МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ЧЕРНІВЕЦЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ
ІМЕНІ ЮРІЯ ФЕДЬКОВИЧА

Кваліфікаційна наукова праця
на правах рукопису

Літвінчук Юлія Анатоліївна

УДК 004.021:004.032.26]:004.8

ДИСЕРТАЦІЯ

ПОБУДОВА САМОАДАПТИВНИХ АЛГОРИТМІВ НА ОСНОВІ НЕЙРОННИХ МЕРЕЖ

113 – Прикладна математика
11 – Математика та статистика

Подається на здобуття наукового ступеня доктора філософії.

Дисертація містить результати власних досліджень. Використання ідей, результатів і текстів авторів мають посилання на відповідне джерело

Ю.А. Літвінчук

Науковий керівник **Малик Ігор Володимирович**, доктор фізико-математичних наук, доцент

Чернівці – 2024

АНОТАЦІЯ

Літвінчук Ю.А. Побудова самоадаптивних алгоритмів на основі нейронних мереж. – Кваліфікаційна наукова праця на правах рукопису.

Дисертація на здобуття наукового ступеня доктора філософії за спеціальністю 113 – Прикладна математика. – Чернівецький національний університет імені Юрія Федьковича МОН України, Чернівці, 2024.

Дисертаційна робота присвячена побудові самоадаптивних алгоритмів за допомогою сумішей розподілів на основі розширеного алгоритму еволюційної стратегії з адаптацією коваріаційної матриці (CMA-ES) для оцінки параметрів складних систем.

Результати дисертаційної роботи є підґрунтям для подальших теоретичних і практичних наукових розробок у дослідженні теорії еволюційних алгоритмів та розв'язанні оптимізаційних задач.

Дисертація складається із вступу, трьох розділів, висновків, переліку використаних джерел та додатку.

У вступі обґрунтовано актуальність теми дослідження, сформульовано мету, завдання, предмет, об'єкт та методи дослідження, вказано наукову новизну, теоретичне та практичне значення отриманих результатів, проаналізовано зв'язок роботи з науковими дослідженнями та особистий внесок здобувача, а також наведено відомості про апробацію та публікації основних результатів дисертації. Описано структуру та обсяг дисертаційної роботи.

У першому розділі здійснено огляд наукової літератури, присвяченої основним напрямкам досліджень еволюційних алгоритмів, розглянуто ключові відомості з теорії складних мереж, наведено опис основних напрямів досліджень та визначено завдання, якими займається теорія еволюційних алгоритмів. Проведено детальний огляд основних генетичних алгоритмів та еволюційної стратегії, зокрема, а також їх сфери застосування. Детально проаналізовано хронологію розвитку метаевристичних алгоритмів для оптимізації складних систем. У першому пункті розділу 1 розглянуто

існуючі методи оптимізації параметрів нейронних мереж на базі генетичних алгоритмів, етапи роботи та математична модель генетичного алгоритму. У другому пункті зроблено огляд еволюційної стратегії з адаптацією коваріаційної матриці, як одного з потужних алгоритмів оптимізації, що використовується для розв'язування задач складних систем. Пункт три відображає огляд інших еволюційних алгоритмів оптимізації, їх математичні моделі та сфери застосування.

У другому розділі запропоновано розширений самоадаптивний алгоритм CMA-ES для оптимізації на областях з обмеженнями. Алгоритм враховує обмеження та використовує лише однопікові розподіли та цільові функції. Він ефективний для багатьох задач, але може бути неефективним для функцій з великою кількістю екстремумів. Загалом, модифікований алгоритм є перспективним для задач оптимізації зі складними обмеженнями на поведінку цільової функції. Також у другому розділі розроблено покращений самоадаптивний алгоритм CMA-ES для оптимізації параметрів складних систем, зокрема нейронних мереж. Алгоритм відрізняється від класичних тим, що динамічно підлаштовує кількість піків у суміші розподілів під час виконання, уникаючи зайвих обчислень.

Основні характеристики та переваги алгоритму:

- самоадаптивність: алгоритм підлаштовується під конкретну задачу, змінюючи розмірність суміші розподілів.
- універсальність: може використовуватись із різними типами розподілів та застосовуватись до інших еволюційних алгоритмів. При цьому від базового розподілу вимагається лише однопіковість.
- ефективність: дозволяє уникнути зайвих обчислень цільової функції.
- обґрунтованість: адаптивність базується на методах визначення оптимальної кількості кластерів в кластерному аналізі.

Розглянуто модифікацію алгоритму для задач з обмеженнями на область пошуку. Вона враховує обмеження та використовує лише однопікові розподіли. Ця

модифікація ефективна для багатьох задач, але може бути неефективною для функцій з великою кількістю екстремумів.

Загалом, запропонований самоадаптивний алгоритм CMA-ES є перспективним інструментом для ефективної оптимізації складних систем, як зі стандартними, так і з обмеженими областями пошуку.

В загальному розширений CMA-ES алгоритм є ефективним інструментом для оптимізації складних нелінійних та неопуклих задач. Він має ряд переваг перед класичним CMA-ES, таких як:

- здатність враховувати мультимодальні цільової функції;
- збільшення шансів на знаходження глобального оптимуму в порівнянні із класичними евристичними алгоритмами;
- більш ефективний пошук гіперпараметрів складних систем.

Однак розширений CMA-ES алгоритм має один істотний недолік – необхідність задавати розмірність суміші. Даний параметр може впливати на ефективність алгоритму, оскільки занадто мала розмірність може призвести до того, що алгоритм не зможе врахувати всі екстремуми цільової функції, а отже призводить до повільної збіжності. З іншого ж боку, занадто велика розмірність може збільшити час роботи алгоритму.

У цілому, розширений CMA-ES алгоритм є потужним інструментом для оптимізації, який може бути корисним для вирішення задач з мультимодальними цільовими функціями.

У третьому розділі розглянуто задачу пошуку оптимального керування для динамічних систем, що описуються стохастичними диференціальними рівнянням Ito із пуассонівськими збуреннями та марковськими перемиканнями. В даному розділі неведено достатні умови синтезу оптимального керування для стохастичної динамічної системи випадкової структури з пуассонівськими збуреннями та марковськими перемиканнями, причому керування спрямоване на стабілізацію системи за ймовірністю. Достатні умови асимптотичної стійкості за ймовірністю

отримані за допомогою другого методу Ляпунова, в якому важливу роль відіграє побудова відповідних функцій Ляпунова. Також наведено загальну схему дослідження задачі оптимальної стабілізації для стохастичних диференгціальних рівнянь випадкової структури.

Для лінійної систем з квадратичним функціоналом якості розроблено метод синтезу оптимального керування на основі розв'язку рівнянь Ріккаті. Також, для автономних систем побудовано систему диференціальних рівнянь для отримання невідомих матриць, які використовуються для задання оптимального керування. Ще один підхід синтезу оптимального керування грунтуться на методів малого параметру із послідовним визначенням коефіцінтів оптимального керування. Цей підхід пропонує нове вирішення проблеми оптимальної керування для стохастичної динамічної системи з випадковою структурою та марковськими переключеннями. Крім теоретичних здобутків, у третьому розділі розглянуто порівняльний аналіз класичних евристичних алгоритмів пошуку оптимальних керувань для різних значень горизонту проблеми оптимізації. Відзначено ряд переваг та недоліків нового розширеного CMA-ES алгоритму, основні з яких стосуються сердньої кількості виклику цільової функції та точності субоптимальних розв'язків.

У висновках підсумовано основні результати дисертаційного дослідження. У **додатках** подано наукові публікації, в яких відображено основні наукові результати роботи.

Теоретичне значення. Результати роботи, отримані в ході наукового дослідження, є удосконаленням теорії еволюційних алгоритмів. Підходи, запропоновані у дисертації, можуть використовуватися для подальших досліджень у цій галузі, у навчальних курсах кафедри прикладної математики та інформаційних технологій Чернівецького національного університету імені Юрія Федьковича (та інших ЗВО), пов'язаних з інтелектуальним аналізом даних, машинним навчанням, у методичних розробках, навчальних посібниках для освітнього процесу та науково-дослідної роботи студентів та аспірантів.

Практичне значення. Розроблені у дисертаційній роботі алгоритми можуть в подальшому використовуватися для практичного дослідження складних мереж. Основні результати роботи грунтуються на твердженнях теорії ймовірності, теорії машинного навчання, випадкових процесів та методу Монте-Карло.

Отримані результати дисертації збагачують і методи еволюційних алгоритмів, і теорію машинного навчання, зокрема методи та підходи вирішення проблеми оптимізації гіперпараметрів нейронних мереж та інших складних систем. Практичну цінність роботи проілюстровано на оцінюванні оптимального керування для стохастичних диференціальних систем Ітo випадкової структури. Результати роботи знайдуть застосування у подальших дослідженнях із даної тематики.

Ключові слова: нейронні мережі, машинне навчання, суміш розподілів, оптимізація гіперпараметрів, коваріаційна матриця, CMA-ES алгоритм, моделювання, алгоритми кластеризації, оптимізаційна задача, генетичний алгоритм, метод рою часток, випадкове оточення, випадкове блукання, система стохастичних диференціальних рівнянь, марковські перемикання.

ABSTRACT

Litvinchuk Yu.A. Construction of self-adaptive algorithms based on neural networks. – Qualifying scientific work with manuscript rights.

Dissertation for the Doctor of Philosophy degree in specialty 113 – Applied Mathematics. – Chernivtsi National University named after Yury Fedkovich, Ministry of Education and Science of Ukraine, Chernivtsi, 2024.

The dissertation is devoted to the construction of self-adaptive algorithms using mixtures of distributions based on the extended algorithm of the evolutionary adaptation strategy based on the covariance matrix (CMA-ES) for estimating the parameters of complex systems.

The results of the dissertation work are the basis for further theoretical and practical scientific developments in achieving the theory of evolutionary algorithms and solving optimization problems.

The dissertation consists of an introduction, three chapters, conclusions, a list of used sources and appendix.

The introduction substantiates the relevance of the research topic, formulates the goal, task, subject, object and research methods, indicates the scientific novelty, theoretical and practical significance of the obtained results, analyzes the connection of the work with scientific research and the personal contribution of the recipient, and also provides information about the approval and publication of the main results of the dissertation. The structure and scope of the dissertation work are described.

In the **first chapter**, a review of the scientific literature devoted to the main areas of evolutionary algorithm research is carried out, key information from the theory of complex networks is considered, a description of the main areas of research is given, and the tasks that the theory of evolutionary algorithms deals with are defined. A detailed review of the main genetic algorithms and evolutionary strategy, in particular, as well as their scope of application, is carried out. The chronology of the development of metaheuristic algorithms for the optimization of complex systems is analyzed in detail. In the first point of section I, the existing methods of optimization of parameters of neural networks based on genetic algorithms, the stages of work and the mathematical model of the genetic algorithm are considered. In the second point, an overview of the evolutionary strategy with the adaptation of the covariance matrix is made, as one of the powerful optimization algorithms used to solve the problems of complex systems. Point three reflects an overview of other evolutionary optimization algorithms, their mathematical models and areas of application.

In the **second chapter**, an extended self-adaptive CMA-ES algorithm for optimization on constrained domains is proposed. The algorithm takes into account the constraints and uses only single-peak distributions and objective functions. It is efficient for many problems, but may be inefficient for functions with a large number of extrema. In general, the modified algorithm is promising for optimization problems with complex constraints on the behavior of the objective function. Also, in the second chapter, an improved self-adaptive CMA-ES algorithm is developed for optimizing the parameters of complex systems, in particular

neural networks. The algorithm differs from classical ones in that it dynamically adjusts the number of peaks in the mixture of distributions during execution, avoiding unnecessary calculations.

The main characteristics and advantages of the algorithm:

- self-adaptability: the algorithm adapts to a specific task by changing the dimension of the mixture of distributions.

- versatility: can be used with different types of distributions and applied to other evolutionary algorithms. At the same time, only single-peakedness is required from the basic distribution.

- efficiency: avoids unnecessary calculations of the objective function.

- reasonableness: adaptability is based on methods of determining the optimal number of clusters in cluster analysis.

The modification of the algorithm for problems with restrictions on the search area is considered. It takes into account the constraints and uses only single-peak distributions. This modification is effective for many problems, but may be inefficient for functions with a large number of extrema.

In general, the proposed self-adaptive CMA-ES algorithm is a promising tool for efficient optimization of complex systems with both standard and restricted search domains.

In general, the extended CMA-ES algorithm is an effective tool for the optimization of complex nonlinear and non-convex problems. It has a number of advantages over the classic CMA-ES, such as:

- the ability to take into account multimodal objective functions;

- increasing the chances of finding the global optimum in comparison with classical heuristic algorithms;

- more effective search for hyperparameters of complex systems.

However, the extended CMA-ES algorithm has one significant drawback – the need to specify the dimension of the mixture. This parameter can affect the efficiency of the algorithm, because too small a dimension can lead to the fact that the algorithm will not be

able to take into account all extrema of the objective function, and therefore leads to slow convergence. On the other hand, too large a dimension can increase the running time of the algorithm.

Overall, the extended CMA-ES algorithm is a powerful optimization tool that can be useful for solving problems with multimodal objective functions.

In **the third chapter**, deals with the problem of finding optimal control for dynamic systems described by the stochastic Ito differential equation with Poisson disturbances and Markov switches. This section does not present sufficient conditions for the synthesis of optimal control for a stochastic dynamic system of a random structure with Poisson disturbances and Markov switches, and the control is aimed at stabilizing the system by probability. Sufficient conditions for asymptotic stability in terms of probability are obtained using the second Lyapunov method, in which the construction of the corresponding Lyapunov functions plays an important role. The general scheme of the study of the problem of optimal stabilization for stochastic differential equations of a random structure is also given.

For linear systems with a quadratic quality function, a method of synthesis of optimal control based on the solution of Riccati equations has been developed. Also, for autonomous systems, a system of differential equations was built to obtain unknown matrices, which are used to specify optimal control. Another approach to the synthesis of optimal control is based on small parameter methods with sequential determination of optimal control coefficients. This approach offers a new solution to the optimal control problem for a stochastic dynamic system with a random structure and Markov switches. In addition to theoretical achievements, the third chapter deals with a comparative analysis of classical heuristic algorithms for finding optimal controls for different values of the horizon of the optimization problem. A number of advantages and disadvantages of the new extended CMA-ES algorithm are noted, the main of which relate to the number of calls to the objective function and the accuracy of suboptimal solutions.

The main results of the dissertation research are summarized in the conclusions. The appendix presents scientific publications that reflect the main scientific results of the work.

Theoretical significance. The results of the work obtained in the course of scientific research are an improvement of the theory of evolutionary algorithms. The approaches proposed in the dissertation can be used for further research in this field, in educational courses of the Department of Applied Mathematics and Information Technologies of the Yuriy Fedkovich Chernivtsi National University (and other higher educational institutions), related to intellectual data analysis, machine learning, in methodological developments , teaching aids for the educational process and research work of students and graduate students.

Practical meaning. Algorithms developed in the dissertation can be used in the future for practical research of complex networks. The main results of the work are based on the statements of the theory of probability, the theory of machine learning, random processes and the Monte Carlo method.

The obtained results of the dissertation enrich the methods of evolutionary algorithms and the theory of machine learning, in particular the methods and approaches of solving the problem of hyperparameter optimization of neural networks and other complex systems. The practical value of the work is illustrated by the estimation of optimal control for stochastic differential Ito systems of random structure. The results of the work will be used in further research on this topic.

Keywords: neural networks, machine learning, mixture of distributions, optimization of hyperparameters, covariance matrix, CMA-ES algorithm, modeling, clustering algorithms, optimization problem, genetic algorithm, particle swarm method, random environment, random walk, system of stochastic differential equations, , Markov switching.

