УДК 519.6 *http://dx.doi.org/10.30970/vam.2024.34.00000*

КЛІЄНТ-СЕРВЕРНИЙ ІНСТРУМЕНТ ДЛЯ ПОБУДОВИ ТА ОПТИМАЛЬНОГО КЕРУВАННЯ КОМПАРТМЕНТНИМИ МОДЕЛЯМИ

Ю. Гнитка, Я. Борисюк, М. Щербатий

Львівський національний університет імені Івана Франка,
вул. Університетська 1, Львів, 79000, Україна
e-mail: yurahnytka3@gmail.com,
yaryna.borysyuk@lnu.edu.ua, mykhaylo.shcherbatyy@lnu.edu.ua

У статті представлено CompLab – клієнт-серверний (Angular / Python + Flask) програмний комплекс для повного циклу роботи з компартментними моделями: від графічної побудови структури до розв'язування прямих задач моделювання та задач оптимального керування. Користувач задає модель у вигляді орієнтованого зваженого графа; на основі цього графа сервер автоматично формує систему звичайних диференціальних рівнянь (з використанням `SymPy` для розбору алгебраїчних виразів потоків), а потім або інтегрує цю систему методом LSODA (реалізація бібліотеки `SciPy` – `solve_ivp`, `method='LSODA'`, `dense_output=True`), або розв'язує задачу оптимального керування за принципом максимуму Понтрягіна з прямим-зворотним проходом. Для оптимізації автоматично будуються функція Гамільтона та спряжена система; оновлення керувань виконується з проєкцією на допустимі межі та релаксацією, а у випадку лінійності за керуваннями застосовується bang-bang схема. Реалізація використовує неперервні інтерполяційні функції розв'язків (`dense_output`) замість фіксованих сіток, що спрощує поєднання прямого і зворотного проходів і забезпечує гнучке відображення результатів з довільним кроком дискретизації. Інструмент має інтерактивний редактор графів (на основі `Cytoscape.js`) з автопозиціюванням, робочі простори, імпорт/експорт моделей і результатів, перевикористання попередніх запусків, вивід проміжних об'єктів (системи рівнянь, Гамільтоніан, спряжені рівняння) тощо. Поточна версія підтримує закриті моделі з часово-інваріантною динамікою та сталими параметрами. Можливості CompLab продемонстровано на базовій SEIR-моделі та двох її модифікаціях з одним і двома керуваннями.

Ключові слова: математичне моделювання, компартментні моделі, оптимальне керування, програмне забезпечення, принцип максимуму Понтрягіна.

1. Вступ

Математичне моделювання є потужним інструментом для дослідження динаміки складних систем та пошуку оптимальних стратегій втручання (керування), зокрема в епідеміології [1–3]. При цьому, робота з математичними моделями, зокрема з такими, які можна привести до компартментних [4], залишається технічно складною, трудомісткою та часто монотонною. Спеціалізоване програмне забезпечення, яке автоматизує певні частини цього процесу, могло б суттєво спростити задачу для дослідників, надаючи можливість більше зосередитися на інтерпретації результатів, а не на технічних нюансах їх отримання. Більше того, подібне програмне забезпечення може знизити поріг входу для дослідників з інших галузей, а також для початківців. Актуальність також обумовлена тим, що епідеміологічні виклики, як-от пандемія COVID-19, продовжують залишатися невід'ємною частиною сучасності і можуть призводити до значних кризових ситуацій [5]. Водночас компартментний

підхід широко застосовується й добре зарекомендував себе при моделюванні поширення епідемій. Відповідно, наявність програмного забезпечення для ефективнішої роботи з подібними системами допоможе завчасно підготуватися до потенційних епідемічних загроз і оперативно реагувати на них.

2. Компартментне моделювання

У даному розділі розглянуто побудову моделей динамічних систем на основі компартментного підходу. Сформульовано задачі аналізу та оптимального керування динамічних систем, які досліджуються за допомогою розробленого програмного продукту.

Динаміка складних систем часто описується системою диференціальних рівнянь, адже в таких системах швидкість зміни значення однієї змінної часто залежить від інших змінних цієї ж системи. Такий підхід дозволяє точно описати складні взаємозалежності системи, проте ускладнює їх розуміння та інтерпретацію отриманих результатів без належної візуалізації. Компартментні моделі слугують візуальним інструментом для більш інтуїтивного представлення диференціальних рівнянь динамічних систем.

Компартмент – абстрактна частина системи, у межах якої досліджувана величина (маса речовини, чисельність популяції, енергія тощо) вважається просторово однорідною; внутрішні відмінності ігноруються. Кожній змінній стану системи відповідає один компартмент (умовно зображується прямокутником).

Обмін між компартментами здійснюється шляхом потоків – швидкостей переносу величини від одного компартменту до іншого. Потоки описуються рівняннями швидкості переходу і зазвичай моделюються системами диференціальних рівнянь першого порядку. Сума потоків, що входять або виходять з компартменту, відповідає зміні відповідної змінної системи в часі.

Структурно модель може бути подана як орієнтований зважений граф: вершини – компартменти, ребра – потоки. Це подання спрощує візуалізацію та структурний аналіз.

Тобто розглядаючи компартменти як змінні станів $y_i(t), i = 1, 2, \ldots, n$, компартментна модель, яка складається з n компартментів, є графічним відображенням системи звичайних диференціальних рівнянь:

$$\frac{dy(t)}{dt} = f(y(t), u(t), \omega(t), \theta(t), t), \tag{1}$$

$$y(0) = y_0, \tag{2}$$

де:

- $y(t) = (y_1(t), y_2(t), \ldots, y_n(t)) \in \mathbb{R}^n$ – n-вимірний вектор стану системи у момент часу $t \in [0, t_f] = \Omega_t \subset \mathbb{R}$;
- $f = (f_1, f_2, \ldots, f_n)$ – функція динаміки системи, яка відповідає ребрам графу компартментної моделі;
- $u(t) = (u_1(t), u_2(t), \ldots, u_m(t))$ – вектор-функція керування системою;
- $\omega(t) = (\omega_1(t), \omega_2(t), \ldots, \omega_l(t))$ – вектор зовнішніх впливів;
- $\theta(t) = (\theta_1(t), \theta_2(t), \ldots, \theta_g(t))$ – вектор параметрів моделі;
- $y_0 = (y_{1_0}, y_{2_0}, \ldots, y_{n_0}) \in \mathbb{R}^n$ – вектор початкових умов (значень) відповідних змінних моделі.

