні величини:

N - загальне число індивідів;

$S(t)$ - число здорових індивідів (без імунітету, susceptibles);

$I(t)$ - число хворих і заразних (Infected);

$R(t)$ - число індивідів з імунітетом (здорових с імунітетом та тих, що видужали з придбаним імунітетом);

$D(t)$ - число померлих;

$u(t)$ - доля вакцинованих (ізолюваних) в одиницю часу;

$p(t)dt = p(I(t)) \cdot dt$ - ймовірність заразитися здоровому за час dt ;

$S(t) \cdot p(I(t))dt$ - середнє число заражених за час dt ;

$\alpha \cdot dt$ - ймовірність одужання інфікованого за час dt ;

$\beta \cdot dt$ - ймовірність смерті інфікованого за час dt ;

$u(t) \cdot dt$ - (керований) відсоток вакцинованих за час dt ;

$S(t) \cdot u(t)dt$ - число вакцинованих за час dt .

Модель зараження (ймовірності зараження $p(I)$)

Нехай

r - інтенсивність контактів індивіда, $r \cdot dt$ - число контактів за час dt ;

c - ймовірність зараження при контакті з хворим.

Вочевидь, I/N - ймовірність зустріти хворого при будь-якому контакті.

Яка ймовірність заразитися за час dt , тобто за $r \cdot dt$ контактів?

Відмітимо, що

$rdt \frac{I}{N}$ - число контактів з хворими за час dt ,

$(1-c)$ - ймовірність не заразитися при контакті з хворим,

$(1-c)^{rdt \frac{I}{N}}$ - ймовірність не заразитися при $rdt \frac{I}{N}$ контактах,

$1 - (1-c)^{rdt \frac{I}{N}}$ - ймовірність заразитися при $rdt \frac{I}{N}$ контактах,

$$1 - (1-c)^{rdt \frac{I}{N}} = 1 - \exp\left(\frac{r \log(1-c)I}{N} dt\right) \approx 1 - \left(1 + \frac{r \log(1-c)I}{N} dt\right) = -\frac{r \log(1-c)I}{N} dt.$$

Таким чином інтенсивність зараження (ймовірність зараження в одиницю часу)

$$\text{дорівнює } p(I) = -\frac{rI \log(1-c)}{N}.$$

Теоретичні відомості

Модель 3. Модель зі зворотним зв'язком

В моделі зі зворотним зв'язком керовані параметри (цими параметрами можуть бути u, r, c) залежать від поточного стану системи (S, I, R, D) . Наприклад, інтенсивність контактів r може бути залежною від кількості інфікованих (або померлих), $r = r(I) = r_0 / (I/N)^\gamma$ або $r = r(I) = r_0 e^{-\gamma I/N}$, $\gamma \geq 0$; інтенсивність вакцинації (у випадку наявності вакцини) також може залежати від кількості інфікованих (або померлих), $u = u(I) = \min\{u_{\max}, u_0 (I/N)^\delta\}$, $\delta \geq 0$; ймовірність підхопити інфекцію c також може бути керованою величиною (залежить від правил соціального дистанціювання та строгості індивідуальних захисних заходів), $c = c(I) = c_0 / (I/N)^\lambda$, $\lambda \geq 0$. Пошук цих та інших функціональних форм $r(I)$, $u(I)$, $c(I)$ є важливою проблемою керування епідемією.

Рекомендації щодо вибору параметрів моделі r, c, q, q_0, u, N, T .

Бажано орієнтуватися на епідемічну ситуацію з короно-вірусом COVID-19,

Наприклад,

$r \in [0.001, 50]$, (дослідити, як радикальне зменшення числа контактів впливає на хід епідемії)

$c \in [0.5, 0.9]$, (у COVID-19 дуже висока ймовірність передачі)

$q \in [0.05, 0.1]$, (у COVID-19 досить повільний процес одужання, $1/q$ - середній час одужання)

$q_0 \in [0.01, 0.1]$, (відсоток смертельних випадків коливається від 1% до 10%)

$u \in [0, 0.01]$, (для COVID-19 практично ще немає вакцини)

$N \in [10^3, 10^6]$, (як виглядає епідемічний процес для невеликих міст і для міст мільйонників)

$T \in [30, 300]$ (моделювання потрібно вести до моменту закінчення епідемії).

Порядок виконання роботи

Завдання.

Промодельовати розвиток епідемії, розв'язав задачу Коши для системи диференціальних рівнянь для різних наборів параметрів моделі

$(N, r, c, q, u(t) = \text{const})$ та керувань $u(t) \in U = \{u : 0 \leq u \leq u_{\max} < 1\}$

В тому числі зі зворотним зв'язком $(u(I/N), r(I/N), c(I/N))$.

Порядок виконання роботи.

- 1) Обрати модель M1 – M5.
- 2) Обрати значення параметрів моделі $(N, r, c, q, q_0, \alpha, \beta, u_{\max})$ відповідно їх змісту (орієнтуватися на пандемію COVID-19).
- 3) Обрати початкові значення, наприклад, $I(0) > 0$, $S(0) = N - I(0)$, $J(0) = 0$, $R(0) = 0$, $D(0) = 0$.
- 4) Обрати проміжок часу $[0, T]$.
- 5) Обрати програму для розв'язання нелінійної системи диференціальних рівнянь (solver, наприклад, ode23 в системі Matlab, або відповідну програму в інших системах програмування).
- 6) Розв'язати задачу Коши на відрізку часу $[0, T]$.
- 7) Візуалізувати розв'язок, тобто побудувати графіки функцій $S(t), I(t), R(t), D(t)$. На графіках вказати назву графіку, назву осей координат, легенду, значення основних параметрів моделі.
- 8) Обчислити основні характеристики епідемічного процесу для обраного набору параметрів моделі (пік процесу - $I_{\max} = \max_{t \in [0, T]} I(t)$, $t_{\max} = \arg \max_{t \in [0, T]} I(t)$; тривалість епідемії; кінцевий результат епідемії, $S(\infty), I(\infty), J(\infty), R(\infty), D(\infty)$).
- 9) Дослідити зміни розвитку епідемії (картину) для різних значень параметрів моделі $(N, r, c, q, q_0, u(t) = \text{const})$.
- 10) Дослідити зміни розвитку епідемії (картину) для різних керувань $u(t) \in U = \{u : 0 \leq u \leq u_{\max} < 1\}$, в тому числі зі зворотним зв'язком $u(I/N), r(I/N), c(I/N)$.
- 11) Підготувати звіт про роботу в електронному вигляді (з графіками, висновками і лістингом програми).
- 12) Надіслати звіт викладачеві на електронну адресу.