СПИСОК ПУБЛІКАЦІЙ ЗА ТЕМОЮ ДИСЕРТАЦІЇ

Наукові праці у періодичних наукових виданнях, проіндексованих у наукометричній базі даних Scopus:

1. Lukashiv T., Litvinchuk Y., Malyk I., Golebiewska A., Nazarov P. Stabilization of Stochastic Dynamical Systems of a Random Structure with Markov Switches and Poisson Perturbations. Mathematics. 2023. Vol. 11. Iss. 3. P. 1-22. (Scopus) (Q2 – URL:https://www.scimagojr.com/journalsearch.php?q=21100830702&tip=sid&clean=0).

Наукові праці у виданнях, внесених до переліку наукових фахових видань України:

2. Літвінчук Ю.А., Малик І.В. Розширений алгоритм стратегії еволюції адаптації коваріаційної матриці. *Буковинський математичний журнал.* 2022. 10 (2). С. 137–143. URL: https://doi.org/10.31861/bmj2022.02.09.
3. Літвінчук Ю. А. Про одне узагальнення еволюційних алгоритмів. International Scientific Technical Journal «Problems of Control and Informatics». 2023. 68(6), с. 64–75. URL: DOI: https://doi.org/10.34229/1028-0979-2023-6-4.
4. Малик І.В., Літвінчук Ю.А. Про один підхід побудови самоадаптивних алгоритмів на основі сумішей розподілу. Буковинський математичний журнал. 2023. 11 (2). С. 183-189. URL: https://doi.org/10.31861/bmj2023.02.18.

Наукові праці, які засвідчують апробацію матеріалів дисертації:

5. Малик І.В., Літвінчук Ю.А. Адаптивний метод навчання обмеженої машини Больцмана з алгоритмом генерації та знищення нейронів. *Інформаційні технології: наука, техніка, технологія, освіта, здоров'я*: тези доп. XXVIII Міжнар. наук.-практ. конф. MicroCAD-2020. Ч. IV. Харків. 2020. С. 174. URL:https://science.kpi.kharkov.ua/wpcontent/uploads/2020/10/Tezi_chastina_4_2020.pdf

6. Малик І. В., Літвінчук Ю. А. Побудова еволюційної стратегії на основі сумішей. *Інформаційне суспільство: технологічні, економічні та технічні аспекти становлення (випуск 70*): матеріали Міжнародної наукової інтернет-конференції, 22-23 вересня 2022 р. Тернопіль, Україна, Переворськ, Польща. URL: http://www.konferenciaonline.org.ua/ua/article/id-656/
7. Літвінчук Ю. А. Порівняльний аналіз оптимізації гіперпараметрів нейронних мереж. *Прикладна математика та інформаційні технології.* Міжнар. наук. конф. присвяченої 60-річчю кафедри прикладної математики та інформаційних технологій (м. Чернівці, 22-24 вересня 2022 р.): Чернівецький нац. ун-т, 2022. С. 181-183. URL: https://archer.chnu.edu.ua/jspui/bitstream/123456789/5858/1/AMIT2022-Materials.pdf
8. Малик І. В., Літвінчук Ю. А. Алгоритм CMA-ES для оптимізації гіперпараметрів нейронної мережі. *Молодіжна наука заради миру та розвитку.* Міжнар. наук.-практ. конф. присвячена Всесвітньому дню науки 9–11 листопада 2022 р. Чернівці, Україна. 2022. С. 647-649. URL: https://archer.chnu.edu.ua/xmlui/handle/123456789/6979
9. Малик І. В., Літвінчук Ю. А. Моделювання розширеного алгоритму CMA-ES на базі сумішей нормальних розподілів. *Інформаційне суспільство: технологічні, економічні та технічні аспекти становлення* (випуск 80). Матеріали Міжнародної наукової інтернет-конференції, 19 вересня 2023 р. Тернопіль, Україна, Переворськ, Польща. URL: http://www.konferenciaonline.org.ua/ua/article/id-1258/

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ

Абревіатура		Трактування
NN	Neural network	Нейронні мережі
GA	Genetic algorithm	Генетичний алгоритм
PSO	Particle Swarm Optimization	Метод рою часток
ACO	Ant colony optimization	Мурашиний алгоритм
BPNN	Back Propagation Neural Network	Нейронна мережа із зворотнім поширенням помилки
TDNN	Time delay neural network	Нейронна мережа з часовою затримкою
ATDNN	Adaptive time delay neural networks	Адаптивна нейронна мережа з часовою затримкою
ES	Evolution strategies	Еволюційна стратегія
GI-ES	Gradient information evolutionary strategy	Градієнтна інформаційна еволюційна стратегія
ACM-ES	Adaptive Covariance Matrix Adaptation Evolution Strategy	Адаптивна коваріаційна матриця стратегії еволюції адаптації
PSA	Population size adaptation	Адаптація чисельності популяції
EA	Evolutionary algorithms	Еволюційні алгоритми

ВСТУП

Актуальність теми дослідження. Методи еволюційних алгоритмів широко використовуються у сфері прикладної математики і вважаються майбутнім комп'ютерного програмування та штучного інтелекту. Еволюційні алгоритми сприяють ефективним інженерним рішенням, досліджуючи складні простори проєктування. У математиці та інших галузях знань еволюційні алгоритми застосовуються для кластеризації, мультимодальної оптимізації, а також для вирішення задач розпаралелювання. У природничих науках дані алгоритми є корисними у таких задачах, як налаштування параметрів, підбір даних та оптимізація розв’язку. Еволюційні алгоритми сприяють науковим відкриттям, ефективно досліджуючи складні простори розв’язків та покращуючи моделі в таких галузях, як фізика, хімія, біологія, екологія, медицина, економіка тощо.

При розгляді вищезгаданих проблем основну роль відіграє оптимізація, що дозволяє розв’язувати прикладні задачі, зокрема задачі штучного інтелекту, машинного навчання та налаштування гіперпараметрів нейронних мереж.

Основні класичні методи еволюційних алгоритмів були розроблені в 50-х роках XX століття, але їхній розвиток прискорився в 70-х роках. У цей період було реалізовано кілька основних типів еволюційних алгоритмів, включаючи генетичні алгоритми та еволюційні стратегії. Пізніше було розроблено ряд нових методів, таких як нейронні мережі.

Задачами теорії еволюційних алгоритмів займалися та займаються багато видатних науковців. Основну увагу слід звернути на цикли робіт Nikolaus Hansen, Stephan Ostermeier, Andreas Auger, Konstantinos Varelas. Їхня робота сприяла тому, що методи еволюційних алгоритмів стали потужним інструментом для вирішення широкого спектру прикладних задач.

Основна увага в дисертаційній роботі зосереджена на побудові самоадаптивних алгоритмів за допомогою сумішей розподілів з використанням розширеного алгоритму еволюційної стратегії з адаптацією коваріаційної матриці (CMA-ES) для

оцінки параметрів складних систем. Даний алгоритм дозволяє знаходити швидше розв'язок оптимізаційної задачі $f(x) \to extr$, де функція $f: R^d \to R$, причому припускається, що розглянута задача оптимізації є безумовною, тобто $x \in R^d$. Ще одним завданням дисертаційного дослідження є розробка самоадаптивного алгоритму оптимізації розміру суміші для розширеного CMA-ES алгоритму.

Об'єктом дослідження є оптимізаційна задача пошуку оптимальних параметрів на основі еволюційних алгоритмів, включаючи оптимізацію складних систем.

Предметом дослідження є оптимізація складних систем та визначення оптимальних гіперпараметрів нейронних мереж.

Мета дослідження полягає у побудові самоадаптивних алгоритмів та конструюванні; у проведенні порівняльного аналізу результатів класичних алгоритмами та запропонованого алгоритму на основі методу Монте-Карло.

Методи дослідження. В основу дисертаційної роботи покладені методи еволюційних алгоритмів та методи інтелектуального аналізу даних, а саме: генетичні алгоритми; еволюційна стратегія, статистичні методи (обчислення статистичних характеристик); теорія матриць (побудова стохастичних матриць та дослідження їх спектральних характеристик); методи машинного навчання (кластерний аналіз); метод Монте-Карло.

Зв'язок роботи з науковими програмами, планами, темами. Тема дисертаційної роботи відповідає науковому напрямку кафедри прикладної математики та інформаційних технологій Чернівецького національного університету імені Юрія Федьковича. Дослідження дисертаційної роботи проведено під керівництвом доктора фізико-математичних наук, доцента Малика Ігоря Володимировича в рамках науково-дослідної теми кафедри «Математичне моделювання і числово-аналітичні методи дослідження динамічних та інформаційних процесів» (номер державної реєстрації 0102U006591).

Наукова новизна отриманих у роботі результатів полягає в тому, що для розв'язання оптимізаційної задачі складних систем уперше:

- описано розширення CMA-ES алгоритму за припущення багатопіковості розподілу хромосом в генетичному алгоритмі;

- запропоновано самоадаптивний алгоритм оптимізації розміру суміші для розширеного CMA-ES алгоритму, при якому мінімізується кількість звернень до цільової функції і таким чином підтверджує ефективність запропонованого алгоритму;

- використовуючи теорію сумішей та оцінку параметрів сумішей нормального розподілу, розроблено алгоритм для оцінки гіперпараметрів складних систем на основі розширеного CMA-ES алгоритму;

- за допомогою методу Монте-Карло показано, що запропонований алгоритм дає кращі результати для вибраної оптимізаційної задачі у порівнянні з рядом класичних еволюційних алгоритмів;

- використано розширений CMA-ES алгоритм для конструювання оптимального керування в задачах синтезу оптимального керування для системи диференціальних рівнянь із випадковими збуреннями.

Практичне значення отриманих результатів. Результати дисертаційної роботи є вагомим внеском у розвиток теорії еволюційних алгоритмів. Розроблені та апробовані запропоновані алгоритми можуть використовуватися для подальших досліджень у цій галузі, адже на сьогодні немає універсального способу проведення оптимізації складних мереж, зокрема нейронних. Цю проблему поставлено в дисертаційній роботі, простежено, що розроблений новий алгоритм пошуку оптимального розв'язку дає кращі результати порівняно з існуючими класичними еволюційними алгоритмами.

Практичне значення дисертаційної роботи полягає в тому, що розроблений самоадаптивний розширений CMA-ES алгоритм визначення оптимального значення

оптимізаційної задачі можна надалі використовувати для подальшого вивчення складних мереж та налаштування гіперпараметрів нейронних мереж.

Особистий внесок здобувача. У дисертаційній роботі зроблено значний внесок у розв'язання конкретних наукових завдань . Зокрема:

- запропоновано нову ідею модифікації класичного CMA-ES алгоритму, який має ряд переваг над класичними еволюційними алгоритмами пошуку рішення оптимізаційної задачі;
- показано, що запропонований алгоритм швидше збігається до оптимального значення у порівнянні з іншими генетичними алгоритмами;
- встановлено , що з ростом кількості піків для розширеного CMA-ES алгоритму кількість звернень до цільової функції спадає у порівнянні з класичними еволюційними алгоритмами (з використанням модуля ***cma*** для мови Python);
- запропоновано новий покращений самоадаптивний алгоритм CMA-ES, з акцентом на параметр, що визначає розмірність суміші нормальних розподілів;
- за допомогою методу Монте-Карло показано, що запропонований алгоритм пошуку оптимального значення цільової функції дає кращі результати для визначення у порівнянні з рядом класичних методів;
- встановлено, що розроблений розширений CMA-ES алгоритм знаходження оптимального розв'язку добре працює для конструювання оптимального керування в задачах синтезу оптимального керування для системи диференціальних рівнянь із випадковими збуреннями.

Апробацію матеріалів дисертації здійснено на таких конференціях:

1. XXVIII Міжнародна науково-практична конференція «Інформаційні технології: наука, техніка, технологія, освіта, здоров'я (MicroCAD)», MicroCAD-2020, м. Харків, 28–30 жовтня 2020 року.
2. Міжнародна наукова інтернет-конференція, м. Тернопіль, Україна – м. Переворськ, Польща, 22-23 вересня 2022 р.

3. Міжнародна наукова конференція «Прикладна математика та інформаційні технології», присвяченої 60-річчю кафедри прикладної математики та інформаційних технологій (22-24 вересня 2022 р., Чернівці).
4. Міжнародна науково-практична конференція «Молодіжна наука заради миру та розвитку», м. Чернівці, 9-11 листопада 2022 р.
5. Міжнародна наукова інтернет-конференція, м. Тернопіль, Україна – м. Переворськ, Польща, 19-20 вересня 2023 р.

Структура й обсяг дисертації. Дисертаційна робота є завершеним науковим дослідженням, загальним обсягом 165 сторінок (з них: 138 сторінки основного тексту, 27 сторінок – література). Дисертація складається зі вступу, трьох розділів з підрозділами, висновків, списку використаних джерел (210 позиції) та додатку (список публікацій здобувачки за темою дисертації).

РОЗДІЛ І. ОГЛЯД ІСНУЮЧИХ ДОСЛІДЖЕНЬ

Прикладні задачі та сценарії часто включають проблеми оптимізації, які характеризуються безліччю обмежень і цільовими функціями, що демонструють розривність, нелінійність, невипуклість і мультимодальність. Ці завдання оптимізації зазвичай є багатовимірними і включають поєднання типів змінних, таких як цілочисленні, дійсні, дискретні і двійкові, кожна з яких має різні діапазони значень, що вимагають нормалізації. Отже, простір пошуку розв’язку за своєю суттю є негладким. Еволюційні алгоритми - це клас алгоритмів стохастичного пошуку, які імітують процес природного відбору для знаходження розв’язку задач оптимізації та пошуку.

Застосування дарвінівських принципів для автоматизованого розв’язку задач було задумано у 1950-х роках ще до появи комп'ютерів. Дарвін у своїй теорії показав, що виживання організмів залежить від генетичної спадковості, що охоплює такі процеси, як розмноження, схрещування та мутацію. Це розуміння проклало шлях використанню еволюційної теорії Дарвіна в алгоритмах обчислювальної оптимізації, забезпечуючи природний підхід до пошуку розв’язків реальних завдань оптимізації [1].

Історія еволюційних алгоритмів розпочалася у 1950-х та 1960-х роках з кількох незалежних спроб використовувати процес еволюції в обчисленнях того часу, які розвивалися окремо протягом 15 років. У 1960-ті роки у різних місцях з'явилися три різні інтерпретації цієї ідеї. Лоуренс Фогель представив еволюційне програмування на початку 1960-х років [2], ставши піонером в інтеграції еволюційних принципів у комп'ютерні програми. Ця концепція отримала значну підтримку з появою генетичних алгоритмів Джоном Холландом у 1975 році [3], надавши нову парадигму оптимізації, натхненну природним відбором та генетикою. Стратегія еволюції була створена на початку 1960-х років і отримала подальший розвиток у дослідженнях Інго Рехенберга, Ганса-Пауля Швефеля та їхніх колег [4]. Четверта гілка – генетичне програмування –

виникла на початку 1990-х років [5]. Четверта інтерпретація, відома як генетичне програмування, з'явилася на початку 90-х років.

Методи еволюційних алгоритмів нині використовуються у широкому спектрі; вважається, що це майбутнє як комп'ютерного програмування загалом, так і штучного інтелекту, зокрема в інженерній справі еволюційні алгоритми відіграють життєво важливу роль [6]. Вони застосовуються для таких завдань, як структурне проєктування, оптимізація параметрів та керування системою. Еволюційні алгоритми сприяють ефективним інженерним рішенням, досліджуючи складні простори проектування, покращуючи продуктивність продукту та вирішуючи завдання багатокритеріальної оптимізації.

В інформатиці та математиці еволюційні алгоритми застосовуються для кластеризації, мультимодальної оптимізації, також для розпаралелювання генетичних алгоритмів та генетичного програмування.

Еволюційні алгоритми у природничих науках допомагають у таких завданнях, як налаштування параметрів, підбір даних та оптимізація експериментальних планів. Еволюційні алгоритми сприяють науковим відкриттям, ефективно досліджуючи складні простори розв’язків та покращуючи моделі в таких галузях, як фізика, хімія, біологія, екологія тощо.