Враховуючи вищевикладене, можна описати загальні кроки компартментного моделювання наступним чином:

1. Визначення величин, які нас цікавлять, у вигляді окремих компартментів та присвоєння кожній з величин змінної, яка залежить від часу. Ці змінні будуть змінними стану системи (1);
2. З'єднання компартментів стрілками, які вказують швидкість та напрям потоку (переміщення) величини від одного компартменту до іншого. Швидкість зазвичай позначається безпосередньо над відповідною стрілкою;
3. Запис відповідних диференціальних рівнянь першого порядку для кожної змінної стану моделі. При записі рівнянь на основі графічного представлення моделі, потоки додаються до (або віднімаються від) рівняння зміни стану кінцевого (початкового) вузла. За наявності зовнішніх впливів, їх можна розцінювати як такі, що беруть початок з зовнішніх вузлів;
4. Надання початкових значень та розв'язок системи рівнянь (1).

Компартментні моделі поділяються на відкриті та закриті. У закритих моделях кількості лише переходять між компартментами, тоді як у відкритих системах кількості можуть надходити ззовні або залишати систему. У закритій компартментній моделі сума всіх рівнянь зміни станів дорівнює нулю (для будь-якого моменту часу t):

$$\sum_{i=1}^{n} f_i = 0, \quad t \in \Omega_t. \tag{3}$$

В якості прикладу наведемо систему отриманої з компартментної моделі зображеної на рис. 1:

$$\begin{aligned} \frac{da(t)}{dt} &= \omega_1 - \mu a(t)b(t) - \gamma a(t), \\ \frac{db(t)}{dt} &= \mu a(t)b(t) - \beta b(t), \\ \frac{dc(t)}{dt} &= \gamma a(t) + \beta b(t) - \omega_2, \end{aligned} \tag{4}$$

де $\omega(t) = (\omega_1, \omega_2)$ – зовнішні впливи. Якщо ω_l входить у рівняння зі знаком "+", це відповідає входу у відповідний компартмент; якщо зі знаком "-" – виходу з компартменту.

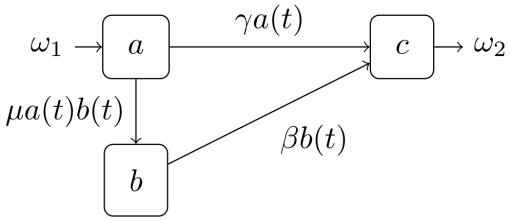


Рис. 1. Приклад компартментної моделі

Компартментні моделі відзначаються високою гнучкістю та адаптивністю, що дозволяє їх застосовувати до широкого спектра задач. Завдяки простоті та наочності графічного представлення, ці моделі легко модифікувати для врахування додаткових процесів або змінних, зовнішніх впливів або інших особливостей реальних систем.

На момент написання статті, програмний продукт підтримує виключно закриті моделі, з незалежними від змінної часу рівняннями станів та сталими параметрами на усьому відрізку дослідження ($\theta(t) = (\theta_1, \theta_2, \ldots, \theta_g) \in \mathbb{R}^g$). Для розглядуваної в роботі SEIR моделі без керування, яка є розширенням класичної моделі SIR [6], компартментна діаграма наведена на рис. 2.

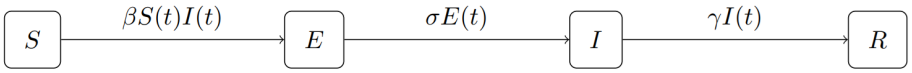


Рис. 2. Компартментна модель SEIR

Система диференціальних рівнянь відповідно має вигляд:

$$\begin{aligned} \frac{dS(t)}{dt} &= -\beta S(t)I(t), \\ \frac{dE(t)}{dt} &= \beta S(t)I(t) - \sigma E(t), \\ \frac{dI(t)}{dt} &= \sigma E(t) - \gamma I(t), \\ \frac{dR(t)}{dt} &= \gamma I(t), \end{aligned} \tag{5}$$

де $\theta\,(t) = (\beta, \sigma, \gamma) \in \mathbb{R}^3$.

У моделі (5) функції стану (компартменти моделі) описують кількість людей певної категорії по відношенню до захворюваності: $S(t)$ – кількість сприйнятливих до захворювання осіб (Susceptible); $E(t)$ – кількість заражених, але ще не заразних осіб (Exposed); $I(t)$ – кількість інфікованих (уже заразних) осіб (Infectious); $R(t)$ – кількість осіб, що одужали та набули імунітет (Recovered).

3. Задача моделювання

Нехай задано систему диференціальних рівнянь (1) на проміжку Ω_t та початкові умови (2). При цьому керування $u(t)$ не застосовується, зовнішні впливи відсутні $\omega(t) = (0, 0, \ldots, 0)$ (модель – закрита), функції динаміки f незалежні від змінної часу, параметри – статичні $\theta(t) = (\theta_1, \theta_2, \ldots, \theta_g) \in \mathbb{R}^g$. Мета задачі – отримати чисельні розв'язки такої системи рівнянь.

Розв'язок задачі отримується шляхом чисельного інтегрування системи (1) з урахуванням початкових умов (2) та заданих параметрів. У роботі використано метод LSODA бібліотеки `SciPy` функції `solve_ivp` (параметр `method` – 'LSODA') [7, 8], який автоматично виявляє жорсткість і перемикається між двома класами багатокрокових схем з адаптивними кроком і порядком: на нежорстких ділянках – предиктор-коректорними формулами Адамса-Моултона (порядок до 12), а на жорстких – формулами зворотного диференціювання BDF (порядок до 5).

Для такого типу задач також широко застосовують інші загальновідомі підходи: метод Рунге-Кутти порядку 5 (4) [9] – вбудована пара порядків 5 і 4 з оцінкою похибки для адаптивного кроку, ефективна для нежорстких систем із хорошим співвідношенням точності та обчислювальної вартості; метод Рунге-Кутти порядку 3(2) [10] – легка вбудована пара порядків 3 і 2 для швидких попередніх проходів та слабко нелінійних нежорстких задач; BDF (backward differentiation formulas) [11] – неявний багатокроковий метод зворотного диференціювання змінного порядку (1-5), ефективний для жорстких систем.

4. Задача оптимального керування

Мета задачі оптимального керування [12–14] полягає у знаходженні такого керування $u^*(t)$ з множини допустимих керувань U, що:

$$\psi(u^*(t)) = \min_{u \in U} \psi(u(t)), \tag{6}$$

де $\psi(u(t))$ – цільовий функціонал, що визначається як:

$$\psi(u(t)) = \int_{\Omega_t} g(y(u(t),t), u(t), \theta)\, dt, \tag{7}$$

де $g(y(u(t),t), u(t), \theta)$ – функція витрат в окремо взятий момент часу.

Функція керування $u(t)$ та функція стану $y(t)$ пов'язані між собою математичною моделлю (1)-(2). На функцію керування накладаються двосторонні обмеження, що визначають множину допустимих керувань U:

$$\begin{aligned} &U = \left\{u : u^- \le u(t) \le u^+,\, t \in \Omega_t\right\}, \\ &u(t) = (u_1(t), u_2(t), \ldots, u_m(t)), \\ &u^-, u^+ \in \mathbb{R}^m, \end{aligned} \tag{8}$$

при цьому зовнішні впливи відсутні $\omega(t) = (0, 0, \ldots, 0)$ (модель – закрита), функції динаміки f незалежні від змінної часу, параметри – статичні $\theta(t) = (\theta_1, \theta_2, \ldots, \theta_g) \in \mathbb{R}^g$.