Основна частина

У цьому дослідженні для моделювання епідемії використано модель №3, яка базується на зворотньому зв'язку. Параметри моделі, що регулюються, залежать від поточного стану системи. Для аналізу процесу поширення епідемії була створена система диференціальних рівнянь, які були розв'язані за допомогою методу Рунге-Кутта 4 порядку.

Змодельовано епідемію на наступних даних:

$N = 1\,000\,000$	$R = 0$	$C_0 = 0.5$	$\alpha = 0.1$
$S = 900\,000$	$D = 0$	$R_0 = 2$	$\beta = 0.05$
$I = 100\,000$		$U_0 = 0.1$	$T = [0, 100]$

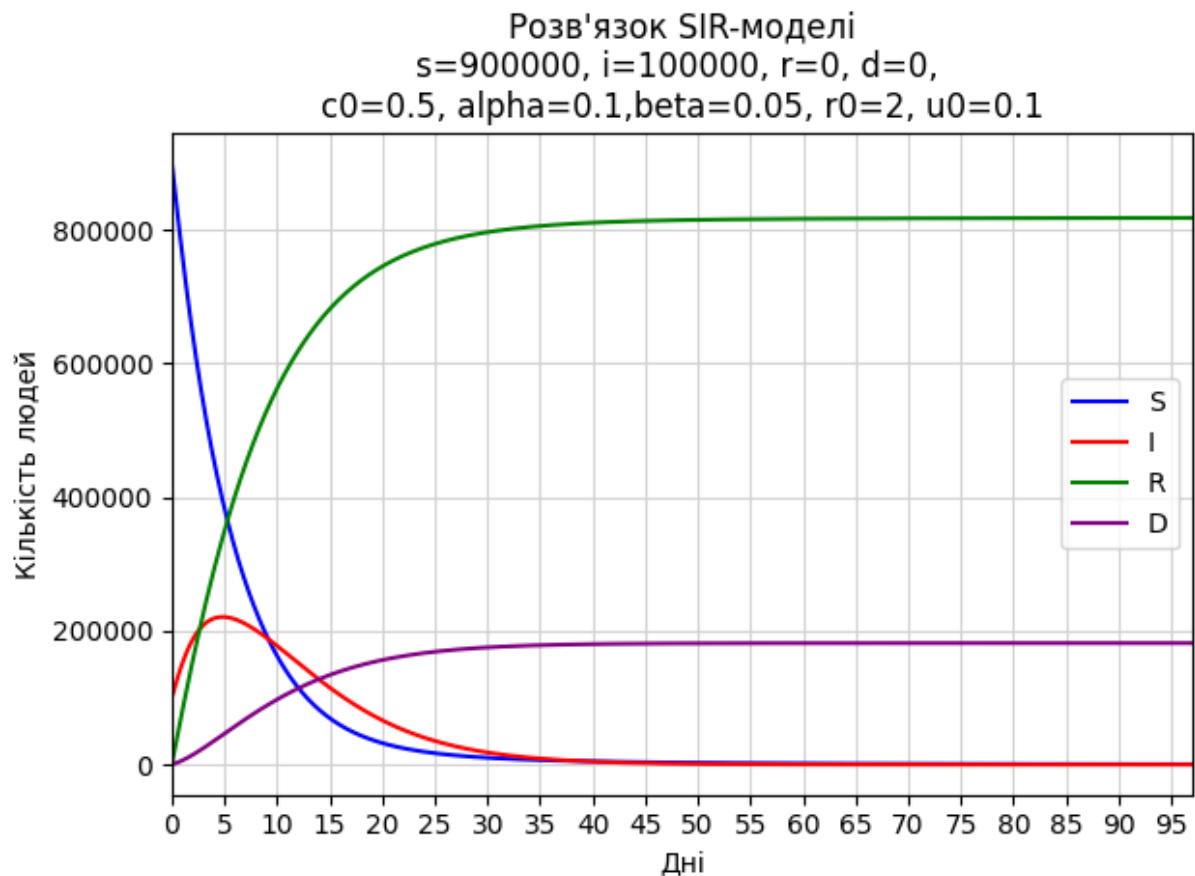


Рис. 1 – Протікання епідемії

Епідемія мала такі характеристики:

Пік інфікованих = 220833 , цей пік відбувся на 5 день

Результати епідемії:

Кількість здорових індивідів: 446

Кількість хворих і заразних: 0

Кількість індивідів із імунітетом: 817818

Число померлих: 181734

Епідемія тривала 96 днів

З цього можна зробити висновок, що за такого набору параметрів перехворіли або отримали імунітет майже всі люди. Трохи більше 181 тис. індивідів померло. Що приблизно рівне 18.1% від усього населення. Оскільки дослідження проводились при s (ймовірність зараження при контакті з хворим) = 0.5, то при збільшенні цього коефіцієнту кількість померлих би збільшилась.