Вагомий внесок еволюційних алгоритмів і у медицині. Вони відіграють важливу роль в оптимізації різних аспектів охорони здоров'я, включаючи відкриття ліків, протоколи лікування, геномний аналіз, персоналізовану медицину, планування клінічних досліджень, діагностику захворювань та реабілітацію. Ці алгоритми сприяють більш ефективним та персоналізованим медичним втручанням, розподілу ресурсів та моделюванню результатів прогнозів захворювань. Їх адаптивність та можливості оптимізації роблять їх цінними інструментами для просування медичних досліджень та покращення догляду за пацієнтами.

Еволюційні алгоритми у фінансах та економіці використовуються для оптимізації, управління ризиками та процесів прийняття рішень. Вони допомагають

оптимізувати портфель, прогнозувати ринкові тенденції, оцінювати ризики та покращувати торгові стратегії. Ці алгоритми сприяють більш ефективному фінансовому моделюванню, розподілу ресурсів та адаптивному прийняттю рішень у динамічному економічному середовищі.

Еволюційні алгоритми знаходять застосування і в інших сферах досліджень, таких як робототехніка, ігри та обробка зображень. Таким чином, еволюційні алгоритми мають практичне застосування у широкому спектрі галузей. Їхня універсальність і здатність вирішувати складні завдання оптимізації роблять їх цінними інструментами в різних галузях науки.

1.1. Генетичні алгоритми пошуку оптимальних параметрів нейронних мереж

Нейронні мережі (NN) та генетичні алгоритми (GA) – це два складні методи машинного навчання, які нині привертають увагу науковців, інженерів, програмістів, статистиків тощо. В останні роки вони набули великої популярності. І, як випливає з назви, нейронні мережі імітують мозок чи механізми обробки біологічної інформації.

З'явилися численні методи штучного інтелекту на основі реальних біологічних систем, а саме нейронних мереж, еволюційних обчислень, алгоритму імітації відпалу та ройового інтелекту, які базувалися на роботі біологічних нервових систем, природного відбору, принципу термодинаміки та поведінки комах відповідно. Незважаючи на обмеження, пов’язані з кожною з цих згаданих методик, вони є надійними та застосовуються для вирішення реальних життєвих проблем у сферах науки, техніки, бізнесу та комерції. У результаті гібридизації двох або більше даних методів на сьогоднішній день створено багато ефективних інтелектуальних систем [7].

Нещодавні дослідження, в яких гібридизовані методи обчислювального інтелекту пошуку оптимальних або майже оптимальних розв’язків включаються, але

не обмежуються: генетичний алгоритм, метод рою часток (PSO) і мурашиний алгоритм (ACO) у [8]; нечітка логіка та інтеграція експертних систем у [9]; поєднання PSO, хаотичного та гауссового локального пошуку у [10]; у [11] поєднання нейронних мереж і нечіткої логіки; у [12] гібридизація GA та PSO; у [13] поєднання механізму нечіткого виведення, онтологій і мови нечіткої розмітки (FML – Fuzzy Markup Language); у [14] гібридизація машини опорних векторів (SVM – Support Vector Machine) і PSO.

Проте нейронні мережі та генетичний алгоритм вважаються найбільш надійними та перспективними методами обчислювального інтелекту. Останнім часом нейронні мережі виявилися потужним і практичним інструментом для моделювання складних і нелінійних систем [15, 16, 17, 18, 19]. GA та NN привертають увагу [20, 21] комп'ютерників та інженерів. Ця увага пояснюється прогресом у розумінні природи та динамічної поведінки даних методів. Крім того, зрозуміло, що гібридизація цих методів може бути застосована для вирішення складних задач [20] та як інструмент для машинного навчання [22].

NN дуже чутлива до параметрів [23, 24], які можуть вплинути на її продуктивність. Оптимізовані NN в основному визначаються трудомісткими методами проб і помилок, які включають деструктивне і конструктивне проєктування [23, 25, 26]. NN дуже схильні до перенавчання - і різні типи NN, які навчаються і тестуються на тому самому наборі даних, можуть давати різні результати, що порушує надійність нейронної мережі [27].

На продуктивність GA впливають такі чинники, як розмір популяції, вибір батьків, швидкість схрещування, частота мутацій та кількість поколінь [21]. Вибір відповідних значень параметрів GA здійснюється шляхом громіздких спроб та помилок, що займає багато часу [28], оскільки немає системи вибору оптимальних значень цих параметрів [29]. Подібно до вибору значень параметрів GA, проектування NN залежить від проблемної області [21].

Оптимізаціїя NN стає можливою за рахунок використання сильних сторін NN і GA та усунення їх обмежень [30]. Експериментальні дані в літературі показують, що оптимізація нейронних мереж за допомогою генетичного алгоритму сходиться до оптимального розв'язку [31, 32] за менший обчислювальний час [29, 31, 32, 33, 34, 35, 36, 37, 38, 39], ніж звичайні NN [37]. Тому оптимізація NN із використанням GA ідеальна, оскільки усуває недоліки дизайну NN, роблячи її ефективнішою.

GA добре працює з NN у пошуках кращої моделі та апроксимації параметрів для підвищення їх ефективності [40, 41, 42, 43]. GA можна використовувати для оптимізації ваг, топології, вибору функцій та навчання. Проблема проєктування нейронних мереж полягає у виборі оптимальних конфігурацій для пошуку розв'язку у конкретній області. Вибір топології NN вважається дуже важливим аспектом, оскільки неефективна топологія NN приведе до падіння NN у локальні мінімуми. Проблема вибору відповідних архітектурних конфігурацій та оптимальних ваг NN є складним завданням у галузі проєктування нейронних мереж [44]. Оптимальна кількість шарів і нейронів у прихованих шарах визначатиметься розробником NN, тоді як чіткої теорії вибору відповідної кофігурації параметрів не існує. GA широко використовується в різних проблемних областях для автоматичного проєктування NN-топології, щоб відійти від проблем, пов'язаних з їх розробкою, та покращити продуктивність і надійність NN.

Топологія NN, як вона визначена в [45], являє собою швидкість навчання, кількість епох, кількість прихованих шарів, кількість нейронів; (вхідні нейрони та вихідні нейрони), частоту помилок, коефіцієнт поділу наборів даних навчання, перевірки та тестування.

Є багато робіт з оптимізації топології NN у різних галузях. Наприклад, у [46] автори оптимізували топологію NN і навчили її з використанням GA, а потім створили класифікатор для класифікації раку молочної залози. Аналогічно, у [47] оптимізували топологію рекурентної нейронної мережі на основі GA-пошуку та побудували модель для класифікації граматичного висновку. Крім того, у [48] використовували GA для

вибору оптимальної топології FNN (Feedforward neural network – нейронна мережа прямого прямого поширення) і розробили модель для прогнозування денного курсу французького франка. У роботі [45] GA використовується для вибору відповідних підмножин функцій, а потім оптимізує топологію нейронної мережі з метою створення моделі NN для розпізнавання долоні. У [49] GA використовувався для оптимізації топології BPNN, як модель прогнозування щільності нанорідин.

Кім та інші [50] використовували GA для отримання оптимальної топології BPNN та розробили модель оцінки вартості будівництва оселей. У [51] оптимізували підмножини функцій, кількість тимчасових затримок та топологію TDNN на основі пошуку GA. Модель TDNN була побудована для виявлення часових закономірностей на фондових ринках. У [52] автори повторили аналогічне дослідження з використанням ATDNN, а GA використовували для оптимізації кількості тимчасових затримок та топології ATDNN. Результат, отриманий за допомогою моделі ATDNN, виявився кращим, ніж результат TDNN у більш ранньому дослідженні, проведеному в [51].

Система виявлення несправностей була розроблена з використанням NN, топологія якої оптимізована на основі пошуку GA [53]. Система ефективно виявляла несправності у гідропонній системі [53]. В іншому дослідженні оптимальна топологія NN була отримана шляхом пошуку GA для побудови моделі. Згодом модель використовувалася для прогнозування вартості продукції машинобудівної промисловості Тайваню [54]. У дослідженні [55] вихідна архітектурна структура NN була створена за допомогою методу к-найближчих сусідів на першому етапі процесу побудови моделі. Потім був застосований GA для розпізнавання та усунення надлишкових нейронів у структурі, щоб утримати середньоквадратичну помилку ближче до потрібного порогу. На третьому етапі модель використовувалася для апроксимації нелінійної функції $sinc$ та ідентифікації нелінійної динамічної системи [55]. У [56] запропоновано гібридну модель, що поєднує NN, GA та нечітку логіку.

GA використовувався для оптимізації топології NN та побудови моделі розпізнавання голосу. Модель змогла правильно ідентифікувати іспанські слова.

Автори [34] використовували GA для оптимізації топології NN, а також для навчання NN побудови моделі. Модель використовувалася для прогнозування небезпеки самозаймання у вугільних пластах для видобутку корисних копалин. У [57] GA використовувався для генерації правил із SVM для побудови наборів правил з метою підвищення його інтерпретованості.

Вибір відповідного та найбільш релевантного набору вхідних ознак є важливою проблемою у процесі моделювання нейронної мережі та інших класифікаторів [58]. Існує кілька методів зменшення розмірів вхідного простору, включаючи кореляцію, індекс Джині та аналіз основних компонентів. Як зазначено в [59], GA працює краще, ніж згадані методи, за точністю вибору ознак.

GA застосовують [60] для вибору відповідних характеристик забруднювачів пальмової олії в якості вхідних даних для предиктора NN. Предиктор використовувався для контролю викидів на заводах із виробництва цього продукту. Бем та ін. [61] використовували GA для вибору підмножини вхідних ознак та побудови NN-класифікатора для ідентифікації когнітивних функцій мозку. У [62] на першому етапі використовувався GA для вибору генів з набору даних мікрочіпа. На другому етапі був застосований GA для оптимізації топології NN у процесі створення класифікатора для класифікації генів.

Аналогічним чином [28] GA застосовувався для вибору набору навчальних даних для моделювання NN. У цій методології вибір навчальних наборів даних для побудови моделі NN досягається за допомогою методу єдиного проєктування. Процес включає апроксимацію функції граничного стану за допомогою навченої моделі NN і подальшу оцінку ймовірності відмови за допомогою генетичних алгоритмів. Помітно, що інтеграція єдиного методу проєктування з NN-GA ефективно усуває властиві неточності при виборі навчальних наборів даних, зберігаючи при цьому переваги GA.

У дослідженні [63] топологія MLP була інтегрована у процес навчання на основі GA для пошуку оптимального класифікатора MLP (Multi-Layer Perceptron – багатошаровий персептрон) для класифікації сейсмічних сигналів. Методологія, що акцентує увагу на виділенні ознак і класифікації сигналів, є цінним інструментом для вивчення та потенційного аналізу інших діючих вулканів у регіоні. [64] використовували GA для вибору підмножини вхідних ознак для NN-предиктора і використовували предиктор для аналізу даних, виміряних датчиком.

У дослідженні [65] автори запропонували двоетапний метод розробки моделі розподілу витрат, що базується на гібридизації нейронної мережі та генетичного алгоритму. На першому етапі застосовувався GA для вибору підмножини вхідних об'єктів. На другому етапі GA використовувався для оптимізації топології NN та побудови моделі розподілу витрат. GA застосовувався для вибору підмножини вхідних ознак для побудови моделі GRNN (General regression neural network – Нейронна мережа узагальненої регресії) [66]. Модель була успішно застосована для прогнозування реакції кристалічності лактози, вмісту вільного жиру та середнього розміру частинок сухого молочного продукту.

Автори [67] використовували GA для вибору підмножин вхідних ознак для побудови моделі процесу ферментації. Модель застосовувалася для прогнозування майбутньої концентрації продукту та ефективності ферментації. Пракашам та ін [68] використали GA для зменшення вхідних розмірностей та побудови NN-предиктора. Предиктор застосовувався для прогнозування оптимального виходу біоводню.

Такеда та ін [69] використовували GA для оптимізації ваг NN і вибору підмножин вхідних ознак у побудові системи розпізнавання банкнот NN. Система тестувалася для розпізнавання десяти різних банкнот (японська єна, долари США, німецькі марки, бельгійські франки, корейська вона, австралійські долари, британські фунти, італійські ліри, іспанські песети та французькі франки). Встановлено, що було досягнуто точності розпізнавання понад 97%.

У дослідженні, проведеному у [70] GA, моделі нечіткого та лінійного перетворення застосовувалися для вибору підмножин ознак, які є вхідними даними для NN. Крім того, GA використовувався для оптимізації ваг NN шляхом навчання побудови предиктора. Предиктор використали для прогнозування моделі фондового ринку.

1.1.1. Основні концепції генетичного алгоритму

Ідея GA була задумана [71] як метод пошуку, заснований на принципі природного відбору та природної генетики [72, 73, 74, 75]. Поштовхом до вивчення принципів еволюції та застосування отриманих знань для розробки алгоритмів стали процеси відбору біологічних генетичних систем [76].

Генетичний алгоритм належить до сімейства еволюційних алгоритмів, які є методами обчислювального пошуку. Вони моделюють еволюцію певних об'єктів, зібраних у популяцію [76]. Ці окремі сутності можуть бути чим завгодно: цифровими (тобто штучними) організмами, бітовими рядками, комп'ютерними програмами, фінансовими стратегіями тощо. Оскільки еволюція впливає на сутності, наступні версії популяції, так звані покоління, розвиваються, щоб максимізувати свою пристосованість, тобто здатність вижити в навколишньому середовищі або для досягнення поставлених цілей, або для максимізації продуктивності. У комп'ютерній програмі весь цей процес еволюції починається з генетичного представлення: **ген, хромосома, алель, фенотип та генотип**.

Робота генетичного алгоритму полягає у моделюванні дії біологічного гена. Його завдання – накласти певну операцію на пристосованість особини, щоб досягти еволюції виживання найбільш пристосованих. З точки зору оптимізації пошуку, генетичні операції дозволяють поступово оптимізувати розв'язок задачі та наблизитися до оптимального розв’язку. Генетичні операції включають такі три основні генетичні оператори: **відбір, схрещування та мутацію**.

Операція відбору є процесом визначення життєздатної особини у групі до створення нової групи. Основні інструкції щодо побудови GA формують ген (бітові рядки довільної довжини). Послідовність генів називається хромосомою. Можливе вирішення проблеми може бути описане генами, але насправді не є відповіддю на проблему. Найменша одиниця хромосом називається алеллю, представленою одним символом або двійковим бітом. Фенотип дає зовнішній опис особинам, тоді як генотип є інформацією, що зберігається в хромосомі [77], як показано на рис. 1.1, де $F1, F2, F3, F4 \ldots Fn$ та $G1, G2, G3, G4 \ldots Gn$ – фенотипи та гени відповідно.

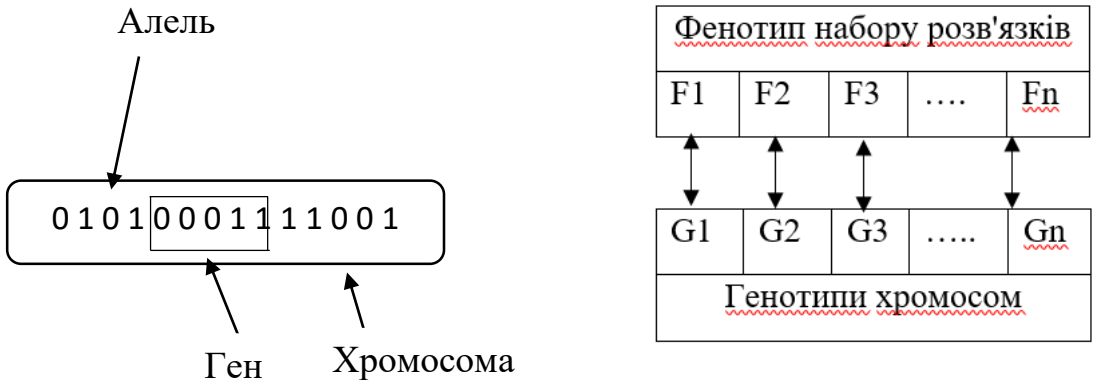


Рис. 1.1. Представлення алелі, гена, хромосоми, генотипу та фенотипу

Особини в групі утворюють популяцію, як показано в таблиці 1 [77]. Оцінюється пристосованість кожної особини в популяції. Особини з вищою пристосованістю дають більше потомства, ніж особини з нижчою пристосованістю. Особини та певна інформація про простір пошуку визначаються параметрами фенотипу.