Задача оптимального керування (6) розв'язується згідно принципу максимуму Понтрягіна [15]. Відповідно, вводиться допоміжна вектор-функція $\lambda(t) = (\lambda_1(t), \lambda_2(t), \ldots, \lambda_n(t))$, $\lambda_i(t) \in C^1(\Omega_t)$, $t \in \Omega_t, i = 1, 2, \ldots, n$, яка повинна задовольняти систему допоміжних рівнянь (9) та умови трансверсальності (10):

$$\frac{d\lambda_i(t)}{dt} = -\frac{\partial H}{\partial y_i}, \quad t \in \Omega_t, \quad i = 1, 2, \ldots, n, \tag{9}$$

$$\lambda_i(t_f) = 0, \tag{10}$$

де $H = H(y(u(t),t), u(t), \lambda(t), \theta)$ – функція Гамільтона:

$$H = g(y(u(t),t), u(t), \theta) + \sum_{i=1}^{n} \lambda_i(t) f_i(y(u(t),t), u(t), \theta). \tag{11}$$

Відповідно до принципу максимуму Понтрягіна, необхідно знайти таке керування $u^*(t)$, яке мінімізуватиме функцію Гамільтона (11) в кожен момент часу t:

$$u^*(t) = \arg\min_{u(t)} H\left(y(u(t),t), u(t), \lambda(t), \theta\right). \tag{12}$$

Відповідно керування $u(t)$ вважається оптимальними якщо:

$$\frac{\partial H}{\partial u_j} = 0, \quad t \in \Omega_t, \quad j = 1, 2, \ldots, m. \tag{13}$$

Умова (13) – це необхідна умова стаціонарності для мінімізації H за u_j при фіксованих y, λ.

Розв'язок задачі (6) отримується методом прямого-зворотного проходу (forward-backward sweep method) [15]. Його суть полягає у ітеративному уточненні керування шляхом побудови послідовності наближень $u^{[k]}(t), k = 0, 1, 2, \ldots$. Формальний алгоритм методу наступний:

1. Обирається допустиме початкове наближення значень керування $u^{[0]}(t)$;
2. Прямий прохід: за поточним $u^{[k]}(t)$ розв'язується система (1) з початковими умовами (2). Інтегрування виконується LSODA з `dense_output`=True. Отримується неперервна інтерполяційна функція $Y^{[k]}(t) = sol^{[k]}_{fwd}(t)$, яка дозволяє отримувати $y^{[k]}(t)$ у довільній точці $t \in [0, t_f]$, та згенерована інтегратором адаптивна сітка $T^{[k]}_{fwd}$;
3. Зворотний прохід: за $Y^{[k]}(t)$ та $u^{[k]}(t)$ розв'язується (9) з умовами (10) у напрямку від t_f до 0. Аналогічно отримується неперервна інтерполяційна функція $\Lambda^{[k]}(t) = sol^{[k]}_{bwd}(t)$, та згенерована інтегратором адаптивна сітка $T^{[k]}_{bwd}$;
4. Оновлення керування за $Y^{[k]}(t)$ та $\Lambda^{[k]}(t)$: якщо H є строго опуклою за u_j, умова (13) визначає єдиний $u^*_j(t)$. Звідси, наступне наближення керування $u^{[k+1]}_j(t)$ формується як проєкція мінімуму $\tilde{u}^{[k]}_j(t)$ (розв'язок (13)) на відрізок допустимих значень $\left[u^-_j, u^+_j\right]$:

$$\tilde{u}^{[k]}_j(t) = \arg\min_{u_j(t)} H\left(Y^{[k]}(t), \left(u^{[k]}_1(t), \ldots, u_j(t), \ldots, u^{[k]}_m(t)\right), \Lambda^{[k]}(t), \theta\right) \tag{14}$$

$$u^{[k+1]}_j(t) = \min\left(u^+_j, \max\left(u^-_j, \tilde{u}^{[k]}_j(t)\right)\right), \tag{15}$$

   де k – порядковий номер поточної ітерації. Тобто $u^{[k+1]}(t)$ формується як неперервна функція від часу;
5. Перевірка умов зупинки: якщо $\|u^{[k+1]}\left(T^{[k]}\right) - u^{[k]}\left(T^{[k]}\right)\| < \varepsilon$, $T^{[k]} = T^{[k]}_{fwd} \cup T^{[k]}_{bwd}$ або досягнуто ліміт ітерацій, то $u^{[k+1]}$ приймається як розв'язок; інакше $k \to k+1$ і перехід до кроку 2.

Для стабілізації збіжності на кроці 4 використовується релаксація:

$$u^{[k+1]}_j(t) = (1-\eta)u^{[k]}_j(t) + \eta \cdot \min\left(u^+_j, \max\left(u^-_j, \tilde{u}^{[k]}_j(t)\right)\right), \tag{16}$$

де $\eta \in (0, 1]$ – коефіцієнт релаксації.

В разі якщо H лінійна за u_j, з (13) та обмежень (8) випливає що оптимум в точці t досягається на одній з меж $\left[u^-_j, u^+_j\right]$. Позначимо $\phi_j(t) = \frac{\partial H}{\partial u_j}(t)$, маємо:

$$u^{[k+1]}_j(t) = \begin{cases} u^-_j, & \phi_j(t) > 0, \\ u^+_j, & \phi_j(t) \le 0. \end{cases} \tag{17}$$

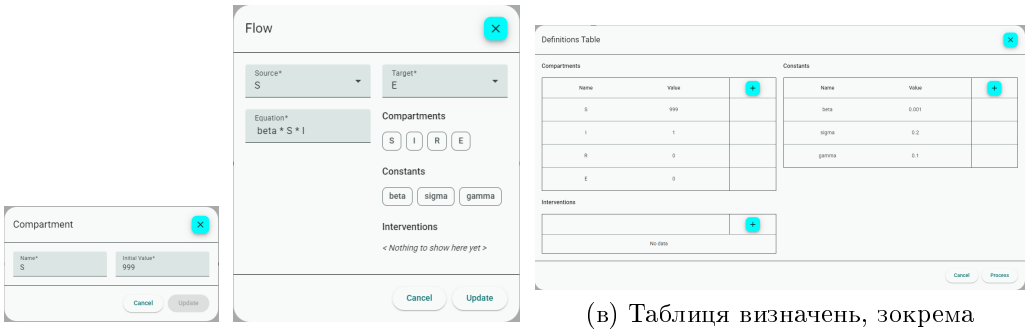
5. Опис програмного забезпечення

Програмне забезпечення (CompLab) реалізовано як інтерактивний, візуальний та інтуїтивний конструктор компартментних моделей з низкою допоміжних функцій: інтерактивний граф моделі, графічне відображення рівнянь / результатів, робочі простори, структуроване авторозміщення компартментів на робочій поверхні, імпорт / експорт моделей/результатів, перевикористання результатів (витяг вхідної моделі / перезапуск розв'язання задачі з можливістю зміни вхідних параметрів задачі), тощо. CompLab реалізовано за архітектурою клієнт-сервер:

- **Клієнт**: Angular проєкт – інтерактивний графічний конструктор/редактор компартментних моделей;
- **Сервер**: Python (Flask) проєкт – обробник надісланих клієнтом задач моделювання та оптимального керування.