Дослідження параметру N (Загальне число індивідів)

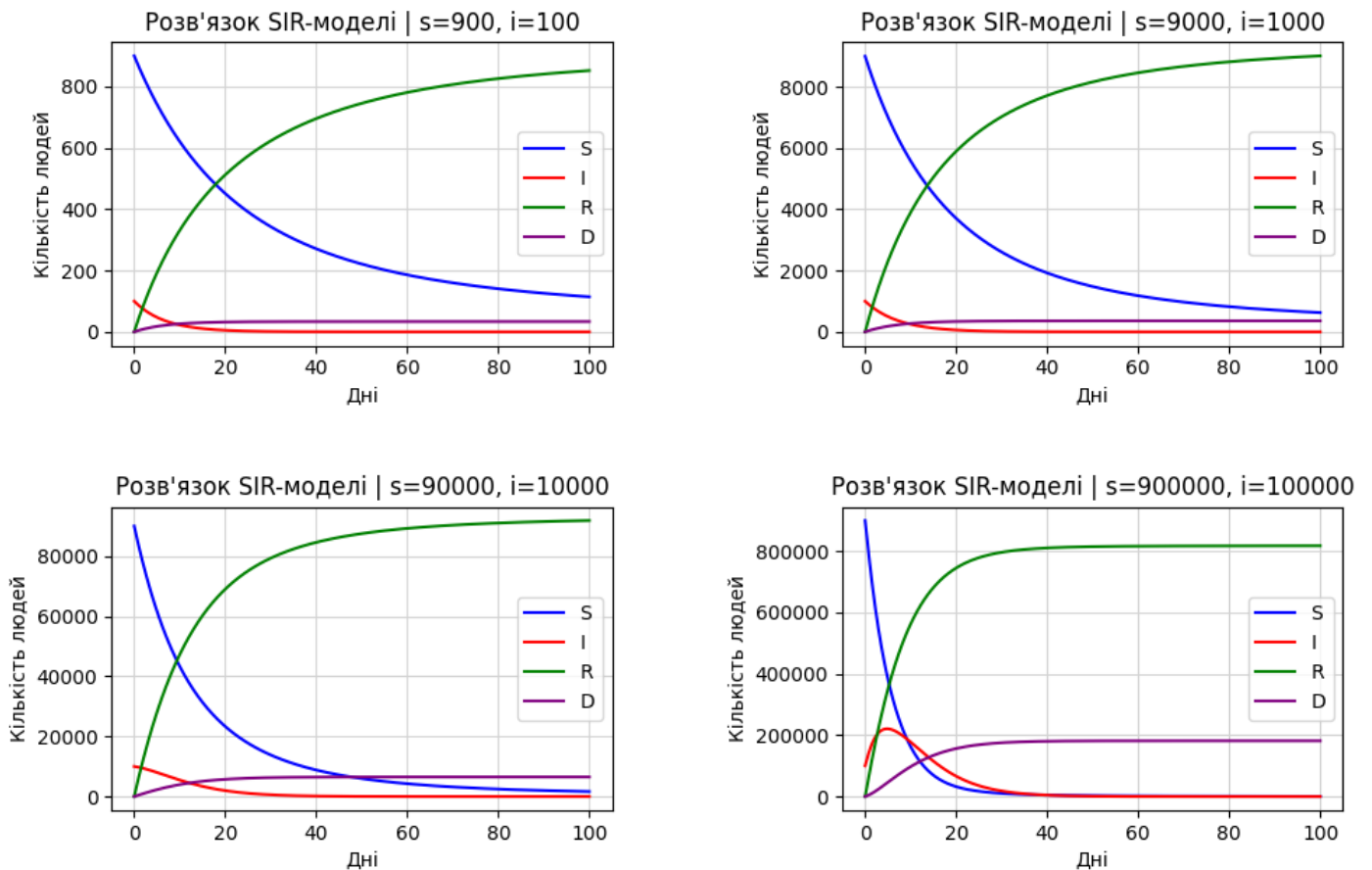


Рис. 2 – Протікання епідемії при $N = [1\ 000, 10\ 000, 100\ 000, 1\ 000\ 000]$

Для цього дослідження було використано такі ж параметри як і в попередньому:

$$\begin{array}{lll}
 R = 0 & C_0 = 0.5 & \alpha = 0.1 \\
 D = 0 & R_0 = 2 & \beta = 0.05 \\
 & U_0 = 0.1 & T = [0, 100]
 \end{array}$$

Де $\alpha \cdot dt$ - ймовірність одужання інфікованого за час dt ;

$\beta \cdot dt$ - ймовірність смерті інфікованого за час dt ;

При розмірах населення $N=1\ 000$ та $N=10\ 000$ відзначається невелика кількість заражених / хворих і мала смертність. Проте цього не можна сказати про випадки, коли $N=100\ 000$ та $N=1\ 000\ 000$. За таких умов кількість заражених значно зростає. Навіть за невисокого коефіцієнта смертності (0.05), при $N=1\ 000\ 000$ кількість померлих становить трохи менше за п'яту частину всього населення.

Дослідження параметру r (Інтенсивність контактів індивіда)