Таблиця 1.1.

Приклад популяції

	Індивідуум	Код
	Хромосома 1	0 1 1 0 1 0 1 1
Популяція	Хромосома 2	1 1 0 0 0 0 1 0
	Хромосома 3	1 0 1 0 0 0 1 0
	Хромосома 4	0 1 1 0 1 0 1 0

Операція схрещування стосується обміну деякими генами між двома хромосомами, які певним чином схрещуються одна з одною, утворюючи двох нових особин. Це основний метод генерації нових особин, що визначає можливості глобального пошуку генетичних алгоритмів та відіграє у них ключову роль. Метою схрещування є відтворення кращих хромосом, що містять «хороші» частини старих хромосом, як показано на рис. 1.2. [78]

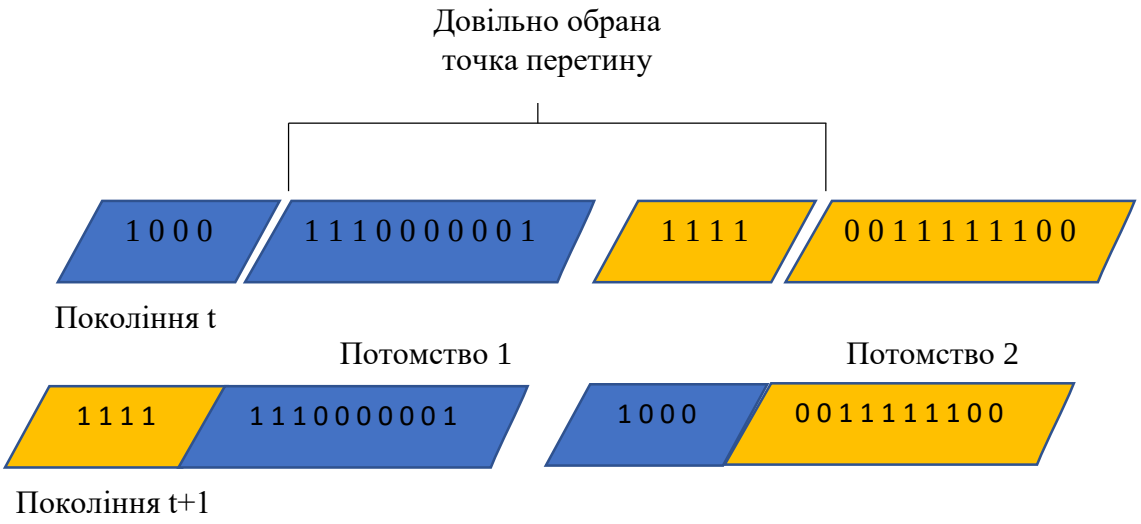


Рис. 1.2. Операція схрещування

Виживання деякого сегмента старої популяції у наступному поколінні можливе завдяки процесу відбору у схрещуванні [77].

Операція мутації визначає заміну значень генів у певних локусах хромосомного кодуючого рядка індивідуума іншими алелями в цьому локусі з метою формування нової особини. Це допоміжний метод генерації нових особин, що визначає можливості локального пошуку генетичних алгоритмів. Оператор мутації та оператор

схрещування взаємодіють один з одним з метою завершення глобального та локального пошуку. Таким чином генетичний алгоритм завершує процес оптимізації.

Це створення потомства від одного батька шляхом інвертування одного або кількох випадково вибраних бітів у хромосомах батька, як показано на рисунку 1.3. Мутація може бути досягнута на будь-якому біті з невеликою ймовірністю, наприклад 0,001 [79]. Рядки, отримані в результаті схрещування, змінюються, щоб уникнути локального мінімуму. Генетичні матеріали, які можуть бути втрачені в процесі схрещування та спотворення генетичної інформації, закріплюються шляхом мутації. Імовірність мутації $p_m\,(x\,)$ відповідає за визначення того, наскільки часто буде ділення хромосоми, що піддається мутації.

Якщо мутація не застосована, нащадки генеруються негайно в результаті схрещування без будь-якої частини хромосом, що підлягає зміні. 100-відсоткова ймовірність мутації означає, що вся хромосома буде змінена, але якщо ймовірність дорівнює 0%, це означає, що жодна з частин хромосоми не буде спотворена [77]. На рисунку 1.3 показано мутацію для двійкового представлення [78].

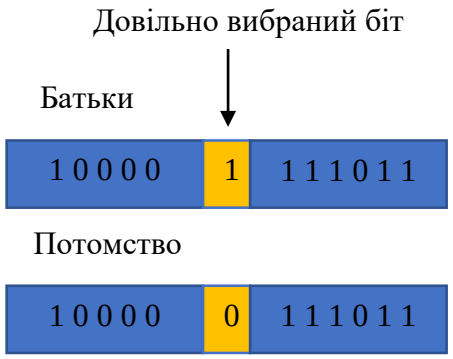


Рис. 1.3. Операція мутації

1.1.2. Математичне представлення генетичного алгоритму

Математичне формулювання [79, 80, 81, 82, 83, 84] генетичного алгоритму включає опис ключових компонентів і операцій алгоритму в математичних термінах. Математично GA можна описати наступним чином:

1. Ініціалізація:

- популяції P, що складається з n особин, кожна з яких є потенційним розв'язком оптимізаційної задачі

$$f(x) \to extr \tag{1.1}$$

де $f: R^d \to R, x \in R^d$

- Популяцію можна представити як:

$$P = \{x_1, x_2, \dots, x_n\}, \tag{1.2}$$

де $x_i -$ особини в популяції, тобто хромосоми генетичного алгоритму.

2. Оцінка пристосованості:

- функція пристосованості (цільова функція) $f(x)$ кількісно визначає, наскільки добре особа справляється із задачею оптимізації, тобто призначає значення фітнесу кожному індивіду на основі його пристосованості.

3. Відбір:

- відбір – це процес вибору особини із поточної популяції для створення пулу схрещування наступного покоління. Імовірність відбору особини можна розрахувати, виходячи з її пристосованості:

$$P(x_i) = \frac{f(x_i)}{\sum_{i=1}^{n} f(x_i)}$$

4. Схрещування:

- схрещування поєднує генетичну інформацію двох батьківських особин для створення нащадків. Одноточкове схрещування можна представити як:

$$\acute{x} = \left(x_{parent1}[1:k] \middle| x_{parent2}[k+1:]\right),$$

де $x_{parent1}$ та $x_{parent2}$ – батьки, k – точка перетину у рядку батьківського організму, а $\acute{x}$ – нащадок.

5. Мутація:

- мутація вносить невеликі випадкові зміни в генетичну інформацію особини з низькою ймовірністю. Це допомагає підтримувати генетичну різноманітність в популяції та запобігає передчасній збіжності алгоритму.

- Операцію мутації для окремого x можна визначити як:

$$\acute{x}_i = mutation(x_i)$$

6. Заміна:

- наступне покоління створюється шляхом заміни старої популяції нащадків на нову . Стратегія заміни може бути різною, але слабших особин часто замінюють сильнішими.

7. Критерії зупинки:

- генетичний алгоритм продовжує перебирати покоління доти, доки не будуть виконані певні критерії завершення. Загальні критерії включають досягнення максимальної кількості поколінь, пошук задовільного розв'язку чи перевищення обчислювальних ресурсів.

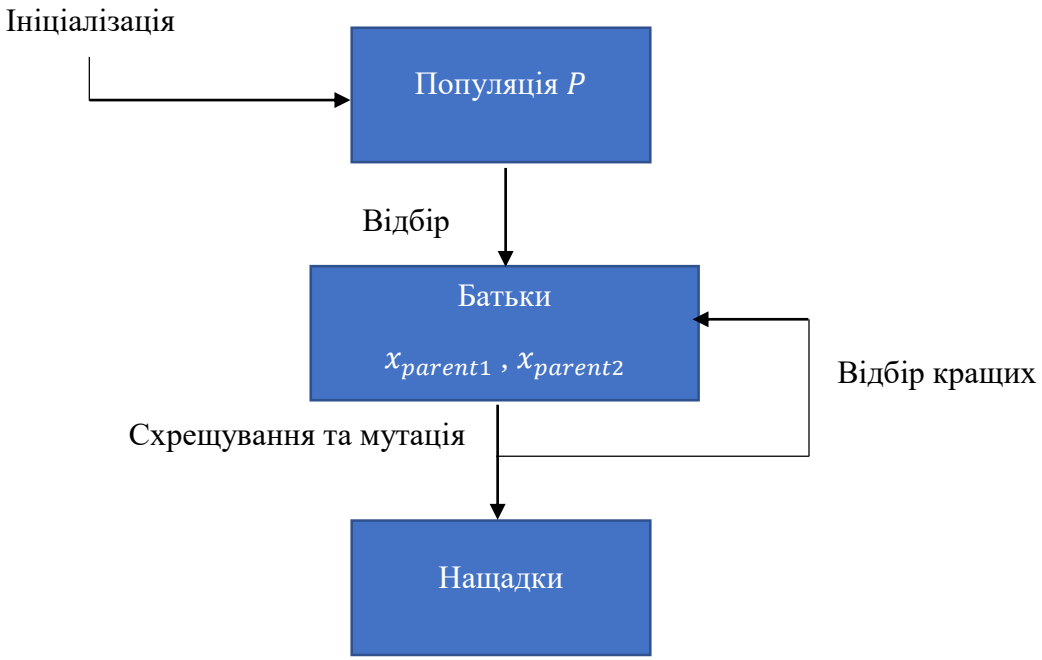


Рис. 1.4. Класичний дизайн генетичного алгоритму

Блок-схема, показана на рисунку 1.4, відображає процес генетичного алгоритму. Генетичні алгоритми особливо підходять для вирішення задач у великих важкодоступних просторах пошуку. Вони вимагають дуже мало припущень про властивості ландшафту пристосованості, добре справляються з існуванням локальних оптимумів чи множинністю екстремумів. GA досягли особливих успіхів у розвитку нейронних мереж, багатокритеріальної оптимізації та генетичному програмуванні. Вони більше, ніж просто оптимізатори, оскільки створюють новизну, безпосередньо пов'язані зі штучним життям, оскільки сприяють виникненню складної цифрової екології, а також чіткого зв'язку з відкритою еволюцією, яка може значно покращити моделювання еволюційних, адаптивних систем.

1.2. CMA-ES алгоритм і модифікації

Еволюційна стратегія (ES) – це метод оптимізації в інформатиці, що базується на принципах еволюції, призначений для неперервної оптимізації функцій.

На відміну від інших еволюційних алгоритмів, еволюційна стратегія не використовує жодних форм схрещування; натомість модифікація можливих розв’язків обмежується операторами мутації. Це робить еволюційну стратегію різновидом паралельного стохастичного сходження на гору [3].

Однією з ключових особливостей еволюційної стратегії є те, що вибір наступного покоління є детермінованим і ґрунтується виключно на рейтингах пристосованості, а не на фактичних її значеннях. Це означає, що отриманий алгоритм інваріантний щодо монотонних перетворень цільової функції [3].

Існує ряд версій та стандартна термінологія для узагальнення алгоритму ES:

- P – набір особин (популяція) заданий формулою (1.2);
- μ – кількість батьків, що вибираються на кожній ітерації;
- λ – кількість нащадків.

(μ, λ)-ES – варіант еволюційної стратегії, у якій нащадки замінюють батьків безпосередньо на кожній ітерації алгоритму.

$(\mu + \lambda)$-ES – варіант еволюційної стратегії, коли діти і батьки додаються до популяції. Поділ плюсом (+) параметрів вказує, що нащадки та батьки разом визначать сукупність для наступної ітерації.

Еволюційна стратегія відрізняється від генетичних алгоритмів тим, що особини кодуються векторами дійсних чисел, а при розмноженні батьки вибираються випадковим чином.

Підсумовуючи, можна сказати, що еволюційна стратегія – це потужний метод оптимізації, який використовує принципи природного відбору для ітеративного покращення можливих розв'язків задач оптимізації. Це цінний інструмент у галузі обчислювального інтелекту та машинного навчання, який знайшов застосування у різних галузях, включаючи глибинне навчання з підкріпленням[85].

1.2.1 Математична модель алгоритму CMA-ES

Еволюційна стратегія з адаптацією коваріаційної матриці (CMA-ES) – потужний алгоритм оптимізації, який використовується для вирішення складних, багатовимірних, нелінійних задач оптимізації. CMA-ES використовує еволюційну стратегію, натхненну принципами природного відбору та адаптації, спеціально розроблену для роботи у просторах неперервного пошуку. Алгоритм особливо популярний у галузі машинного навчання, оптимізації та робототехніки.

CMA-ES був спочатку представлений Ніколасом Хансеном та Андреасом Остермайєром наприкінці 1990-х років [86]. Він був розроблений як удосконалення традиційних стратегій еволюції за рахунок включення адаптивної матриці коварацій для кращої адаптації до форми простору пошуку. Цей адаптивний механізм має вирішальне значення для оптимізації складних, нелінійних та багатовимірних функцій.

CMA-ES працює, підтримуючи сукупність можливих рішень, де кожний розв’язок представлений як багатовимірний розподіл Гаусса із вектором середніх та матрицею коваріації. Ці параметри змінюються протягом поколінь, щоб покращити розв’язок. Таким чином алгоритм наближається до глобального оптимуму.

Алгоритм CMA-ES призначений для адаптації параметрів пошукового розподілу для покращення процесу пошуку. На кожній ітерації виважена комбінація найкращих кандидатів розв’язків використовується для оновлення параметрів розподілу [87]. Основний цикл алгоритму складається з трьох основних частин: вибір нових рішень, зміна порядку обраних розв’язків на основі їх придатності та оновлення внутрішніх змінних станів на основі перевпорядкованих вибірок.

CMA-ES спирається на модель розподілу популяції кандидатів (параметризований багатовимірний нормальний розподіл) та дослідження простору проєктування [88]. У кожному поколінні батько для наступного покоління визначається з використанням середньозваженого значення вибраних кандидатів з μ нащадків, створених у цьому поколінні з використанням (λ, μ)-вибору. Популяція наступного покоління генерується шляхом вибірки багатовимірного нормального розподілу з коваріаційною матрицею покоління n вище середнього значення m покоління $N(m^{(n)}, \left(\sigma^{(n)}\right)^2 C^{(n)})$ [89]. Розмір кроку $\sigma^{(n)}$ визначає загальну дисперсію мутації в поколінні n. Змінна властивість розміру кроку у кожному поколінні відіграє життєво важливу роль у контролі передчасної збіжності та близької збіжності до глобального оптимуму. На рисунку 1.5 [89] продемонстровано основні етапи роботи CMA-ES.