У CompLab робота відбувається відносно компартментного представлення математичної моделі, тобто зваженого орієнтованого графа. Користувач у клієнтському інтерфейсі графічно будує цей граф і визначає його властивості (Рис. 3). Редактор графа реалізовано на базі бібліотеки `Cytoscape.js` [16] та власних модулів її розширення, завдяки чому граф є інтерактивним: вузли можна вільно переміщувати, є окрема кнопка "авторозміщення" та відповідні тригери (зміна розміру вікна, додавання нових вузлів) автоматично впорядковують його на екрані, тощо. Відповідно, користувач повинен задати:

- компартменти моделі та їх початкові значення;
- рівняння потоків та компартменти, які вони з'єднують;
- позначення констант та їхні значення;
- позначення керувань (за наявності).



(а) Компартмент (б) Потік (в) Таблиця визначень, зокрема констант та керувань

Рис. 3. Діалогові вікна створення компартментів, потоків та визначення констант і керувань в CompLab

Процес моделювання. Для запуску процесу моделювання користувач задає вхідні параметри задачі: кінцевий момент часу t_f та кількість рівновіддалених точок розбиття N проміжку Ω_t в яких будуть отримані розв'язки для виведення графіків динаміки. Клієнт надсилає запит на розв'язання задачі на сервер. Запит містить наступну інформацію:

- **Компартментна модель**:
  - Визначення компартментів (імена та початкові значення);
  - Визначення потоків (рівняння потоку, ідентифікатори компартмента-джерела та компартмента-призначення);
  - Визначення констант (імена та значення);
  - Визначення керувань (імена).
- **Параметри задачі:** кінцевий момент часу t_f та кількість рівновіддалених точок розбиття N проміжку Ω_t.

Загалом, після отримання запиту сервер виконує наступне:

1. Конвертує компартментну модель в систему диференціальних рівнянь. При цьому рівняння потоків за допомогою методів бібліотеки `SymPy` переводяться в символьні вирази символьної математики, оскільки вони задаються користувачем з клавіатури текстом;
2. Чисельно інтегрує отриману систему з урахуванням заданих початкових умов та параметрів. При цьому:
   - У разі якщо досліджується модель з визначеними керуваннями, вони прирівнюються до нуля під час обчислень;
   - Для інтегрування використовується реалізація методу LSODA бібліотеки `SciPy` функції `solve_ivp` (параметр `method` – 'LSODA');
   - За допомогою параметра `dense_output = True` функції `solve_ivp` отримується інтерполяційна функція компартментів.
3. Формує та повертає розв'язок задачі клієнту. Розв'язок складається з вхідної моделі та параметрів, а також значень компартментів в $N+1$ рівновіддалених точках розбиття проміжку Ω_t, обчислених за інтерполяційною функцією з кроку 2.

Отриманий розв'язок задачі та вхідні дані клієнт відображає в окремо виділеній панелі. З неї користувач може:

- Експортувати результати (з/без моделі та вхідних параметрів задачі);
- Створити новий робочий простір на основі вхідної моделі відображених результатів;
- Перезапустити розв'язання задачі з тими ж вхідними даними. При цьому користувачу дається можливість змінити параметри самої задачі (кінцевий момент часу, кількість точок розбиття).

Процес оптимального керування. Для задачі оптимального керування вхідними параметрами є:

- Кінцевий момент часу t_f;
- Кількість рівновіддалених точок розбиття N проміжку Ω_t в яких будуть отримані розв'язки для виведення графіків динаміки;
- Верхні та нижні межі значень кожного керування u_j;
- Функція витрат $g(y(u(t),t),u(t),\theta)$.

Запит на розв'язання задачі, подібно до задачі моделювання, містить саму компартментну модель та параметри задачі оптимального керування. Сервер виконує наступні кроки:

1. Конвертує компартментну модель в систему диференціальних рівнянь;

2. Конвертує функцію витрат в символьний вираз;
3. Складає функцію Гамільтона за (11);
4. Виводить допоміжну систему рівнянь за (9);
5. Чисельно інтегрує систему отриману з кроку 1 з урахуванням заданих початкових умов та параметрів. При цьому керування прирівнюються до нуля.
6. Обчислює значення функціоналу (7). Тобто його значення без будь-якого керування;
7. Методом прямого-зворотного проходу отримує чисельний розв'язок задачі (6). При цьому:
   - Початкове наближення керувань: $u^{[0]}(t) = \frac{u^+ + u^-}{2}$;
   - Програмне забезпечення, для кожного керування u_j, здійснює спробу розв'язання умов оптимальності (13) за допомогою методу `solve` бібліотеки `SymPy`. За наявності розв'язку застосовується схема (16); за його відсутності – схема (17).
   - За допомогою параметра `dense_output = True` функції `solve_ivp` отримується інтерполяційна функція компартментів.
8. Формує та повертає розв'язок задачі клієнту. Розв'язок складається з вхідної моделі та параметрів, значень функціоналу (7) до та після застосування оптимального керування, функції Гамільтона, допоміжної системи рівнянь, а також значень компартментів (до та після застосування оптимального керування) і керувань в $N + 1$ рівновіддалених точках розбиття проміжку Ω_t.

Як і для задачі моделювання, панель результатів відображає вхідні модель та параметри задачі, а також графічні результати. Для задачі оптимального керування додатково відображаються неграфічні результати: значення функціоналу (7) до та після застосування оптимального керування; функція Гамільтона; допоміжна система рівнянь.

6. Результати числових експериментів

Розглянемо результати роботи програмного забезпечення на серії прикладів на основі SEIR моделі (5) та її модифікацій з додаванням керувань.

Приклад 1. Базова SEIR модель без керування. Розглядається задача моделювання для базової SEIR моделі (5). Вигляд SEIR моделі в програмі співставний з рис. 2. Вигляд моделі та параметри задачі відображені на рис. 4, результати – на рис. 5.