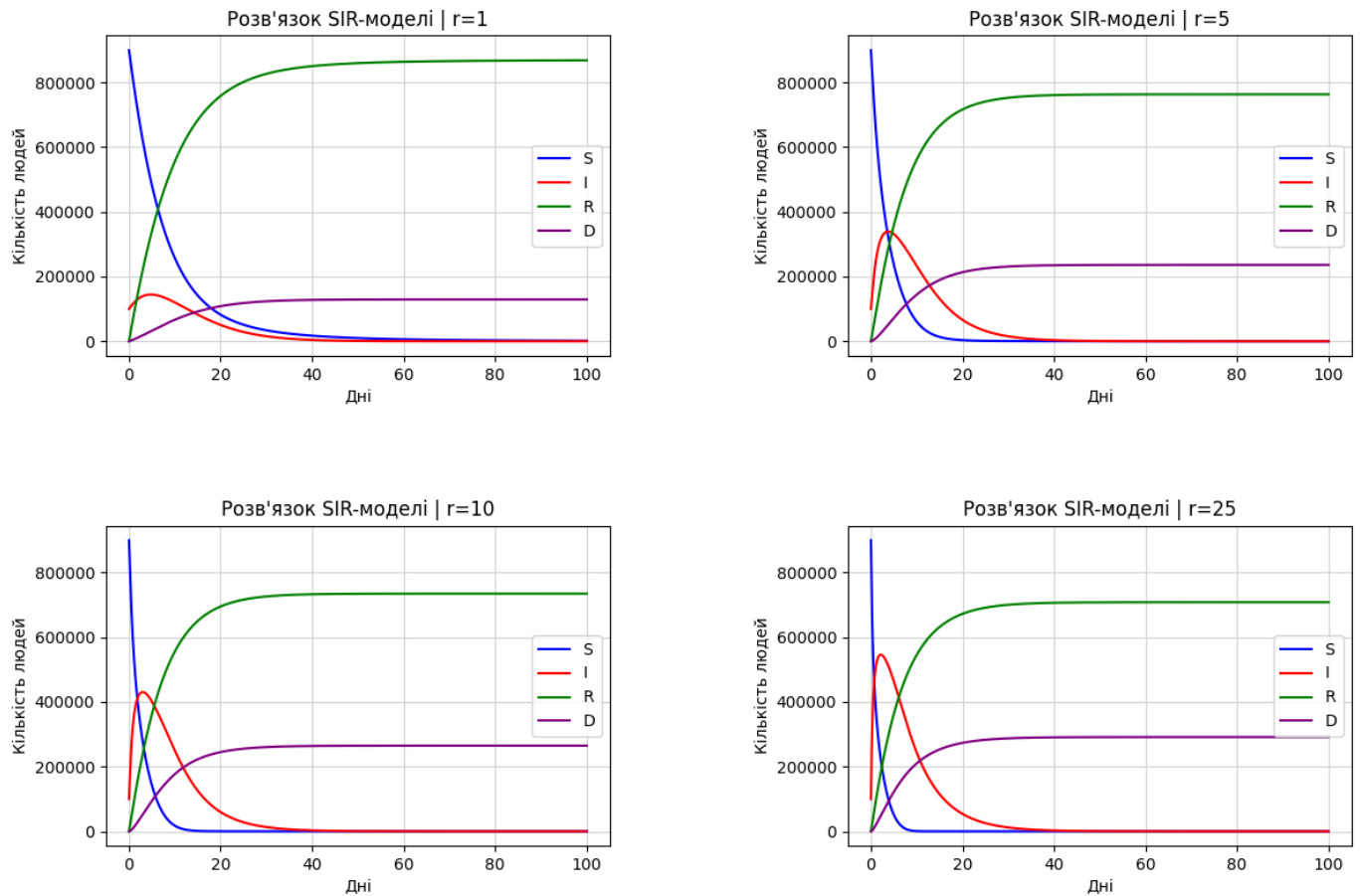


Рис. 3 – Протікання епідемії при $r_0 = [1, 5, 10, 25]$

Отримані висновки виявились досить очевидними: кількість контактів особи має значний вплив на швидкість поширення інфекції серед населення і тривалість епідемії. Зі збільшенням кількості контактів інфекція поширюється швидше, що може призвести до набуття імунітету або смерті.

Дослідження параметру c (Ймовірність зараження при контакті з хворим)