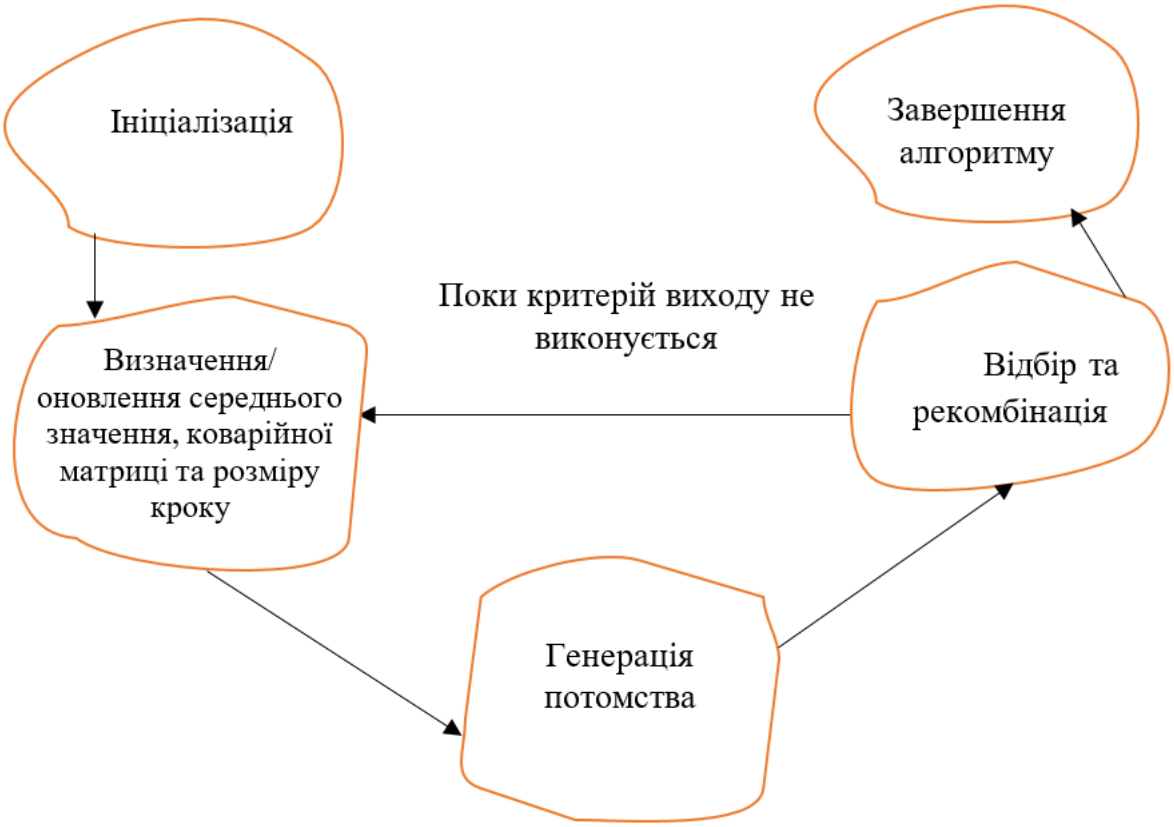


Рис. 1.5. Життєвий цикл CMA-ES

Дослідження глобального оптимуму у вимірах d за допомогою CMA-ES починається з генерації μ d-вимірних особин з оцінкою через вибірку багатовимірного нормального розподілу, зосередженого навколо середнього m у кожному поколінні . У загальному випадку рівняння можна записати так:

$$x_k^{(n+1)} \sim N\left(m^{(n)}, \left(\sigma^{(n)}\right)^2 C^{(n)}\right) \text{ для } k = 1, \dots, \lambda, \quad (1.3)$$

де $x_k^{(n+1)}$ є член вибірки, згенерований при $(n+1)$ генерації. Кожен член популяції є вектором і подається як

$$X_i = [x_1, x_2, x_3, \dots, x_d] \text{ для } i = 1,2,3 \dots, \mu \quad (1.4)$$

Щоб перейти до наступного покоління $(n+2)$ з рівняння (1.3), потрібно перерахувати параметри $m^{(n+1)}, C^{(n+1)}, \sigma^{(n+1)}$ [89]. Розглянемо кроки алгоритму адаптації коваріаційної матриці.

1. Відбір та рекомбінація

Кросоверний характер еволюційного процесу досягається шляхом обчислення вектора середніх для кожного покоління та подальшої зміни вектора середніх для створення нащадків. Вектор середніх $m^{(n+1)}$ для покоління n являє собою середньозважене значення μ обраних кращих особин у порядку ранжування пристосованості цільової функції з вибіркового простору $x_k^{(n+1)}$ для $k = 0,1,2, \dots \mu$. У випадку рівнозваженого вектора кожна отримана вибірка робить рівний внесок у результуюче середнє, при цьому вага, призначена кожній вибірці, є рівномірною ($w_i = 1/\lambda$). При використанні векторної конфігурації лінійної ваги w_i найбільш пристосована точка має більшу частку генів, ніж точка з нижчою пристосованістю [89] у результуючому середньому порівняно з точкою з нижчою придатністю.

$$\sum_{i=1}^{\lambda} w_i = 1,\ w_1 > w_2 > w_3 > \cdots > w_\lambda > 0 \text{ для } i = 1,2, \dots \lambda$$

2. Адаптація коваріаційної матриці

Коваріаційна матриця адаптації визначає різні мутації популяції нащадків у процесі еволюції [90]. Нащадок у поколінні відбирається відповідно до багатовимірного нормального розподілу в R^d, в той час як рекомбінація зводиться до вибору нового середнього значення в поколінні $n + 1$, мутація зводиться до вибірки нормального розподілу коварійної матриці, помноженої на розмір кроку навколо середнього. Залежності між змінними у розподілі представлені коваріаційною матрицею. Адаптація коваріаційної матриці – це метод оновлення коваріаційної матриці цього розподілу, який є вивченням моделі другого порядку базової цільової функції. Середнє значення розподілу оновлюється таким чином, щоби ймовірність успішного розв'язку-кандидата була максимальним, а матриця коваріації оновлюється таким чином, щоби ймовірність раніше успішних кроків пошуку збільшувалася [89].

$$C^{(n+1)} = \left(1 - c_1 - c_\mu\right)C^n + c_1 P_c^{(n+1)} P_c^{(n+1)^T} + c_\mu \times \sum_{i=1}^{\mu} w_i \left(\frac{X_{1:\lambda}^{(n+1)} - m^n}{\sigma^{(n)}}\right)\left(\frac{X_{1:\lambda}^{(n+1)} - m^n}{\sigma^{(n)}}\right)^T \quad (1.5)$$

3. Контроль розміру кроку

Наведена вище адаптація коваріаційної матриці контролює загальний масштаб розподілу, тобто розмір кроку. Коваріаційна матриця збільшує масштаб лише в одному напрямку для кожного вибраного кроку і може зменшувати масштаб тільки за рахунок зникнення старої інформації за допомогою меншого коефіцієнта інформації $1 - C_1 - C_\mu$. Щоб контролювати розмір кроку $\sigma^{(n)}$, ми використовуємо шлях еволюції. Щоразу, коли шлях еволюції короткий, окремі кроки скасовують один одного. І тут розмір кроку слід зменшити. Якщо шлях еволюції довгий та окремі кроки спрямовані в одному напрямку, розмір кроку слід збільшити, аби зменшити кількість кроків.

Щоб вирішити, чи є шлях еволюції довгим чи коротким, довжина шляху порівнюється з очікуваною довжиною при випадковому відборі. Якщо шлях відбору робить шлях еволюції довшим, ніж очікувалося, то σ збільшується, а якщо навпаки, то σ зменшується [89].

Узагальнення етапів алгоритму CMA-ES наведено нижче [91]:

Крок 1: Ініціалізація параметрів CMA-ES
$d \leftarrow$ Розмірність
$\lambda \leftarrow$ Чисельність потомства$(4.0 + 3.0\, Log(d))$
μ ← Популяція батьків для наступного покоління $\left(floor(\lambda/2)\right)$
$\sigma_{Start} \leftarrow$ Початкове середньоквадратичне відхилення
$c_{cov} \leftarrow$ Швидкість навчання коваріації
Крок 2: **While** критерій зупинки не дотримано **do**
Крок 3: Оновлення коваріаційної матриці $C^{(n+1)}$

$$C^{(n+1)} \leftarrow (1 - c_{cov})C^n + \frac{c_{cov}}{\mu_{cov}} P_c^{(n+1)} P_c^{(n+1)^T} + c_{cov}\left(1 - \frac{1}{\mu_{cov}}\right) \times \sum_{i=1}^{x} w_i \left(\frac{X_{1:\lambda}^{(n+1)} - m^n}{\sigma^{(n)}}\right)\left(\frac{X_{1:\lambda}^{(n+1)} - m^n}{\sigma^{(n)}}\right)^T$$

Крок 4: Оновлення розміру кроку σ^n

$$\sigma_{n+1} \leftarrow \sigma_n \times exp\left(\frac{c_\sigma}{d_\sigma}\left(\frac{\|P_\sigma\|}{E\,\|N(0,I)\|} - 1\right)\right)$$

Крок 5: Створення вибіркової сукупності для генерації $n+1$

$$x_k^{(n+1)} \sim n\left(m^{(n)}, \left(\sigma^{(n)}\right)^2 C^{(n)}\right) \quad \text{для } k = 1, \dots, \lambda$$

Крок 6: Оновлення середнього значення для генерації $n+1$

$$m^{(n+1)} \leftarrow \sum_{i=1}^{\mu} w_i x_{i:\lambda}^{(n+1)}$$

Крок 7: Оновлення найкращого розв'язку

Крок 8: Завершення алгоритму **While**

Алгоритм CMA-ES складається з трьох основних кроків, як описано вище. Для будь-якого алгоритму критерій зупинки має бути невід'ємною частиною, а для CMA-ES його можна вказати такими способами:

1) Запуск алгоритму для заздалегідь певної кількості поколінь.

2) Зупинка подальшого виконання, коли алгоритм не наближається до розв'язку.

3) Зупинка, коли алгоритм досягає наперед визначеного значення цільової функції.

Популяція нової точки пошуку при генерації $(n+1)$ генерується шляхом вибірки багатовимірного нормального розподілу із середнім $m^{(n)}$ за допомогою рівняння (1.3).

Нове середнє значення шуканого розподілу для покоління $n+1$ є середньозваженим значенням кращих нащадків вибраних з λ, отриманих за допомогою рівняння (1.3).

$$m^{(n+1)} = \sum_{i=1}^{\mu} w_i x_{i:\lambda}^{(n+1)} \tag{1.6}$$

$$\sum_{i=1}^{\mu} w_i = 1, \quad w_i > 0 \ \text{ для } \ w_i = 1, \dots, \mu \tag{1.7}$$

Окремі ваги для вектора ваг визначаються на основі μ_{eff} так, що $1 < \mu_{eff} < \mu$. У нашому випадку ми вибираємо $\mu_{eff} \approx \lambda/4$, що є розумним параметром w_i.

$$\mu_{eff} = \left(\sum_{i=1}^{\mu} {w_i}^2\right)^{-1} \tag{1.7}$$

4. Адаптація коварійної матриці

Мета цього кроку – обчислити коварійну матрицю в поколінні n. Нижче наведено деякі параметри, які використовуються в процесі розрахунку коваріаційної матриці. Коварійна матриця поточного покоління залежить від кривої навчання, що грунтується на попередній матриці коваріації.

$$C^{(n+1)} = (1 - c_{cov})C^n + \frac{c_{cov}}{\mu_{cov}} P_c^{(n+1)} P_c^{(n+1)^T} + c_{cov}\left(1 - \frac{1}{\mu_{cov}}\right) \times$$

$$\sum_{i=1}^{x} w_i \left(\frac{X_{1:\lambda}^{(n+1)} - m^n}{\sigma^{(n)}}\right)\left(\frac{X_{1:\lambda}^{(n+1)} - m^n}{\sigma^{(n)}}\right)^T \quad (1.8)$$

де c_{cov} — це параметр стратегії, що дорівнює $min(\mu_{cov}, \mu_{eff}, n^2)/n^2$, $\mu_{cov} > 0$ і $\mu_{cov} \sim \mu_{eff}$ у більшості випадків. У нашому випадку $\mu_{eff} = 3.4$, $P_c^{(n+1)}$– шлях еволюції. Стратегія є послідовністю кроків, що охоплює кілька поколінь. Шлях еволюції можна визначити як:

$$P_C^{(n+1)} = (1 - c_c)P_C^{(n)} + \sqrt{c_c(2 - c_c)\mu_{eff}}\,\frac{m^{(n+1)} - m^n}{\sigma^{(n)}} \quad (1.9)$$

5. Адаптація дисперсії

Разом з адаптацією коваріаційної матриці розмір дисперсії/кроку оновлюється кожного покоління за допомогою сукупної адаптації розміру кроку і може бути визначений як:

$$\sigma_{n+1} = \sigma_n \times exp\left(\frac{c_\sigma}{d_\sigma}\left(\frac{\|P_\sigma\|}{E\,\|N(0,I)\|} - 1\right)\right), \quad (1.10)$$

де P_σ визначається як

$$P_\sigma = (1 - c_\sigma)P_\sigma + \sqrt{1 - (1 - c_\sigma)^2}\sqrt{\mu_w}\,{C_K}^{-1/2}\,\frac{m_{K+1} - m_{K+1}}{\sigma_k} \quad (1.11)$$

і $\mu_w = \left(\sum_{i=1}^{\mu} {w_i}^2\right)^{-1}$

Як і багато інших еволюційних алгоритмів, CMA-ES сприймається з точки зору користувача як квазіпараметричний. Користувачеві необхідно вибрати початкову точку розв'язку $m^{(0)} \in R^d$ та початковий розмір кроку $\sigma^{(0)}$. Доступна також опція налаштування кількості вибірок-кандидатів (розмір популяції), позначений як λ, щоб вплинути на характерну поведінку пошуку. Крім того, критерії зупинки можуть бути змінені, щоб краще відповідати конкретній проблемі.

CMA-ES продемонстрував емпіричний успіх у багатьох додатках, особливо у сценаріях, що включають неопуклі, нероздільні, мультимодальні або зашумлені цільові функції. У порівняльному огляді оптимізації «чорної скриньки» [92] CMA-ES перевершив 31 інший алгоритм оптимізації, продемонструвавши помітну ефективність при роботі зі складними функціями або у більших просторах пошуку.

CMA-ES, зазвичай застосовуваний у просторах пошуку з розмірами від двох до кількох сотень, найбільш ефективний у сценаріях оптимізації «чорної скриньки», де градієнти недоступні, а оцінки функцій є основним фактором вартості пошуку.

1.2.2. Переваги використання алгоритму CMA-ES та його модифікацій для оптимізації параметрів

Еволюційна стратегія з адаптацією коваріаційної матриці (CMA-ES) - це потужний алгоритм оптимізації, який особливо надійний і конкурентоспроможний як для локальних, так і для глобальних завдань оптимізації. Однією з основних переваг використання CMA-ES для оптимізації параметрів є те, що він не містить квазіпараметрів, а пошук хороших параметрів стратегії (за замовчуванням) розглядається, як частина розробки алгоритму [86, 91]. Це робить його простим у використанні і не вимагає складних налаштувань параметрів. Крім того, CMA-ES має кілька властивостей інваріантності, таких, як інваріантність до збереження порядку та інваріантність до афінних перетворень, що робить алгоритм придатним для надійних задач оптимізації [91].

CMA-ES виявився емпірично успішним у сотнях додатків і вважається корисним під час вирішення складних завдань оптимізації.

У статті [93] пропонується покращена версія алгоритму стратегії розвитку адаптації матриці коваріації (CMA-ES), який має назву градієнтна інформаційна еволюційна стратегія (GI-ES), для задач великомасштабної оптимізації. Автори звертаються до проблем високої складності та передчасної стагнації CMA-ES при

вирішенні великомасштабних проблем оптимізації. Запропонований алгоритм GI-ES розширює застосування CMA-ES у сфері великомасштабної оптимізації шляхом використання градієнтної інформації.

Автори порівнюють продуктивність GI-ES з іншими сучасними алгоритмами оптимізації, такими як оригінальний CMA-ES, алгоритм диференціальної еволюції та алгоритм оптимізації рою частинок. Результати експериментів показують, що GI-ES перевершує інші алгоритми з точки зору швидкості збіжності та якості розв'язку щодо кількох еталонних функцій. Запропонований алгоритм GI-ES замінює обчислення матриці коваріаційної стратегії оцінки очікуваного ступеня відповідності і оцінює інформацію про градієнт, не використовуючи її для відбору.

У [94] автори досліджують застосування стратегії еволюції адаптації коваріаційної матриці (CMA-ES) для оптимізації гіперпараметрів у машинному навчанні.