Приклад 2. SEIR модель із одним керуванням u_1 (наприклад щепленням). Розглянемо задачу оптимального керування для модифікації моделі наведеної в попередньому прикладі. Вводяться керування u_1, новий параметр (константа) δ та додатковий потік що з'єднує компартмент S з компартментом R:

$$\begin{aligned}
\frac{dS(t)}{dt} &= -\beta S(t)I(t) - \delta u_1(t)S(t),\\
\frac{dE(t)}{dt} &= \beta S(t)I(t) - \sigma E(t),\\
\frac{dI(t)}{dt} &= \sigma E(t) - \gamma I(t),\\
\frac{dR(t)}{dt} &= \gamma I(t) + \delta u_1(t)S(t).
\end{aligned} \tag{18}$$

Потрібно знайти таке керування $u^*(t)$ з множини допустимих керувань U, що:

$$\begin{aligned}
\psi(u^*(t)) &= \min_{u\in U}\psi(u(t)),\\
\psi(u(t)) &= \int_0^{100}\left(I(t) + 0.5u_1^2(t)\right)\,dt.
\end{aligned} \tag{19}$$

Множина допустимих керувань, визначена обмеженнями:

$$\begin{aligned}
U &= \left\{u : u^- \le u(t) \le u^+,\, t\in[0,100]\right\},\\
u(t) &= u_1(t),\\
u^- &= 0, u^+ = 1.
\end{aligned} \tag{20}$$

Вигляд моделі CompLab та вхідні параметри задачі приведено на рис. 6, результати розв'язку задачі оптимального керування (18)-(20) приведені на рис. 7.

Приклад 3. SEIR модель із двома керуваннями u_1 та u_2 (наприклад щепленням та носінням масок відповідно). Модифікуємо задачу приведену в попередньому прикладі. Введемо ще одне керування u_2 та новий параметр (константу) ρ:

$$\begin{aligned}
\frac{dS(t)}{dt} &= -\left(1 - u_2(t) + (1-\rho)\,u_2(t)\right)\beta S(t)I(t) - \delta u_1(t)S(t),\\
\frac{dE(t)}{dt} &= \left(1 - u_2(t) + (1-\rho)\,u_2(t)\right)\beta S(t)I(t) - \sigma E(t),\\
\frac{dI(t)}{dt} &= \sigma E(t) - \gamma I(t),\\
\frac{dR(t)}{dt} &= \gamma I(t) + \delta u_1(t)S(t).
\end{aligned} \tag{21}$$

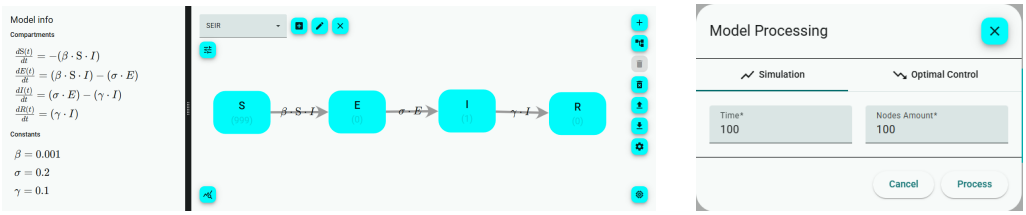
Потрібно знайти таке керування $u^*(t)$ з множини допустимих керувань U, що:

$$\begin{aligned}
\psi(u^*(t)) &= \min_{u\in U}\psi(u(t)),\\
\psi(u(t)) &= \int_0^{100}\left(I(t) + 0.5u_1^2(t) + 0.4u_2^2(t)\right)\,dt.
\end{aligned} \tag{22}$$

Множина допустимих керувань, визначена обмеженнями:

$$U = \left\{u : u^- \leq u(t) \leq u^+,\, t \in [0, 100]\right\},$$
$$u(t) = (u_1(t), u_2(t)), \tag{23}$$
$$u^- = (0, 0), u^+ = (1, 1).$$

Вигляд моделі в CompLab та параметри задачі приведено на рис. 8, результати розв'язку задачі оптимального керування (21)-(23) – на рис. 9.



(а) (б)

Рис. 4. (а) SEIR модель (5) в CompLab: компартментне подання, система диференціальних рівнянь, початкові значення компартментів ($S(0) = 999$, $E(0) = 0$, $I(0) = 1$, $R(0) = 0$) та значення параметрів моделі ($\beta = 0.001, \sigma = 0.2, \gamma = 0.1$); (б) параметри задачі моделювання: кінцевий момент часу ($t_f = 100$) та кількість рівновіддалених точок розбиття ($N = 100$)

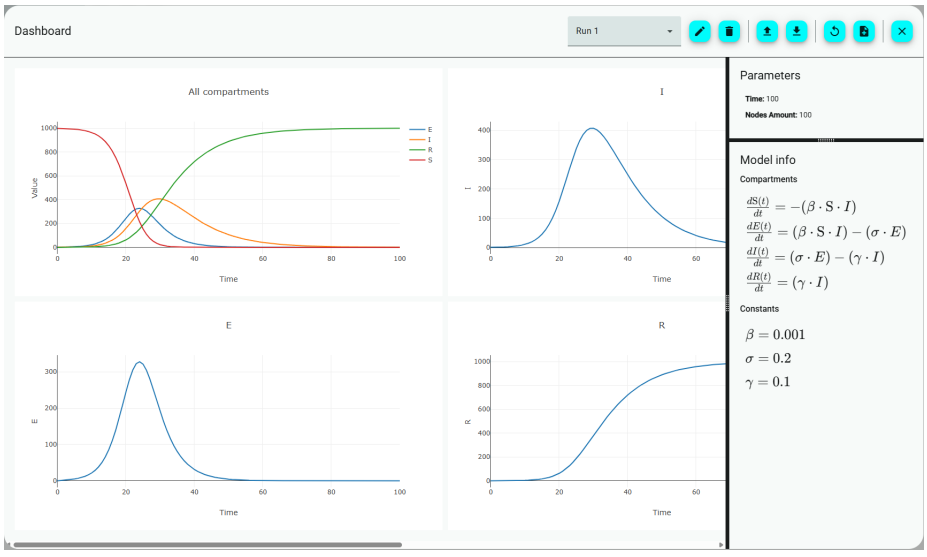
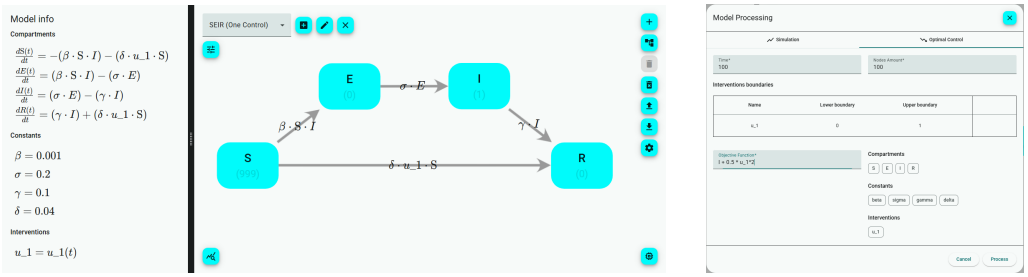
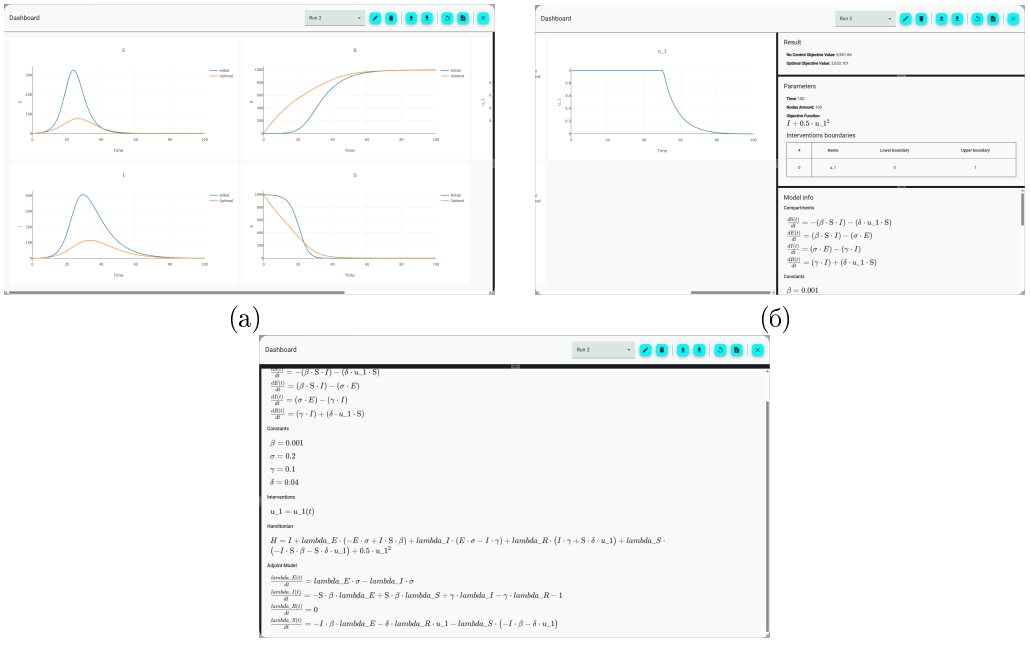