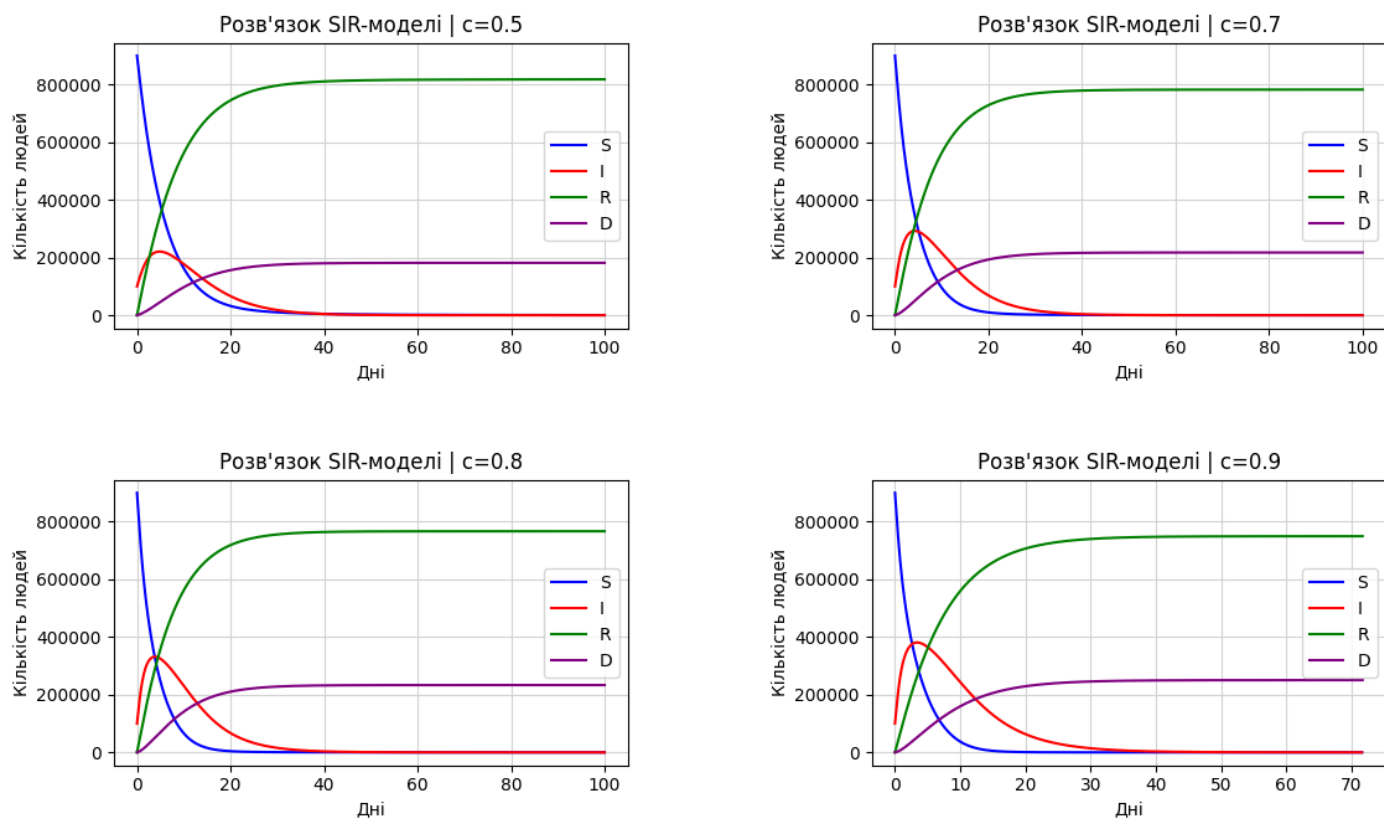


Рис. 4 – Протікання епідемії при $c_0 = [0.5, 0.7, 0.8, 0.9]$

Зміна цього параметра також впливає на швидкість поширення інфекції серед населення, час досягнення піку епідемії і тривалість її перебігу. Проте, за визначених умов моделі, зміна цього параметра має обмежений вплив на загальну епідеміологічну ситуацію.

Дослідження параметру u (Відсоток вакцинованих за час)

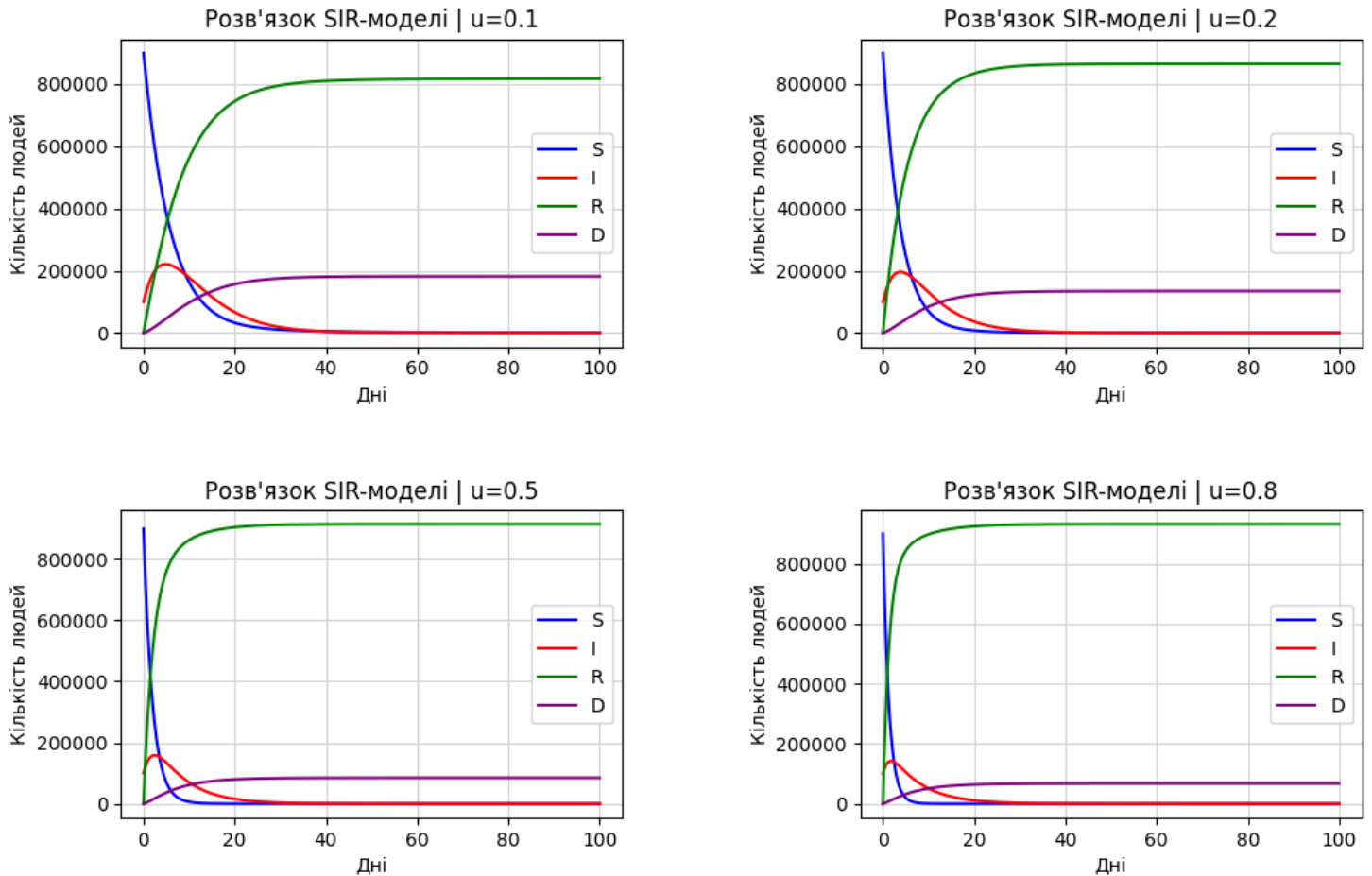


Рис. 5 – Протікання епідемії при $u_0 = [0.1, 0.2, 0.5, 0.8]$

З цього дослідження видно, як впливає даний регулюючий параметр на епідемічну обстановку. Зокрема, при збільшенні значення параметра " u " збільшується кількість осіб, що отримують імунітет, і спостерігається прискорене завершення епідемії. Це очікувано, оскільки зі зростанням цього коефіцієнта більше людей вакцинується проти хвороби.

Висновок

Під час проведення даної лабораторної роботи було ретельно проаналізовано основні принципи формування математичних моделей епідемій, що виявилось ключовим для розуміння динаміки поширення захворювань.

Отримані результати підтвердили, що збільшення чисельності населення призводить до швидшого розповсюдження інфекції та збільшення смертності.

Збільшення кількості контактів підвищує ризик зараження, хоча це не завжди впливає на смертність.

Варто зазначити, що підвищення відсотку вакцинованих суттєво знижує ризик захворювання та тривалість епідемії.