Дослідники пропонують метод під назвою «Теплий запуск CMA-ES» (Warm Starting CMA-ES) для вирішення проблеми перенесення оптимізації гіперпараметрів на інші набори даних або цілі. У статті наведено експерименти, в яких WS-CMA-ES застосовується до кількох проблем оптимізації гіперпараметрів для перевірки його ефективності. Крім того, він є іграшковим прикладом використання CMA-ES для налаштування гіперпараметрів згорткової нейронної мережі для набору даних MNIST на 30 графічних процесорах паралельно.

У дослідженні підкреслюється потенціал CMA-ES, особливо WS-CMA-ES, як ефективного підходу до оптимізації гіперпараметрів у завданнях машинного навчання, що демонструє його надійність та ефективність при роботі з різними наборами даних.

У статті [95] представлено інноваційний підхід, який використовує сурогатну модель зі змінною точністю, засновану на пошуку знань, для підвищення продуктивності багатокритеріальної моделі. Дослідження спрямоване на підвищення ефективності CMA-ES під час вирішення задач багатокритеріальної оптимізації за

рахунок включення сурогатної моделі, яка отримує знання про завдання багатокритеріальної оптимізації.

Дослідження є значним внеском у теорію еволюційних алгоритмів, особливо у контексті багатокритеріальної оптимізації. Шляхом інтеграції сурогатних моделей, заснованих на пошуку знань зі CMA-ES, запропонований алгоритм демонструє потенціал для досягнення ефективних розв'язків при розгляді багатокритеріальної оптимізації. Адаптивне оновлення основних параметрів алгоритму ще більше підвищує його ефективність при вирішенні складних та різноманітних завдань оптимізації. Результати та методологія, відображені у статті, сприяють розвитку еволюційних алгоритмів, демонструючи потенціал практичного застосування у різних галузях.

Стаття [96] Френка Хаттера пропонує використання еволюційної стратегії з адаптацією коваріаційної матриці (CMA-ES) для оптимізації гіперпараметрів глибинних нейронних мереж. Дослідження пропонує CMA-ES як альтернативу пошуку по сітці, випадковому пошуку або байєсівській оптимізації для оптимізації гіперпараметрів глибинних нейронних мереж. У статті наведено приклад порівняння CMA-ES із найсучаснішими алгоритмами байєсівської оптимізації для налаштування гіперпараметрів згорткової нейронної мережі для набору даних MNIST. Дослідження демонструє потенціал CMA-ES в оптимізації гіперпараметрів глибинних нейронних мереж, підкреслюючи його ефективність у вирішенні складних завдань оптимізації. Дослідження є значним внеском у сферу оптимізації гіперпараметрів, особливо в контексті глибинних нейронних мереж.

У статті [97] пропонується новий підхід до оптимізації «чорної скриньки» шляхом непараметричного моделювання вхідних просторів при відтворенні гільбертових просторів ядра (RKHS – Reproducing kernel Hilbert space) та CMA-ES алгоритму. У статті розглядається задача оптимізації цільових функцій «чорної скриньки» у чітко визначеному просторі параметрів, де функції ознак часто визначаються вручну. Використовуючи RKHS, дослідження спрямоване на

підвищення продуктивності CMA-ES у сценаріях, де якість вибраних функцій або базового простору параметричних функцій впливає на методи оптимізації. У статті також наведено експериментальні результати, що демонструють ефективність запропонованого підходу при вирішенні двох простих завдань функціональної оптимізації та двох завдань навчання з підкріпленням.

Дослідження є значним внеском в область оптимізації чорної скриньки, особливо в контексті використання CMA-ES в RKHS для непараметричного моделювання вхідних просторів. Результати наголошують на потенціалі розширення можливостей CMA-ES у вирішенні складних завдань оптимізації за рахунок використання непараметричного моделювання в RKHS.

У роботі [98] представлена модифікація CMA-ES для досягнення лінійної складності у часі та просторі. Пропонована модифікація спрямована на усунення внутрішньої обчислювальної складності CMA-ES, особливо в контексті вибірки загального багатовимірного нормально розподіленого випадкового вектора та оновлення матриці коваріаційної. У статті обговорюються обчислювальні складності цих операцій та представлена проста модифікація для досягнення лінійної складності за часом та простором з метою підвищення ефективності CMA-ES.

Дослідження дає цінну інформацію про обчислювальні складності CMA-ES та пропонує практичну модифікацію для підвищення його ефективності. Досягаючи лінійної складності у часі та просторі, пропонована модифікація сприяє масштабованості та застосовності CMA-ES при вирішенні багатовимірних задач оптимізації.

У статті [99] авторами розглянуто новий механізм адаптації популяційних алгоритмів із сурогатною підтримкою, що застосовується до Стратегії еволюції адаптації матриці коварації. Запропонований механізм, званий s∗ACM-ES, регулює в онлайновому режимі тривалість життя поточної сурогатної моделі, кількість поколінь CMA-ES перед вивченням нової сурогатної моделі та кількість поколінь CMA-ES перед повторним вивченням сурогатної моделі. У роботі демонструється ефективність

s∗ACM-ES у підвищенні якості сурогатної моделі, що призводить до значного прискорення s∗ACM-ES у порівнянні з базовими моделями ACM-ES та CMA-ES. Емпірична перевірка s∗ACM-ES демонструє ефективність та масштабованість запропонованого підходу, досягаючи нових, кращих результатів за деякими з еталонних завдань.

Дослідження дає важливу інформацію про розробку та продуктивність популяційних алгоритмів із сурогатною підтримкою, особливо в контексті CMA-ES. Запропонований механізм s*ACM-ES демонструє значні покращення якості сурогатної моделі, сприяючи розвитку еволюційних алгоритмів у сфері чисельної оптимізації.

У роботі [100] запропоновано інноваційний підхід, який використовує методи інтелектуального аналізу даних для розробки конкурентоспроможного алгоритму стратегії еволюції коварійної матриці з використанням сурогатної матриці. Дослідження зосереджено на підвищенні ефективності та результативності CMA-ES за рахунок включення сурогатних моделей зі змінною точністю та використання методів інтелектуального аналізу даних для адаптивного настроювання точності сурогатних моделей.

Дослідження робить значний внесок у область еволюційних алгоритмів, особливо у контексті сурогатної оптимізації та інтелектуального аналізу даних. За рахунок інтеграції сурогатних моделей змінної точності та методів інтелектуального аналізу даних запропонований алгоритм демонструє потенціал у досягненні ефективних розв’язків для складних завдань оптимізації.

У статті Кенти Нісіди та Юхея Акімото [101] запропоновано новий підхід до алгоритму CMA-ES, званий PSA-CMA-ES, який включає механізм адаптації розміру популяції. Пропонований алгоритм спрямовано на підвищення продуктивності CMA-ES з допомогою адаптивного регулювання розміру популяції з урахуванням поточного стану процесу оптимізації. У роботі науковців відзначено ефективність

PSA-CMA-ES у вирішенні низки еталонних завдань, продемонстровано його чудову продуктивність у порівняні з іншими сучасними алгоритмами.

Дослідження [102] описує новий підхід до алгоритму CMA-ES, який називається кооперативним коеволюційним CMA-ES з угрупуванням з урахуванням ландшафту (CC-CMA-ES-LAG Cooperative coevolutionary CMA-ES landscape-aware grouping), який включає кооперативну коеволюційну структуру і метод угруповання з урахуванням ландшафту для поліпшення продуктивності CMA-ES в шумному середовищі. Пропонований алгоритм спрямований на підвищення ефективності та результативності CMA-ES за рахунок угруповання змінних розв’язку на основі їх поведінки та адаптації коварійної матриці кожної групи окремо.

За рахунок включення методу угруповання з урахуванням ландшафту та кооперативної коеволюційної структури запропонований алгоритм демонструє потенціал у досягненні ефективних розв'язків складних завдань оптимізації в шумному середовищі. Адаптивне налаштування коваріаційної матриці кожної групи окремо сприяє ефективності алгоритму у вирішенні різноманітних оптимізаційних завдань.

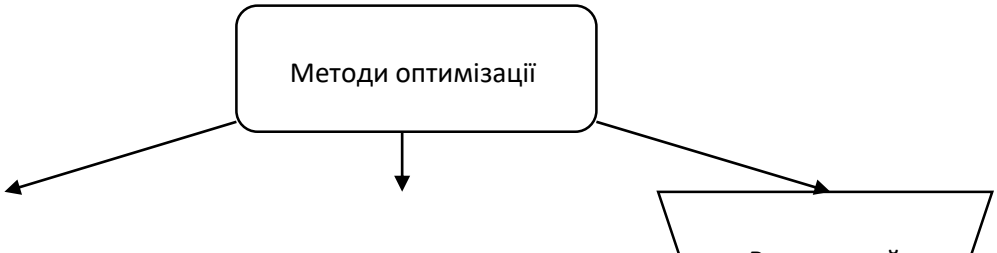
У [103] розглянуто новий гібридний алгоритм, який поєднує в собі метод рою часток (PSO) і CMA-ES для підвищення продуктивності глобальної оптимізації. Автори спочатку пропонують PSO у тимчасовому вікні (TW (Temporary Window) - PSO), як вдосконалення PSO, що розширює алгоритм дослідницьких можливостей. Потім вони об'єднують TW-PSO зі CMA-ES, щоб сформувати гібридний алгоритм, який використовує сильні сторони обох алгоритмів. Запропонований алгоритм протестований на набір еталонних функцій, і результати показують, що він перевершує PSO і CMA-ES з точки зору швидкості схожості та якості розв’язку. У цілому в статті представлено багатообіцяючий підхід до глобальної оптимізації, який може бути корисним у різних додатках.

У роботі [104] автори запропонували інноваційний підхід до багатоцільової оптимізації з використанням алгоритму стратегії адаптації коваріаційної матриці за

допомогою сурогатної моделі змінної точності на основі вилучення знань. Автори розглянули основний принцип багатоцільового CMA-ES і обговорили адаптивне оновлення основних параметрів в алгоритмі. Запропонований підхід спрямований на підвищення ефективності та результативності алгоритму CMA-ES у розв'язанні складних, невипуклих, мультимодальних та зашумлених задач оптимізації. Використання сурогатної моделі зі змінною точністю, заснованої на вилученні знань, є новим внеском у покращення продуктивності CMA-ES у задачах багатоцільової оптимізації.

1.3. Інші еволюційні алгоритми пошуку гіперпараметрів

У 1960-ті роки у різних місцях з'явилися три різні інтерпретації цієї ідеї. Лоуренс Джером Фогель розробив еволюційне програмування у США, а Джон Генрі в Голландії ініціював свою методологію як генетичний алгоритм, натхненну еволюційними концепціями Дарвіна. У Німеччині Інго Рехенберг та Ханс-Пауль Швефель розробили стратегії еволюції. Четверта інтерпретація, відома як генетичне програмування, з'явилася на початку 90-х років [1]. Незважаючи на різноманітність термінології, ці чотири підходи – еволюційне програмування, стратегії еволюції, генетичні алгоритми та генетичне програмування – вважаються представниками єдиної технології, відомої як еволюційні алгоритми (EA – Evolutionary algorithms). Еволюційні алгоритми охоплюють всю область, включаючи ці підобласті, як показано на рисунку 1.6 [105].



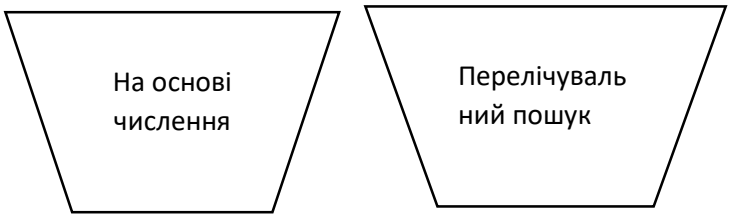


Рис. 1.6. Методи оптимізації

Незважаючи на подібність концепції, а в деяких випадках і формулювання, ці напрями на початок 1990-х років розвивалися незалежно [106, 107]. Після цього взаємодія між ними поступово налагоджувалася, дозволяючи кожному напрямку запозичувати деякі ідеї із розробок іншого. Пізніше для оптимізації параметрів було запропоновано диференціальну еволюцію [108]. Крім того, є ще два методи, які не призначені для оптимізації параметрів [109]:

- генетичне програмування [110], що спрямоване на пошук оптимальної програми або моделі, що найкраще відповідає вхідним даним.

- класичні системи навчання [111], що використовують принципи еволюції для пошуку оптимального набору правил для оптимізації поведінки системи для різних вхідних даних.

Генетичні алгоритми та стратегії еволюції є особливо відомими варіантами ЕА. Генетичний алгоритм GA приділяє основну увагу рекомбінації як основному оператору пошуку, тоді як еволюційна стратегія приділяє більше уваги мутаціям.

Генетичні алгоритми являють собою варіант еволюційних алгоритмів, що спочатку спираються на вивчення бітових рядків, аналогічних біологічному алфавіту ДНК. У контексті GA двійкові рядки можна декодувати в цілі чи дійсні значення різними методами. Сам рядок служить генотипом індивідуума, а фенотип реалізується у вигляді зіставлення параметрів об'єкта, зазвичай званого «порівнянням генотипу-фенотипу» [112]. Основним методом відбору у GA є відбір пропорційної пристосованості. Імовірність того, що особина буде обрана для розмноження, залежить від її пристосованості. Наприклад, якщо популяція складається з особини з пристосованістю 1 і другої особини з пристосованістю 3, існує ймовірність 1/(1+3) = 1/4 для вибору першої особини і 3/(1 +3) = 3/4 вибору другої особини . Для генетичного алгоритму існує багато інших методів відбору, наприклад турнірний відбір.

Мутація у GA додає до популяції нову інформацію і гарантує, що пошук не зупиниться. Простий метод мутації - це переворот бітів для індивідуумів із двійковим кодуванням: у випадково обраній позиції «0» змінюється на «1» і навпаки. Окрім мутації, важливим оператором варіації для GA є схрещування. Він має дві мети: скорочення пошуку розв’язку до більш перспективних областей і успадкування хороших властивостей генів.

У ES розв’язки λ-нащадків генеруються з μ-батьків шляхом модифікації батьків за допомогою мутації та, за бажання, рекомбінації (μ – кількість батьків, що вибираються на кожній ітерації, а λ – кількість нащадків). Звичайне співвідношення між μ та λ зазвичай знаходиться в діапазоні від 1:5 до 1:7 [113]. Це контрастує з циклом генерації GA, який зазвичай підтримує постійний розмір популяції. Крім того, відбір в ES є детермінованим, тоді як у ГА часто стохастичний.

У класичній стратегії еволюції мутація передбачає використання нормально розподілених випадкових величин, внаслідок чого невеликі зміни у рішенні відбуваються частіше, ніж великі зміни. З іншого боку, для GA розроблено різні оператори мутації, призначені як для неперервної оптимізації параметрів, так комбінаторних завдань. Концепція параметрів стратегії, що самоадаптуються, була введена на початку досліджень ES. Зокрема, у ES розмір кроку мутації можна зробити самоадаптивним, застосувавши логарифмічно нормально розподілену випадкову величину для зміни розміру кроку перед використанням цього розміру кроку для зміни параметрів розв'язку [114].

Еволюційне програмування, на відміну від генетичних алгоритмів та стратегій еволюції, використовує оператор варіації лише для мутацій там, де існують або можуть бути легко розроблені адаптивні та/або самоадаптивні методи для редагування параметрів оператора мутації у процесі еволюції.

Еволюційне програмування приймає фіксовану структуру програми із можливістю зміни її числових параметрів. Фундаментальні етапи підходу еволюційного програмування [115] містять наступні кроки:

1) створення потомства у вигляді мутації особин у поточній популяції;

2) відбір наступного покоління як із нащадків, так і з батьківської популяції.

Еволюційне програмування діє безпосередньо на природному представленні проблеми, тому відображення генотипу-фенотипу не використовується. На відміну від GA та ES, еволюційне програмування не використовує рекомбінацію (схрещування).

1.3.1 Математична модель еволюційного алгоритму

Еволюційні алгоритми працюють, розвиваючи популяцію можливих розв'язків протягом наступних поколінь, імітуючи те, як біологічні організми розвиваються та адаптуються до навколишнього середовища.