Рис. 5. Результати розв'язку задачі моделювання: графіки компартментів, параметри задачі, система диференціальних рівнянь моделі та значення параметрів моделі



(а) (б)

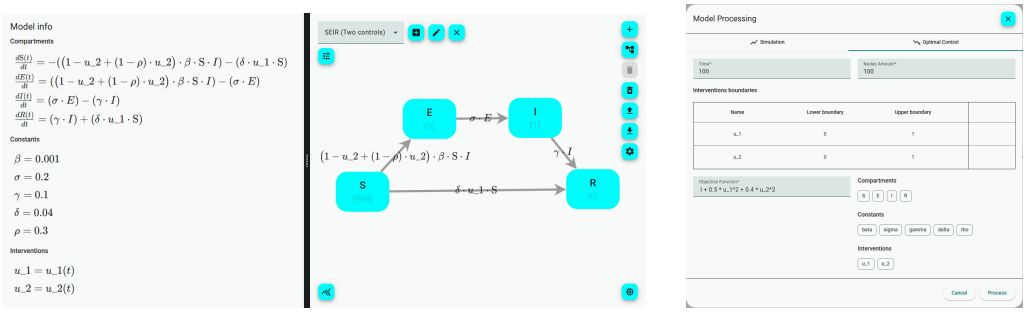
Рис. 6. (а) Модифікована SEIR модель (18) з одним керуванням в CompLab: компартментне подання, система диференціальних рівнянь, початкові значення компартментів ($S(0) = 999$, $E(0) = 0$, $I(0) = 1$, $R(0) = 0$), значення параметрів моделі ($\beta = 0.001, \sigma = 0.2, \gamma = 0.1, \delta = 0.04$) та визначення керування ($u_1 = u_1(t)$); (б) параметри задачі оптимального керування (18)-(20): кінцевий момент часу ($t_f = 100$), кількість рівновіддалених точок розбиття ($N = 100$), межі керування ($u_1^- = 0, u_1^+ = 1$) та функція витрат ($g = I + 0.5u_1^2$)



(а) (б)

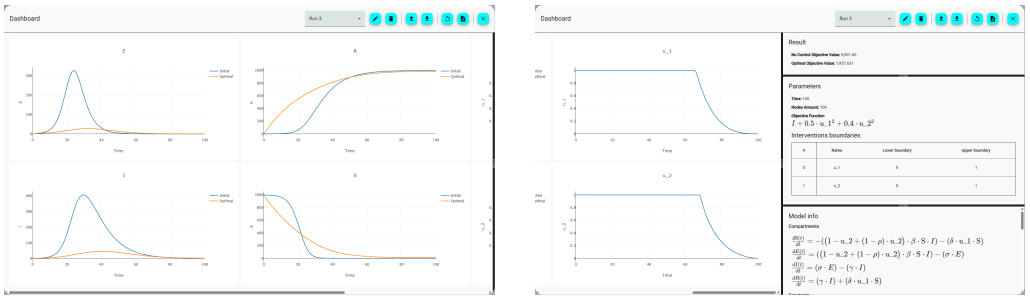
(в)

Рис. 7. (а) графіки компартментів до (Initial. Синій колір) та після (Optimal. Помаранчевий колір) застосування оптимального керування; (б) графік керування u_1, значення функціоналу до та після застосування оптимального керування, параметри задачі та система диференціальних рівнянь моделі; (в) значення параметрів моделі, визначення керування, функція Гамільтона та допоміжна система рівнянь

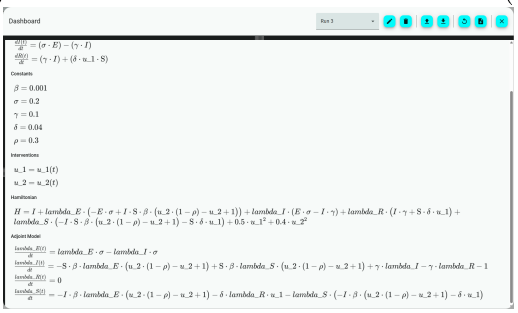


(а) (б)

Рис. 8. (а) Модифікована SEIR модель (21) з двома керуваннями в CompLab: компартментне подання, система диференціальних рівнянь, початкові значення компартментів ($S(0) = 999$, $E(0) = 0$, $I(0) = 1$, $R(0) = 0$), значення параметрів моделі ($\beta = 0.001, \sigma = 0.2, \gamma = 0.1, \delta = 0.04, \rho = 0.3$) та визначення керувань ($u_1 = u_1(t), u_2 = u_2(t)$); (б) параметри задачі оптимального керування (21)-(23): кінцевий момент часу ($t_f = 100$), кількість рівновіддалених точок розбиття ($N = 100$), межі керувань ($u_1^- = 0$, $u_1^+ = 1$, $u_2^- = 0$, $u_2^+ = 1$) та функція витрат ($g = I + 0.5u_1^2 + 0.4u_2^2$)