Загальний висновок полягає в необхідності вжиття ефективних заходів контролю та профілактики для ефективного управління епідемічною ситуацією.

Використана література

1. Методичні вказівки до лабораторної роботи.
2. A. Huppert, G. Katriel, Mathematical modelling and prediction in infectious disease epidemiology, Clin. Microbiol. Infection 19 (2013), 999-1005.

Додаток А – Код програми

КМ-03 Ковальчук Лаб 2.py:

```
import numpy as np
import matplotlib.pyplot as plt

def u(_i, _u0):
    return min(umax, _u0*((_i/N)**delta))

def p(_i, _r0, _c0):
    _r = _r0*np.exp(-gamma*_i/N)
    _c = _c0/((_i/N)**lamda)
    _p = -_r*_i*np.log(1-_c)/N
    return _p

def st(_s, _i, _r0, _c0, _u0):
    return -_s*p(_i, _r0, _c0) - _s*u(_i, _u0)

def it(_s, _i, _r0, _c0):
    return _s*p(_i, _r0, _c0) - alpha*_i - beta*_i

def rt(_s, _i, _u0):
    return alpha*_i + _s*u(_i, _u0)

def dt(_i):
    return beta*_i

def rk4(_s, _i, _r, _d, _h, _r0, _c0, _u0):
    k1 = _h * st(_s, _i, _r0, _c0, _u0)
    q1 = _h * it(_s, _i, _r0, _c0)
    l1 = _h * rt(_s, _i, _u0)
```

```

m1 = _h * dt(_i)
k2 = _h * st(_s, _i, _r0, _c0, _u0)
q2 = _h * it(_s, _i, _r0, _c0)
l2 = _h * rt(_s, _i, _u0)
m2 = _h * dt(_i)
k3 = _h * st(_s, _i, _r0, _c0, _u0)
q3 = _h * it(_s, _i, _r0, _c0)
l3 = _h * rt(_s, _i, _u0)
m3 = _h * dt(_i)
k4 = _h * st(_s, _i, _r0, _c0, _u0)
q4 = _h * it(_s, _i, _r0, _c0)
l4 = _h * rt(_s, _i, _u0)
m4 = _h * dt(_i)
s_next = _s + (k1 + 2 * k2 + 2 * k3 + k4) / 6
i_next = _i + (q1 + 2 * q2 + 2 * q3 + q4) / 6
r_next = _r + (l1 + 2 * l2 + 2 * l3 + l4) / 6
d_next = _d + (m1 + 2 * m2 + 2 * m3 + m4) / 6
return s_next, i_next, r_next, d_next

def test_n():
    N = [1_000, 10_000, 100_000, 1_000_000]
    s = [900, 9_000, 90_000, 900_000]
    i = [100, 1_000, 10_000, 100_000]
    s_res = []
    i_res = []
    r_res = []
    d_res = []
    for k in range(len(N)):
        s_list = [s[k]]
        i_list = [i[k]]
        r_list = [r]
        d_list = [d]
        for j in range(step):
            res1, res2, res3, res4 = rk4(s_list[j], i_list[j], r_list[j],
d_list[j], h, r0, c0, u0)
            s_list.append(res1)
            i_list.append(res2)
            r_list.append(res3)
            d_list.append(res4)

        s_list = np.array(s_list)
        i_list = np.array(i_list)
        r_list = np.array(r_list)
        d_list = np.array(d_list)
        s_res.append(s_list)

```

```

        i_res.append(i_list)
        r_res.append(r_list)
        d_res.append(d_list)

x = np.arange(t0, t + h, h)
plt.subplot(2, 2, 1)
plt.title('Розв\'язок SIR-моделі | s={}, i={}'.format(s[0], i[0]))
plt.plot(x, s_res[0], color='blue', label="S")
plt.plot(x, i_res[0], color='red', label="I")
plt.plot(x, r_res[0], color='green', label="R")
plt.plot(x, d_res[0], color='purple', label="D")
plt.legend()
plt.grid(c='lightgrey')
ax = plt.gca()
ax.set_axisbelow(True)
plt.xlabel('Дні')
plt.ylabel('Кількість людей')

plt.subplot(2, 2, 2)
plt.title('Розв\'язок SIR-моделі | s={}, i={}'.format(s[1], i[1]))
plt.plot(x, s_res[1], color='blue', label="S")
plt.plot(x, i_res[1], color='red', label="I")
plt.plot(x, r_res[1], color='green', label="R")
plt.plot(x, d_res[1], color='purple', label="D")
plt.legend()
plt.grid(c='lightgrey')
ax = plt.gca()
ax.set_axisbelow(True)
plt.xlabel('Дні')
plt.ylabel('Кількість людей')

plt.subplot(2, 2, 3)
plt.title('Розв\'язок SIR-моделі | s={}, i={}'.format(s[2], i[2]))
plt.plot(x, s_res[2], color='blue', label="S")
plt.plot(x, i_res[2], color='red', label="I")
plt.plot(x, r_res[2], color='green', label="R")
plt.plot(x, d_res[2], color='purple', label="D")
plt.legend()
plt.grid(c='lightgrey')
ax = plt.gca()
ax.set_axisbelow(True)
plt.xlabel('Дні')
plt.ylabel('Кількість людей')