Процес роботи ЕА починається з початкового набору чи сукупності альтернативних розв'язків (окремих особин) для зазначеної проблеми. Ці початкові

розв’язки часто генеруються випадковим чином. Далі слідує ітераційний цикл, в якому нові та модифіковані пропозиції розв’язків систематично генеруються на основі попередніх розв’язків останнього циклу. При створенні потомства інформація про розв’язки копіюється від батьків, а модифікації вносяться з допомогою одного чи кількох операторів варіації, як мутація чи схрещування. Отримане потомство оцінюється, а найкращі особини відбираються і інтегруються у нову популяцію. Залежно від варіанта ЕА, за виживання можуть боротися як батьки, так і їхні нащадки.

Взаємодія між стохастичними змінами, що вносяться за допомогою операторів варіації, та перевагою кращих розв’язків у процесі вибору, сприяє ітеративному поліпшенню пропозицій розв’язків протягом численних циклів генерації. Цей циклічний процес триває до того часу, доки буде виконано критерій завершення. Зрештою користувачеві надається результат, зазвичай кращий розв’язок, визначений під час виконання, і алгоритм завершує роботу [113].

Базову математичну модель еволюційного алгоритму можна описати наступними етапами (рис. 1.7):

1. Ініціалізація:

- створення початкової групи потенційних розв’язків (окремі особини)

$P_0 = (x_1,\ x_1, \dots, x_n,)$, де $n -$ чисельність популяції.

Оцінка пристосованості кожної особини в популяції, яка кількісно визначає, наскільки добре кожний розв’язок вирішує задану проблему. Нехай $f(x_1)$ представляє пристосованість особи x_i: $F_0 = \big(f(x_1),\ f(x_2), \dots, f(x_n)\big)$.

Тут P_0 представляє початкову популяцію, а F_0 – відповідний масив значень пристосованості для кожної особини. Функція пристосованості $f(x_i\ )$ вимірює якість або продуктивність кожного розв’язку x_i щодо розглянутої задачі оптимізації.

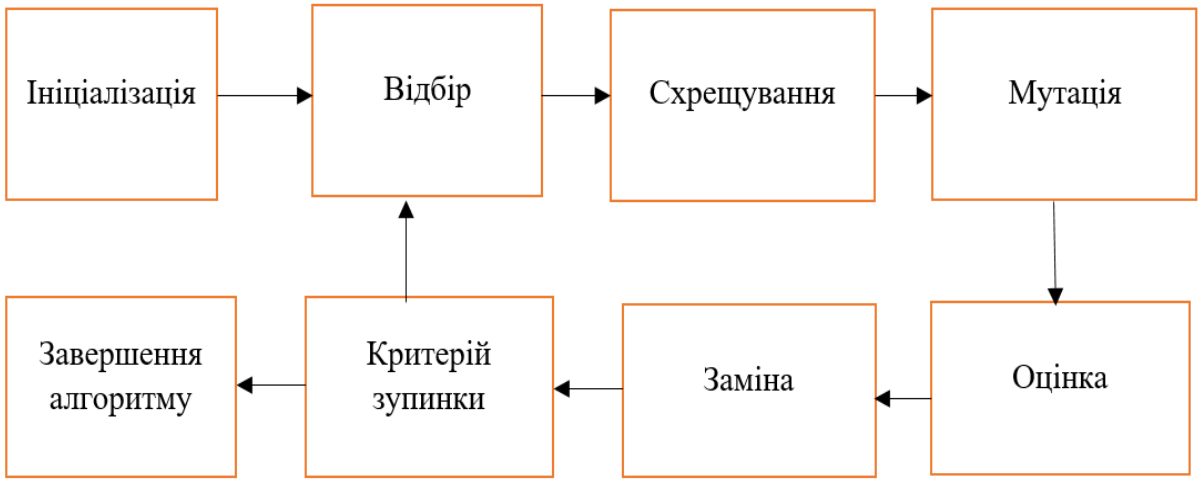


Рис. 1.7. Базова схема роботи всіх еволюційних алгоритмів

2. Відбір:

- відбір виступає як фундаментальний оператор ЕА, що безпосередньо пов'язаний з дарвінівським принципом виживання найбільш пристосованих. Після завершення відбору кожного покоління обирається нова популяція рішень-кандидатів, що стане популяцією наступного покоління. Основна мета оператора відбору - гарантувати, що особини, які демонструють високу пристосованість або якість, збережуться у наступних поколіннях. Існують такі методи відбору, як:

- *Відбір колеса рулетки* – особи відбираються з ймовірністю, пропорційною їх пристосованості.

- *Відбір рангу* – індивіди ранжуються на основі їхньої пристосованості, а ймовірності вибору призначаються відповідно до їх рангу.

- *Відбір сталого стану* – у кожному поколінні для створення нового потомства відбирається підмножина хромосом з високою пристосованістю. Одночасно видаляється підмножина хромосом з низькою пристосованістю. Новостворене потомство потім займає місце видалених хромосом, а популяція найбільш пристосованих переходить до наступного покоління.

- *Відбір турніру* – це механізм імовірнісного відбору, який імітує змагання у стилі турніру між випадково вибраними особами, з яких вибирається найсильніший.

- *Елітарний відбір* – основна ідея даного методу полягає в тому, щоб ідентифікувати та захистити найефективніших індивідів, яких часто називають елітою, забезпечуючи їх прямий перехід до наступного покоління без рекомбінації чи мутації.

- *Відбір Больцмана* – цей підхід ґрунтується на розподілі Больцмана із статистичної механіки та вводить рівень стохастичності у процес відбору. Значення пристосованості особин у популяції перетворюються із застосуванням температурного параметра. Параметр температури контролює ступінь випадковості у процесі вибору, що дозволяє алгоритму досліджувати різні області пошукового простору, при цьому віддаючи перевагу особинам з більшою пристосованістю.

- Відбір особин-батьків із поточної популяції залежно від їхньої пристосованості. Одним з найпоширеніших методів є вибір колеса рулетки, в якому використовується розподіл ймовірностей, заснований на здатності особин вибирати батьків. Ймовірність P_i відбору особин пропорційна його пристосованості $f(x_i)$:

$$P_i = \frac{f(x_i)}{\sum_{j=1}^{n} f(x_i)}$$

Особи з вищою пристосованістю мають більше шансів бути обраними як батьки, що імітує принцип природного відбору, коли більш підготовлені особини мають кращі шанси на розмноження.

3. Схрещування:

- об'єднання обраних батьків $x_{parent1}$ та $x_{parent2}$ у пару та створення нового потомства x_{child}, об'єднанням їх генетичної інформації. Конкретний метод схрещування залежить від алгоритму та проблеми. Одним з найпоширеніших методів є одноточкове схрещування, яке включає вибір випадкової точки (положення) схрещування C і обмін генетичним матеріалом між батьками в цій точці.

$$x_{child} = x_{parent1}[1:C] + x_{parent1}[C+1:n]$$

4. Мутація:

- застосування випадкових змін до генетичної інформації потомства. Конкретні методи мутації можуть відрізнятися один від одного. Найбільш поширеними типами мутацій є мутація біта і мутація дійсного значення.

- У мутації зі зміною біта випадковий біт у нащадку перевертається (тобто змінюється з 0 на 1 або з 1 на 0) з певною ймовірністю p_m. Нехай x_{child} представляє потомство, тоді:

$$x_{mutated}[i] = \begin{cases} 1 - x_{child}[i], & \text{з ймовірністю } p_m \\ x_{child}[i], & \text{з ймовірністю } (1 - p_m) \end{cases}$$

- $x_{child}[i]$ – i-й ген у генетичному матеріалі нащадка.

- У мутації дійсного значення, що використовується під час роботи з неперервними змінними, до гена потомства застосовується випадкове збурення. Нехай x_{child} – нащадок і σ представляє розмір кроку мутації, тоді

$$x_{mutated}[i] = x_{child}[i] + n(0, \sigma^2),$$

де $n(0, \sigma^2)$ - випадкове значення, взяте з нормального розподілу із середнім 0 і дисперсією σ^2

- Ці формули демонструють, як різні методи мутації змінюють генетичний матеріал потомства в еволюційних алгоритмах. Вибір методу мутації та параметрів мутації, таких як p_m та σ, залежить від алгоритму та характеристик задачі. Випадкові зміни до генетичної інформації потомства вносять різноманітність у популяцію та перешкоджають збіжності до локальних оптимумів.

5. Оцінка:

- оцінка пристосованості новоствореного нащадка для конкретного завдання. Нехай $f(x_{mutated})$ позначають пристосованість особини, що мутувала. На цьому етапі використовується функція пристосованості, спеціально вибрана для проблеми, яку потрібно вирішити, щоб оцінити, наскільки добре працює потомство, що мутувало. Функція пристосованості кількісно визначає якість чи продуктивність особини стосовно цілей завдання. Фактична формула для $f(x_{mutated})$ варіюватиметься в залежності від характеру проблеми та цілей оптимізації.

6. Заміна:

- створення нової популяції для наступного покоління шляхом вибору особин із поточної популяції та потомства. Вибір може ґрунтуватися на придатності чи інших критеріях. Одним із найпоширеніших методів заміни є заміна поколінь, при якій нинішня популяція повністю замінюється новоствореним потомством.

$$P_{t+1} = P_{mutated,}$$

де P_{t+1} – популяція для наступного покоління,

$P_{mutated}$ – набір мутованих нащадків, створених у течії поточного покоління.

- При заміні поколінь вся популяція замінюється новоствореним потомством. Інші стратегії заміни - елітарність або стаціонарна заміна- можуть зберегти деякі особини із поточної популяції на основі їх придатності чи інших критеріїв під час впровадження нового потомства. Вибір конкретного методу залежить від алгоритму та вимог задачі.

7. Завершення алгоритму:

- крок завершення визначає, коли зупинити алгоритм. Можуть бути різні критерії завершення, але загальний із них — вказати фіксовану кількість поколінь ($n_{max\_nenerations}$) як умову зупинки. Умова завершення:

$$t \geq n_{max\_nenerations},$$

де t – репрезентує нинішнє покоління, $n_{max\_nenerations}$ – максимальна кількість поколінь, дозволена до зупинення алгоритму.

- У цьому випадку алгоритм завершиться після виконання наперед визначеної кількості поколінь. Однак критерії завершення можуть відрізнятися, і для визначення того, коли алгоритм повинен зупинитися, також можуть використовуватися інші умови, такі як досягнення певного порога придатності, обмеження часу або інші умови, специфічні для проблеми. Конкретний вибір умови завершення залежить від завдання та цілей оптимізації.

1.3.2. Інші еволюційні алгоритми оптимізації

Еволюційні алгоритми продовжують бути предметом значного інтересу та досліджень через їх потенціал у пошуку оптимальних гіперпараметрів у задачах машинного навчання та оптимізації. Чимало досліджень показують доцільність, застосування та майбутні перспективи еволюційних алгоритмів для цієї мети. Зокрема, у роботі [116] представлено всебічний огляд розвитку та прогресу в галузі еволюційних алгоритмів оптимізації параметрів за останні три десятиліття. У статті обговорюється кілька важливих розробок та досягнень у цій галузі, включаючи впровадження нових алгоритмів, автоматизоване проєктування алгоритмів та гіперевристику. Загалом автори здійснили детальний огляд прогресу та розробок у галузі еволюційних алгоритмів оптимізації параметрів за останні три десятиліття. У ньому наголошується на важливості цих алгоритмів для вирішення складних завдань оптимізації та їх потенціал для подальшого розвитку і застосування в різних галузях.

У роботі [117] автори пропонують новий еволюційний алгоритм оптимізації часткового підкріплення (PRO – Partial Reinforcement Optimizer). Ідея для PRO береться з теорії часткового ефекту підкріплення (PRE – Partial Reinforcement Effect) в еволюційному навчанні та тренінговій психології. Теорія PRE передбачає, що періодичне підкріплення покращує навчання або зміцнення певної поведінки. Він підкреслює значний вплив моделей підкріплення на швидкість реакції та силу учня під час розкладу підкріплення, що досягається шляхом відповідного вибору поведінки підкріплення та часу процесу підкріплення.

В алгоритмі PRO теорія PRE математично внесена в еволюційну структуру оптимізації для вирішень проблем глобальної оптимізації:

$$SF_i \leftarrow \tau + U(0, \bar{\beta}), where\ \bar{\beta} \leftarrow \sum_{j \in \mu} \left(\frac{Shedule_{i,j}}{max(Shedule_i)} \right)$$

$$S_i^{\mu} \leftarrow \begin{cases} \left(X_{best}^{\mu} - X_i^{\mu}\right)\ If\ rand < 0.5 \\ \left(X_i^{\mu} - X_i^{\mu}\right)\ Otherwise. \end{cases}$$

$$X_{i,new}^{\mu} \leftarrow X_i^{\mu} + SF_i \times S_i^{\mu},$$

де SF_i є фактором стимулювання, а $\bar{\beta}$ є середнім значенням нормалізованого бала (пріоритету) вибраних змінних розв'язків для i-го учня на основі його планувальника $Shedule_{i,j}$.

Ефективність алгоритму PRO оцінюється шляхом його порівняння з добре відомими метаевристичними алгоритмами, такими як SMA (Slime Mould Algorithm), HHO (Harris Hawks Optimization), AHA (Artificial Hummingbird Algorithm) за допомогою статистичних тестів Вілкоксона та Фрідмана. Аналіз базується на результатах, отриманих за допомогою 75 тестів тестів CEC2005, CEC2014 і CEC-BC-2017, що охоплюють унімодальні, мультимодальні, гібридні та композиційні функції.

У статті [118] обговорюється використання еволюційних алгоритмів, зокрема PSO та GA, для оптимізації гіперпараметрів у машинному навчанні для фізики високих енергій. Алгоритми тестуються на різних наборах даних та порівнюються з альтернативними методами. Продуктивність оцінюється за допомогою функції Розенброка, складної задачі мінімізації функції, а також задачі машинного навчання бозона Хіггса ATLAS. Результати показують потенціал цих еволюційних алгоритмів для автономної оптимізації гіперпараметрів.

Гіперпараметри оптимізуються шляхом перетворення задачі оптимізації гіперпараметрів на задачу максимізації функції, де цільова функція зіставляє точку в просторі гіперпараметрів з оцінкою:

$$\hat{h} = \operatorname{argmax}_{h \in \mathcal{H}} s(h),$$

де $s : \mathcal{H} \to \mathbb{R}$ відноситься до цільової функції, яка відображає точку h в $\mathcal{H}$ до оцінки $s(h)$.Тоді оптимальними гіперпараметрами є ті, що максимізують цю цільову функцію.

Цей підхід дозволяє оцінити продуктивність еволюційних алгоритмів, таких як PSO та GA, при пошуку оптимальних значень гіперпараметрів в автономному режимі.

У статті [119] обговорюються різні типи методів оптимізації гіперпараметрів і порівнюється їхня продуктивність у наборах даних класифікації зображень за допомогою моделей AutoML. У статті порівнюється ефективність різних методів оптимізації гіперпараметрів, включаючи пошук по сітці, випадковий пошук і CMA-ES на різних моделях і наборах даних.

Зокрема, у статті розглядається байєсівська оптимізація та пропонується використання генетичного алгоритму та диференціальної еволюції для оптимізації функції збору даних. Автори зауважують, що байєсівська оптимізація має тенденцію працювати погано, коли для оптимізації функції збору даних використовується генетичний алгоритм. Проте було виявлено, що використання байєсівської оптимізації в поєднанні з CMA-ES із диференціальною еволюцією потенційно покращує продуктивність і ефективність систем AutoML.

У публікації [120] увага зосереджена на використанні еволюційних алгоритмів для оптимізації параметрів у моделях глибинного навчання. У дослідженні також обговорюються різні стратегії інтеграції еволюційних алгоритмів у налаштування параметрів. Також описано оптимізацію гіперпараметрів багатошарового перцептрону у наборі даних MNIST, а також результати оптимізації з використанням різних алгоритмів.