(а) (б)



(в)

Рис. 9. (а) графіки компартментів до (Initial. Синій колір) та після (Optimal. Помаранчевий колір) застосування оптимального керування; (б) графіки керувань u_1 та u_2, значення функціоналу до та після застосування оптимального керування, параметри задачі та система диференціальних рівнянь моделі; (в) значення параметрів моделі, визначення керувань, функція Гамільтона та допоміжна система рівнянь

7. Аналіз результатів

Представлене програмне забезпечення – це робоче середовище для постановки та розв'язання задач моделювання та оптимального керування закритими компартментними моделями з незалежними від змінної часу рівняннями станів та сталими параметрами на усьому відрізку дослідження (наприклад, SEIR). Одна з ключових переваг – автоматизація. Від користувача вимагається задати структуру моделі, її початкові умови та параметри відповідної задачі. Далі CompLab використовуючи надану інформацію самостійно формує потрібні для розв'язання структури (систему диференціальних рівнянь моделі, функцію Гамільтона, систему допоміжних рівнянь, тощо) та виконує саме розв'язання задачі з їх допомогою. CompLab відображає користувачу доречні елементи цих структур як частину результатів. Результати розв'язання задачі, які бачить користувач, надають йому повну інформацію щодо того якими були вхідні дані, за яких ці результати було отримано, сам розв'язок, а також додаткові дані, якщо доречно. Це забезпечує відтворюваність та прозорість процесу. Розглянемо детальніше результати отримані в вищеописаних прикладах.

Приклад 1. Базова SEIR модель без керування. Динаміка SEIR відтворює спалах із короткою латентною фазою та повним вичерпанням сприйнятливих осіб: S стрімко прямує до нуля на проміжку $t \in [10, 35]$; E зростає раніше за I (затримка приблизно 5 днів) та досягає піку $E_{max} \approx 330$ на $t \approx 24$, після чого згасає; I пікує пізніше – $I_{max} \approx 407$ на $t \approx 30$ і спадає повільніше; R наростає сигмоїдально й насичується під кінець періоду спостереження. Це відповідає очікуваній поведінці SEIR: ранній пік заражених E, відкладений пік інфікованих I, монотонне насичення одужалих R.

Приклад 2. SEIR модель із одним керуванням u_1 (наприклад щепленням). Оптимальна траєкторія $u_1(t)$ тримається верхньої межі до $t \approx 50$ після чого монотонно прямує до нуля, що створює інтенсивне раннє переведення $S \to R$. Це зменшує кількість сприйнятливих осіб завчасно (крива S починає спадати раніше й плавніше), прискорює наростання R та суттєво "зрізає" і дещо зміщує піки E та I: з $E_{max} \approx 330$ на $t \approx 24$ до $E_{max}^{[u_1]} \approx 77$ на $t \approx 27$ та з $I_{max} \approx 407$ на $t \approx 30$ до $I_{max}^{[u_1]} \approx 116$ на $t \approx 33$ відповідно. За функціоналом $\psi = \int_0^{100}(I + 0.5u_1^2)dt$ це дає зниження значення з $9,991.66$ до $3,633.101$ ($\approx 2.75\times$), тобто значний виграш навіть з урахуванням "ціни" керування.

Приклад 3. SEIR модель із двома керуваннями u_1 та u_2 (наприклад щепленням та носінням масок відповідно). Оптимум тримає обидва керування близько верхньої межі: u_1 до $t \approx 65$; u_2 до $t \approx 68$. Далі обидва керування монотонно прямують до нуля. Тобто, як і в прикладі 2, спостерігається інтенсивне раннє переведення $S \to R$ завдяки u_1, а також паралельне уповільнення переведення $S \to E$ завдяки u_2. Як наслідок S починає спадати від самого початку періоду спостереження і виснажується набагато плавніше й пізніше, R наростає дещо повільніше, у порівнянні з прикладом 2, а піки E та I ще більше "урізаються" та зміщуються: з $E_{max} \approx 330$ на $t \approx 24$ до $E_{max}^{[u_1,u_2]} \approx 28$ на $t \approx 32$ та з $I_{max} \approx 407$ на $t \approx 30$ до $I_{max}^{[u_1,u_2]} \approx 47$ на $t \approx 40$ відповідно. Значення функціоналу $\psi =$

$\int_0^{100}(I + 0.5u_1^2 + 0.4u_2^2)dt$ зменшується з $9,991.66$ до $1,957.631$ ($\approx 5.1\times$). Це свідчить про те, що уповільнюючий ефект u_2 в комбінації з прискорюючим ефектом u_1 дає значно кращий результат при меншій ціні.

Варто зазначити, що існують інші програмні продукти, що реалізують схожі ідеї, зокрема K-SEIR-Sim [17], STEM (Spatiotemporal Epidemiological Modeler) [18], GLEAMviz [19] та EPIGUI [20]. На їхньому тлі CompLab виділяється поєднанням кількох цінних рис в одному інструменті: він дозволяє самостійно задавати структуру моделі (не обмежуючись заздалегідь визначеним набором); має зручний та інтуїтивний інтерфейс з набором допоміжних інструментів; забезпечує низький поріг входу – користувачу достатньо окреслити граф моделі, початкові умови та параметри задачі, далі програма працює самостійно; архітектурно готовий працювати як застосунок і як веб-сервіс; і, зокрема на відміну від згаданих рішень, здатний розв'язувати задачі оптимального керування. Кожна з цих рис має самостійну дослідницьку цінність, а їх поєднання у єдиному середовищі додатково підсилює унікальність й прикладну корисність інструменту.

8. Висновки

Запропоновано й реалізовано CompLab – готовий до використання програмний продукт, який надає користувачу повний функціонал для моделювання та оптимального керування закритими компартментними моделями з незалежними від змінної часу рівняннями станів та сталими параметрами на усьому відрізку дослідження. Для моделювання використано метод LSODA, для задач оптимізації – принцип максимуму Понтрягіна з прямим-зворотним проходом; їх роботу продемонстровано на серії прикладів. Додатково продукт надає набір допоміжних інструментів: імпорт/експорт, структуроване автопозиціювання, робочі простори, повторне використання результатів та ряд інших допоміжних функцій. Водночас, проєкт CompLab продовжує активно вдосконалюватись та розвиватись. Найближчими планами щодо його розвитку є:

- Розширення функціоналу додаванням можливості розв'язання задачі ідентифікації параметрів моделі, що дозволить оцінювати невідомі характеристики досліджуваної системи;
- Додавання підтримки відкритих компартментних моделей, що сприятиме застосуванню продукту до більш широкого класу задач;
- Портування CompLab у desktop-версію (додаток), що додасть ще один спосіб використання продукту зі спрощеним процесом встановлення та запуску.