```



```

plt.subplot(2, 2, 4)
plt.title('Розв\'язок SIR-моделі | s={}, i={}'.format(s[3], i[3]))
plt.plot(x, s_res[3], color='blue', label="S")
plt.plot(x, i_res[3], color='red', label="I")
plt.plot(x, r_res[3], color='green', label="R")
plt.plot(x, d_res[3], color='purple', label="D")
plt.legend()
plt.grid(c='lightgrey')
ax = plt.gca()
ax.set_axisbelow(True)
plt.xlabel('Дні')
plt.ylabel('Кількість людей')
plt.tight_layout()
plt.show()

def test_r():
    r_test = [1, 5, 10, 25]
    s_res = []
    i_res = []
    r_res = []
    d_res = []
    for k in range(len(r_test)):
        s_list = [s]
        i_list = [i]
        r_list = [r]
        d_list = [d]
        for j in range(step):
            res1, res2, res3, res4 = rk4(s_list[j], i_list[j], r_list[j],
d_list[j], h, r_test[k], c0, u0)
            s_list.append(res1)
            i_list.append(res2)
            r_list.append(res3)
            d_list.append(res4)

        s_list = np.array(s_list)
        i_list = np.array(i_list)
        r_list = np.array(r_list)
        d_list = np.array(d_list)
        s_res.append(s_list)
        i_res.append(i_list)
        r_res.append(r_list)
        d_res.append(d_list)

    x = np.arange(t0, t + h, h)
    plt.subplot(2, 2, 1)

```

```

plt.title('Розв\'язок SIR-моделі | r={}'.format(r_test[0]))
plt.plot(x, s_res[0], color='blue', label="S")
plt.plot(x, i_res[0], color='red', label="I")
plt.plot(x, r_res[0], color='green', label="R")
plt.plot(x, d_res[0], color='purple', label="D")
plt.legend()
plt.grid(c='lightgrey')
ax = plt.gca()
ax.set_axisbelow(True)
plt.xlabel('Дні')
plt.ylabel('Кількість людей')
plt.tight_layout()

plt.subplot(2, 2, 2)
plt.title('Розв\'язок SIR-моделі | r={}'.format(r_test[1]))
plt.plot(x, s_res[1], color='blue', label="S")
plt.plot(x, i_res[1], color='red', label="I")
plt.plot(x, r_res[1], color='green', label="R")
plt.plot(x, d_res[1], color='purple', label="D")
plt.legend()
plt.grid(c='lightgrey')
ax = plt.gca()
ax.set_axisbelow(True)
plt.xlabel('Дні')
plt.ylabel('Кількість людей')
plt.tight_layout()

plt.subplot(2, 2, 3)
plt.title('Розв\'язок SIR-моделі | r={}'.format(r_test[2]))
plt.plot(x, s_res[2], color='blue', label="S")
plt.plot(x, i_res[2], color='red', label="I")
plt.plot(x, r_res[2], color='green', label="R")
plt.plot(x, d_res[2], color='purple', label="D")
plt.legend()
plt.grid(c='lightgrey')
ax = plt.gca()
ax.set_axisbelow(True)
plt.xlabel('Дні')
plt.ylabel('Кількість людей')
plt.tight_layout()

plt.subplot(2, 2, 4)
plt.title('Розв\'язок SIR-моделі | r={}'.format(r_test[3]))
plt.plot(x, s_res[3], color='blue', label="S")

```

```

plt.plot(x, i_res[3], color='red', label="I")
plt.plot(x, r_res[3], color='green', label="R")
plt.plot(x, d_res[3], color='purple', label="D")
plt.legend()
plt.grid(c='lightgrey')
ax = plt.gca()
ax.set_axisbelow(True)
plt.xlabel('Дні')
plt.ylabel('Кількість людей')
plt.tight_layout()
plt.show()

def test_c():
    c_test = [0.5, 0.7, 0.8, 0.9]
    s_res = []
    i_res = []
    r_res = []
    d_res = []
    for k in range(len(c_test)):
        s_list = [s]
        i_list = [i]
        r_list = [r]
        d_list = [d]
        for j in range(step):
            res1, res2, res3, res4 = rk4(s_list[j], i_list[j], r_list[j],
d_list[j], h, r0, c_test[k], u0)
            s_list.append(res1)
            i_list.append(res2)
            r_list.append(res3)
            d_list.append(res4)

        s_list = np.array(s_list)
        i_list = np.array(i_list)
        r_list = np.array(r_list)
        d_list = np.array(d_list)
        s_res.append(s_list)
        i_res.append(i_list)
        r_res.append(r_list)
        d_res.append(d_list)

    x = np.arange(t0, t + h, h)
    plt.subplot(2, 2, 1)
    plt.title('Розв\'язок SIR-моделі | c={}'.format(c_test[0]))
    plt.plot(x, s_res[0], color='blue', label="S")
    plt.plot(x, i_res[0], color='red', label="I")

```

```

plt.plot(x, r_res[0], color='green', label="R")
plt.plot(x, d_res[0], color='purple', label="D")
plt.legend()
plt.grid(c='lightgrey')
ax = plt.gca()
ax.set_axisbelow(True)
plt.xlabel('Дні')
plt.ylabel('Кількість людей')
plt.tight_layout()

plt.subplot(2, 2, 2)
plt.title('Розв\`язок SIR-моделі | c={}'.format(c_test[1]))
plt.plot(x, s_res[1], color='blue', label="S")
plt.plot(x, i_res[1], color='red', label="I")
plt.plot(x, r_res[1], color='green', label="R")
plt.plot(x, d_res[1], color='purple', label="D")
plt.legend()
plt.grid(c='lightgrey')
ax = plt.gca()
ax.set_axisbelow(True)
plt.xlabel('Дні')
plt.ylabel('Кількість людей')
plt.tight_layout()

plt.subplot(2, 2, 3)
plt.title('Розв\`язок SIR-моделі | c={}'.format(c_test[2]))
plt.plot(x, s_res[2], color='blue', label="S")
plt.plot(x, i_res[2], color='red', label="I")
plt.plot(x, r_res[2], color='green', label="R")
plt.plot(x, d_res[2], color='purple', label="D")
plt.legend()
plt.grid(c='lightgrey')
ax = plt.gca()
ax.set_axisbelow(True)
plt.xlabel('Дні')
plt.ylabel('Кількість людей')
plt.tight_layout()

plt.subplot(2, 2, 4)
plt.title('Розв\`язок SIR-моделі | c={}'.format(c_test[3]))
plt.plot(x, s_res[3], color='blue', label="S")
plt.plot(x, i_res[3], color='red', label="I")
plt.plot(x, r_res[3], color='green', label="R")
plt.plot(x, d_res[3], color='purple', label="D")

```

```

plt.legend()
plt.grid(c='lightgrey')
ax = plt.gca()
ax.set_axisbelow(True)
plt.xlabel('Дні')
plt.ylabel('Кількість людей')
plt.tight_layout()
plt.show()

def test_u():
    u_test = [0.1, 0.2, 0.5, 0.8]
    s_res = []
    i_res = []
    r_res = []
    d_res = []
    for k in range(len(u_test)):
        s_list = [s]
        i_list = [i]
        r_list = [r]
        d_list = [d]
        for j in range(step):
            res1, res2, res3, res4 = rk4(s_list[j], i_list[j], r_list[j],
d_list[j], h, r0, c0, u_test[k])
            s_list.append(res1)
            i_list.append(res2)
            r_list.append(res3)
            d_list.append(res4)

        s_list = np.array(s_list)
        i_list = np.array(i_list)
        r_list = np.array(r_list)
        d_list = np.array(d_list)
        s_res.append(s_list)
        i_res.append(i_list)
        r_res.append(r_list)
        d_res.append(d_list)

    x = np.arange(t0, t + h, h)
    plt.subplot(2, 2, 1)
    plt.title('Розв'язок SIR-моделі | u={}'.format(u_test[0]))
    plt.plot(x, s_res[0], color='blue', label="S")
    plt.plot(x, i_res[0], color='red', label="I")
    plt.plot(x, r_res[0], color='green', label="R")

```

```

plt.plot(x, d_res[0], color='purple', label="D")
plt.legend()
plt.grid(c='lightgrey')
ax = plt.gca()
ax.set_axisbelow(True)
plt.xlabel('Дні')
plt.ylabel('Кількість людей')
plt.tight_layout()

plt.subplot(2, 2, 2)
plt.title('Розв\'язок SIR-моделі | u={}'.format(u_test[1]))
plt.plot(x, s_res[1], color='blue', label="S")
plt.plot(x, i_res[1], color='red', label="I")
plt.plot(x, r_res[1], color='green', label="R")
plt.plot(x, d_res[1], color='purple', label="D")
plt.legend()
plt.grid(c='lightgrey')
ax = plt.gca()
ax.set_axisbelow(True)
plt.xlabel('Дні')
plt.ylabel('Кількість людей')
plt.tight_layout()

plt.subplot(2, 2, 3)
plt.title('Розв\'язок SIR-моделі | u={}'.format(u_test[2]))
plt.plot(x, s_res[2], color='blue', label="S")
plt.plot(x, i_res[2], color='red', label="I")
plt.plot(x, r_res[2], color='green', label="R")
plt.plot(x, d_res[2], color='purple', label="D")
plt.legend()
plt.grid(c='lightgrey')
ax = plt.gca()
ax.set_axisbelow(True)
plt.xlabel('Дні')
plt.ylabel('Кількість людей')
plt.tight_layout()

plt.subplot(2, 2, 4)
plt.title('Розв\'язок SIR-моделі | u={}'.format(u_test[3]))
plt.plot(x, s_res[3], color='blue', label="S")
plt.plot(x, i_res[3], color='red', label="I")
plt.plot(x, r_res[3], color='green', label="R")

```

```

plt.plot(x, d_res[3], color='purple', label="D")
plt.legend()
plt.grid(c='lightgrey')
ax = plt.gca()
ax.set_axisbelow(True)
plt.xlabel('Дні')
plt.ylabel('Кількість людей')
plt.tight_layout()
plt.show()

def first():
    s_list = [s]
    i_list = [i]
    r_list = [r]
    d_list = [d]

    for j in range(step):
        res1, res2, res3, res4 = rk4(s_list[j], i_list[j], r_list[j], d_list[j], h,
r0, c0, u0)
        s_list.append(res1)
        i_list.append(res2)
        r_list.append(res3)
        d_list.append(res4)

    s_list = np.array(s_list)
    i_list = np.array(i_list)
    r_list = np.array(r_list)
    d_list = np.array(d_list)
    max_infected = max(i_list)
    max_time = t0 + h*(np.where(i_list == max_infected)[0])

    print('Пік інфікованих = ', int(max_infected), ', цей пік відбувся на ',
int(round(max_time[0], 0)), 'день')
    end_time = t0 + h*np.where(i_list < 1)[0][0]

    print("Результати епідемії: ")
    print("Кількість здорових індивідів: ", int(s_list[-1]))
    print("Кількість хворих і заразних: ", int(i_list[-1]))
    print("Кількість індивідів із імунітетом: ", int(r_list[-1]))
    print("Число померлих: ", int(d_list[-1]))
    print(f"Епідемія тривала {int(end_time)} днів")

x = np.arange(t0, t+h, h)

```

```

plt.title('Розв\'язок SIR-моделі\ncs={}, i={}, r={}, d={}, \nc0={}, alpha={}, '
          'beta={}, r0={}, u0={}'.format(s, i, r, d, c0, alpha, beta, r0, u0))
plt.plot(x, s_list, color='blue', label="S")
plt.plot(x, i_list, color='red', label="I")
plt.plot(x, r_list, color='green', label="R")
plt.plot(x, d_list, color='purple', label="D")
plt.legend()
plt.grid(c='lightgrey')
ax = plt.gca()
ax.set_axisbelow(True)
plt.xlabel('Дні')
plt.ylabel('Кількість людей')
plt.xlim(0, int(end_time)+1)
plt.xticks(np.arange(0, t, 5))
plt.tight_layout()
plt.show()

if __name__ == '__main__':
    N = 1_000_000
    s = 900_000
    i = 100_000
    r = 0
    d = 0
    c0 = 0.5
    r0 = 2
    u0 = 0.1
    alpha = 0.1
    beta = 0.05
    t0, t = 0, 100

    h = 0.1
    step = int((t-t0)/h)

    gamma = 6
    delta = 0.1
    lamda = 0.01
    umax = 0.9

    first()

    test_n()

    test_r()

    test_c()

```



```
test_u()
```