Ці кроки допоможуть розширити сферу застосувань та підвищать ефективність роботи з динамічними системами.

Список використаної літератури

1. Okyere E. Deterministic Epidemic Models For Ebola Infection With Time-dependent Controls / E.Okyere, J.D. Ankamah, A.K. Hunkpe, D. Mensah // arXiv, math. DS. – 2019.
2. Ferjouchia H. Optimal control strategy of COVID-19 spread in Morocco using SEIRD model / H. Ferjouchia, A. Kouidere, O. Zakary, M. Rachik // Moroccan Journal of Pure and Applied Analysis. – 2020. – Vol. 7, № 1. – P. 66–79.
3. Melo L. Modeling COVID-19 Spread through the SEIRD Epidemic Model and Optimal Control / L. Melo // Proceedings of GREAT Day. – 2022. – Vol. 2021. – Article 19.
4. Sameni R. Mathematical Modeling of Epidemic Diseases: A Case Study of the COVID-19 Coronavirus / R. Sameni // arXiv. – 2020.

5. Remuzzi A. COVID-19 and Italy: what next? / A.Remuzzi, G. Remuzzi // The Lancet. – 2020. – Vol. 395, № 10231. – P. 1225–1228.
6. Kermack W.O. A Contribution to the Mathematical Theory of Epidemics / W.O. Kermack, A.G. McKendrick // Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences. – 1927. – Vol. 115, № 772. – P. 700–721.
7. Hindmarsh A.C. ODEPACK, A Systematized Collection of ODE Solvers / A.C. Hindmarsh // IMACS Transactions on Scientific Computation. – 1983. – Vol. 1. – P. 55–64.
8. Petzold L. Automatic Selection of Methods for Solving Stiff and Nonstiff Systems of Ordinary Differential Equations / L. Petzold // SIAM Journal on Scientific and Statistical Computing. – 1983, March. – Vol. 4. – P. 136–148.
9. Dormand J.R. A family of embedded Runge-Kutta formulae / J.R. Dormand., P.J. Prince // Journal of Computational and Applied Mathematics. – 1980. – Vol. 6, № 1. – P. 19–26.
10. Bogacki P. A 3(2) pair of Runge-Kutta formulas / P. Bogacki, L.F. Shampine // Applied Mathematics Letters. – 1989. – Vol. 2, № 4. – P. 321–325.
11. Curtiss C.F. Integration of Stiff Equations / C.F. Curtiss, J.O. Hirschfelder // Proceedings of the National Academy of Sciences. – 1952. – Vol. 38, № 3. – P. 235–243.
12. Fleming W. Deterministic and Stochastic Optimal Control / W.Fleming, R. Rishel New-York: Springer, 1975.
13. Aldila D. Optimal control problem on COVID-19 disease transmission model considering medical mask, disinfectants and media campaign / D. Aldila // E3S Web of Conferences. – 2020. – Vol. 202. – 12 p.
14. Aldila D. Cost-effectiveness and backward bifurcation analysis on COVID-19 transmission model considering direct and indirect transmission / D. Aldila // Communications in Mathematical Biology and Neuroscience. – 2020. – Vol. 2020. – 28 p.
15. Pontryagin L. L.S. Pontryagin Selected Works: In Four Volumes / L. Pontryagin // Gordon and Breach Science Publishers. – 1986.
16. Franz M. Cytoscape.js: a graph theory library for visualisation and analysis / M. Franz, C.T. Lopes, G. Huck, Y. Dong, O. Sumer, G.D. Bader // Bioinformatics. – 2016. – Vol. 32, № 2. – P. 309–311.
17. Wang H. K-SEIR-Sim: A simple customized software for simulating the spread of infectious diseases / H. Wang, Z. Miao, C. Zhang, X. Wei, X. Li // Computational and Structural Biotechnology Journal. – 2021. – Vol. 19. – P. 1966–1975.
18. Douglas J.V. STEM: An Open Source Tool for Disease Modeling / J.V. Douglas, S. Bianco, S. Edlund, T. Engelhardt, M. Filter, T. Gunther, K.M. Hu, E.J. Nixon, N.L. Sevilla, A. Swaid, J.H. Kaufman // Health Security. – 2019. – Vol. 17, № 4. – P. 291–306.
19. Van den Broeck W. The GLEaMviz computational tool, a publicly available software to explore realistic epidemic spreading scenarios at the global scale / W. Van den Broeck, C. Gioannini, B. Goncalves, M. Quaggiotto, V. Colizza, A. Vespignani // BMC Infectious Diseases. – 2011. – Vol. 11, Article 37.
20. Pinto E. EPIGUI: Graphical User Interface for Simulating Epidemics on Networks / E. Pinto, E. Nepomuceno, J. Kusak, A. Campanharo // Trends in Computational and Applied Mathematics. – 2023. – Vol. 24. – P. 91–120.

Стаття: надійшла до редколегії 02.09.2025
доопрацьована 30.09.2025
прийнята до друку 08.10.2025

CLIENT-SERVER TOOLKIT FOR COMPARTMENTAL MODELING AND OPTIMAL CONTROL

Y. Hnytka, Y. Borysyuk, M. Shcherbatyy

Ivan Franko National University of Lviv,
1, Universytetska str., 79000, Lviv, Ukraine,
e-mail: yurahnytka3@gmail.com,
yaryna.borysyuk@lnu.edu.ua, mykhaylo.shcherbatyy@lnu.edu.ua

This paper presents CompLab – a client-server (Angular / Python + Flask) software suite for an end-to-end workflow with compartmental models: from graphical construction of the structure to solving forward-simulation problems and optimal-control problems. The user specifies the model as a directed weighted graph; the server then automatically constructs the system of ordinary differential equations from that graph (using `SymPy` to parse algebraic flow expressions), and either integrates that system with the LSODA method (the `SciPy` implementation `solve_ivp`, `method='LSODA'`, `dense_output=True`) or solves the optimal-control problem via Pontryagin's maximum principle with a forward-backward sweep. For optimization, the Hamiltonian and the adjoint system are generated automatically; control updates are performed with projection onto admissible bounds and relaxation, and when the Hamiltonian is linear in the controls a bang-bang scheme is applied. The implementation uses continuous solution interpolants (`dense_output`) instead of fixed grids, which simplifies coupling of the forward and backward passes and provides flexible rendering of results at arbitrary resolutions. The tool features an interactive graph editor (based on `Cytoscape.js`) with auto-layout, workspaces, import/export of models and results, reuse of previous runs, output of intermediate artifacts (equation system, Hamiltonian, adjoint equations), etc. The current version supports closed models with time-invariant dynamics and constant parameters. CompLab's capabilities are demonstrated on the basic SEIR model and on two of its modifications with one and two controls.

Key words: mathematical modeling, compartmental models, optimal control, software, Pontryagin's maximum principle.