У дослідженні використовуються різні еволюційні алгоритми, такі як спрощена групова оптимізація (SSO), бактеріальний еволюційний алгоритм (BEA), метод рою часток (PSO), диференціальна еволюція та генетичний алгоритм. Перевага цих алгоритмів полягає в тому, що вони здатні вирішувати та оптимізувати розривні, багатомодальні, багатовимірні та нелінійні задачі. Вказані алгоритми ефективні при вирішенні багатокритеріальної, нелінійної та обмеженої оптимізації та здатні досліджувати великі сфери застосування у прийнятних областях, але при цьому вимагають відхилень цільової функції, на відміну від підходів до навчання на основі градієнта. Ці характеристики роблять еволюційні алгоритми цінним інструментом для оптимізації параметрів у моделях глибинного навчання.

Висновки до розділу І

У розділі І дисертаційної роботи здійснено огляд актуальних наукових досліджень у галузі еволюційних алгоритмів, зокрема генетичних алгоритмів, еволюційних стратегій та їхніх модифікацій. Дана тематика важлива у різних галузях: у промисловості, машинобудуванні та складному плануванні, фінансах, медицині, мистецтві тощо. Огляд досліджень показує, що еволюційні алгоритми та їх модифікації стрімко розвиваються, створюються нові підходи та методи для розв'язання задач оптимізації.

У застосуванні еволюційні алгоритми стикаються із низкою проблем. Це,зокрема, масштабованість, конфігурація параметрів, швидкість збіжності. Генетичні алгоритми, наприклад, можуть погано масштабуватися, що ускладнює їх використання для вирішення завдань з великою кількістю елементів, схильних до мутацій. Також еволюційні алгоритми вимагають ретельного налаштування параметрів, що може бути складним завданням і забирати багато часу. Неправильні параметри можуть призвести до неоптимальних розв'язків або повільної збіжності. Деякі задачі оптимізації можуть вимагати дуже багато обчислень функцій, щоб знайти розв'язок. Еволюційні алгоритми можуть бути менш ефективними, ніж інші методи оптимізації, такі як імітація відпалу або ціле лінійне програмування з точки зору швидкості збіжності.

Огляд останніх досліджень дозволяє зробити висновок, що область еволюційних алгоритмів є міждисциплінарною, і дослідники часто співпрацюють із експертами з інших галузей – таких як соціальні науки, відкрита еволюція, штучне життя та штучний інтелект. Це вимагає глибокого розуміння багатьох галузей та здатності інтегрувати знання з різних дисциплін.

Незважаючи на ці проблеми, еволюційні алгоритми продовжують розроблятися та застосовуватися в різних галузях завдяки їхній здатності забезпечувати хороші наближені розв'язання складних задач.

РОЗДІЛ ІІ.
ПОБУДОВА САМОАДАПТИВНИХ АЛГОРИТМІВ НА ОСНОВІ СУМІШЕЙ РОЗПОДІЛ

Теорія оптимізації є однією з найбільш вживаних у прикладній математиці. Це пов'язано з тим, що оптимізаційні задачі активно використовуються на виробництві, у машинобудуванні, логістиці та інших сферах життєдіяльності людини. Слід зауважити, що задачі оптимізації є ключовими підзадачами машинного навчання, глибинного навчання та штучного інтелекту. Наприклад, задача кластеризації даних $X = \{x_1, \dots, x_N\}, x_i \in R^n$ на основі методу k-середніх ґрунтується на оптимізаційній задачі, що мінімізує сумарну дисперсію в кластерах, тобто оптимізаційна функція в даній задачі визначаються співвідношенням

$$f(C;k) = \sum_{i=1}^{k} \frac{1}{N_i} \sum_{x_j \in S_i} \left|\left|x_j - c_j\right|\right|^2,$$

де $C = (c_1, \dots, c_k)$ – центри кластерів, причому $c_i \in R^d$; k – кількість кластерів; S_i – кластери множини X; $N_i > 0$ – розмірність кластеру S_i, тобто кількість елементів в S_i. Цільова функція залежить від змінних $C = (c_1, \dots, c_k)$ або, об'єднавши центри кластерів та перейшовши до одного вектору, від вектору розмірності $C \in R^{nk}$. Таким чином, описана вище задача є складною, по-перше, за рахунок розмірності, по-друге, за рахунок мультимодальності цільової функції. Метод k-середніх будується на випадковому виборі початкових центрів кластерів, що в остаточному результаті призводить до знаходження локального мінімуму цільової функції $f(C;k)$. Крім того, алгоритм k-середніх не можна віднести до самоадаптивних алгоритмів, оскільки залежність від початкових значень центрів є однією із основних деталей даного алгоритму. Надалі в цьому розділі буде наведено ще декілька класичних задач, які потребують більш скрупульозного підходу з використанням самоадаптивних алгоритмів за рахунок складності обчислення цільової функції f.

Оскільки в першу чергу розглядаються складні системи, такі як нейронні мережі, то основним показником якості алгоритмів в даному розділі буде кількість викликів цільової функції f. Для прикладу, оцінка параметрів нейронної мережі при відомих значеннях гіперпараметрів [121, 122] є дуже трудомісткою задачею, яка включає використання деяких градієнтних або квазіградієнтних алгоритмів. У зв'язку із цим кількість викликів функції, на нашу думку, є одним із найбільш важливих характеристик еволюційних алгоритмів.

Побудова самоадаптивних алгоритмів – один із найважливіших напрямків машинного навчання, глибинного навчання, штучного інтелекту, зокрема, і теорії оптимізації в цілому. Цей напрямок набув настільки широкого розвитку за рахунок багатьох задач, які призводять до потреби налаштування параметрів складних систем, наприклад нейронних мереж [96, 124]. Задачам оцінок параметрів нейронних мереж присвячена велика кількість робіт, і основні методи оцінок параметрів зводяться до градієнтних методів (метод градієнтного спуску, метод зворотного поширення тощо) [96] або їх стохастичних аналогів (методів стохастичного градієнтного спуску, стохастичного градієнтного спуску другого порядку, суперлінійних оптимізаційних алгоритмів тощо) [125, 126, 127]. Проте оцінкам гіперпараметрів нейронних мереж (кількість та розмір прихованих шарів, тип передавальних функцій, функцій активацій, типу з'єднань між шарами тощо) приділяється значно менше уваги. Це пов'язано насамперед із обчислювальною складністю, яку слід враховувати при переобчисленні гіперпараметрів. По-друге, цільові функції, які використовуються при оптимізації гіперпараметрів нейронних мереж залежать від цілих параметрів, що унеможливлює використання класичних алгоритмів із неперервними розподілами параметрів, таких як нормальний розподіл в класичному CMA-ES алгоритмі. Найбільш поширеними методами для налаштування (tuning) гіперпараметрів нейронних мереж є методи грубої сили, а точніше, пошук по сітці та його "м'які" стохастичні аналоги – випадковий пошук по сітці (stochastic grid search) [128, 129]. Метод випадкового пошуку по сітці і став відправною точкою у даному розділі, в

якому основна увага зосереджена на виборі нових хромосом для наступних ітерацій процесу пошуку оптимальних гіперпараметрів нейронної мережі. Слід зауважити, що інші ключові аспекти еволюційних алгоритмів, а саме визначення функцій мутацій та схрещування, у роботі не взято до уваги, оскільки даний напрямок дослідження,по-перше, знову ж таки потребує вибору між скінченною кількість стратегій, по-друге, не є основним завданням дисертаційної роботи.

Більш детальне дослідженням методів оцінки параметрів чи гіперпараметрів складних систем присвячено велику кількість робіт, наприклад роботи [96, 124, 130, 131, 132, 133, 134, 135]. Основні проблеми при оцінці параметрів пов’язані зі складністю алгоритмів, які використовуються у стохастичних диференціальних рівняннях та інших складних системах, наприклад, у нейронних мережах. Основну увагу даної роботи сконцентровано на розширеному алгоритмі еволюційної стратегії з адаптацією коваріаційної матриці (covariance matrix adaptation evolution strategy, CMA-ES) та його модифікаціях як прикладі створення самоадаптивних алгоритмів. Розглянемо більш детально класичний CMA-ES алгоритм та його модифікації, створення кожної з яких продиктовано конкретною прикладною задачею.

Для прикладу, у [136] описано метод стратегії диференціальної еволюції (DES) - алгоритм, що є чимось середнім між диференціальною еволюцією (DE) та CMA-ES. DES використовує комбінації різницевих векторів між архівними індивідами і одновимірними гаусівськими випадковими векторами вздовж напрямків минулих зсувів середніх точок. DES - метод певною мірою відповідає CMA-ES, але обробляє точки, а не розподіли. Згідно з експериментальними результатами, DES показує швидкість лінійної збіжності для квадратичних функцій у широкому спектрі чисел обумовленості матриці Гессе. Чисельні результати, представлені у статті, показують, що DES конкурентоспроможний проти CMA-ES під час як локальної, так і глобальної оптимізації.

У [137] пропонується новий варіант CMA-ES, названий AEALSCE, для задач оптимізації в неперервній області. AEALSCE отримано шляхом інтеграції CMA-ES із

двома стратегіями, які можуть коригувати еволюційні напрямки та збагачувати різноманітність популяції індивідів. Автори пропонують нову стратегію адаптації анізотропного власного значення (AEA), що адаптує область пошуку до оптимальних еволюційних напрямків та анізотропно масштабує власні значення коваріаційної матриці на основі виявлення локального ландшафту (local fitness landscape) відповідності:

$$\begin{cases} x_L = m_t + \delta v_j \\ x_R = m_t - \delta v_j \end{cases}$$

де $\delta \sim N(0, \lambda_j), v_j$ вказує на $j-$й власний вектор коваріаційної матриці 'найбільш хороших' хромосом x_i для цільової функції f, $\lambda_j - j$-те власне значення даної коваріаційної матриці відповідно. Із даного алгоритму зрозуміло, що ідея дуже тісно корелює із виділенням основних компонент у класичному методі головних компонент (principal component analysis, PCA), причому розширення чи звуження коваріаційної матриці у напрямку найбільш ймовірному для знаходження оптимуму.

Інша стратегія називається стратегією локального пошуку (local search, LS), яка виконується у власній системі координат і як локальний метод експлуатації може збагатити різноманітність популяції. Згідно зі статистичними результатами експериментів, запропонований авторами алгоритм AEALSCE не поступається іншим алгоритмам ефективної точності. Слід зауважити, що ідея локального пошуку також може слугувати відправною точкою для побудови самоадаптивних алгоритмів, оскільки включає в себе визначення локальних екстремумів на основі інформації про попередні етапи еволюційного чи генетичного алгоритму.

У [139] запропоновано модифікований метод обробки обмеження блоку (box constraint) для CMA-ES. Також, як і в [137] ідея полягає в оптимізації без обмежень за рахунок введення штучного фітнес-ландшафту (artificial fitness landscape), де функція штрафу додається до значень функції в найближчих допустимих розв'язках. Адаптуючи штрафні коефіцієнти, що визначають чутливість обмежень до значення цільової функції, він створює розумний ландшафт віртуальної функції за межами

допустимої області. Оптимізація працює доти, доки не буде досягнуто розумних штрафних коефіцієнтів. Експериментальні результати показують, що запропонований алгоритм може не збігатися на функції показникового множника, тоді як запропонований у роботі алгоритм демонструє збіжність.

Застосування гібридизованої еволюційної стратегії з адаптацією коваріаційної матриці крос-ентропії (CE-CMAES) пропонує автор у роботі [140]. Запропонований CE-CMA-ES використовує перехресну ентропію для глобального дослідження простору пошуку та стратегію адаптації коваріаційної матриці для локальної експлуатації. Основна новизна алгоритму CE-CMA-ES - це адаптивний механізм зміни розміру кроку (стандартного відхилення), що контролюється шляхом еволюції і є сумою послідовних кроків у кожному вимірі. Порівняльний аналіз показує, що CE-CMA-ES досягає кращих результатів у порівнянні з сучасними алгоритмами.

У [141] розроблено метод самоадаптації коваріаційної матриці з відштовхувальними субпопуляціями (RS-CMSA), який гібридизує та об’єднує кілька концепцій і технік із різних існуючих методів, таких як точки табу, нормалізована відстань Махаланобіса тощо. RS-CMSA в основному складається з методу еволюційної стратегії з адаптацією коваріаційної матриці (CMSA-ES), підсиленої елітарністю, як основної пошукової системи кількох субпопуляцій однакового розміру. RS-CMSA показав значну перевагу над іншими методами (NMMSO та NEA2), які також тестувалися на більш складних композитних функціях.

У [142] автор розглядає непараметричне моделювання вхідних просторів в оптимізації чорної скриньки в просторах Гільберта відтворюючого ядра (RKHS). Це моделювання призводить до проблеми функціональної оптимізації, предметною областю якої є функціональний простір RKHS, що дозволяє проводити оптимізацію для різного роду функцій. Автор пропонує CMA-ES-RKHS, узагальнений алгоритм CMA-ES, здатний виконувати оптимізацію функцій чорної скриньки в RKHS. Результати CMA-ES-RKHS показують реальне виконання функціональної оптимізації та подолання проблеми підбору значень початкових параметрів.

2.1. Розширений CMA-ES алгоритм

У даному підрозділі запропоновано розширення одного із класичних еволюційних алгоритмів, а саме CMA-ES алгоритму, який в деталях описано в попередньому підрозділі. Слід зауважити, що розширення спрямовано на подолання проблеми мультимодальності цільової функції [143], а не подолання інших питань щодо інших параметрів алгоритму. Ґрунтуючись на попередніх дослідженнях, структуру визначення квазіоптимальних значень цільової функції можна вибирати згідно з декількома основними принципами:

- ***Принцип логарифмічного росту***. У даному випадку ваги хромосом будуються згідно з логарифмічним законом, тобто

$$w_i = \frac{log(i+1)}{log(N+1) + \cdots + log(2)},$$

де впорядкування хромосом проводиться від найменш оптимальної до найбільш оптимальної;

- ***Принцип приросту логарифмів***. У даному випадку ваги хромосом будуються відповідно до наступного закону:

$$w_i = \frac{log(i+1) - log(i)}{log(N+1)},$$

де впорядкування хромосом проводиться від найбільш оптимальної до найменш оптимальної;

- ***Принцип натурального впорядкування***. У даному випадку ваги хромосом будуть пропорційні значенням відповідної цільової функції, тобто

$$w_i = \frac{f(x_i)}{\sum_{j=1}^{N} f(x_j)},$$

де $f(x_i)$ – значення цільової функції для хромосоми x_i. Слід зауважити, що використання принципу натурального впорядкування можливе лише за умови невід'ємності цільової функції, тобто $f(x) \geq 0$ і, крім того, $f(x_i) > 0$ для деякої хромосоми x_i. Цей метод є корисним для використання при оптимізації функції правдоподібності або логарифмічної функції правдоподібності, проте має обмеження на вибір початкових хромосом, оскільки в результаті невдалого підбору може виникнути ситуація, при якій $f(x_i) = 0, i = 1, \dots, N$.

- ***Принцип степеневого росту***. У даному випадку ваги хромосом будуються відповідно до степеневого закону, тобто

$$w_i = \frac{i^\gamma}{N^\gamma + \cdots 2^\gamma + 1},$$

де впорядкування хромосом проводиться від найменш оптимальної до найбільш оптимальної, $\gamma > 0$ – додатковий параметр. При такому підході додатковий параметр γ оцінюються із врахуванням властивостей цільової функції (мультимодальності, неперервності тощо).

2.1.1. Дві модельні задачі

На початку даного розділу розглянуто задачу кластеризації з машинного навчання із мультимодальною цільовою функцією, яка вказує на неефективність багатьох класичних алгоритмів. Розглянемо ще дві задачі прикладного характеру, які будуть базуватися на оцінці гіперпараметрів складних систем, а саме оцінках параметрів стохастичних диференціальних рівнянь та гіперпараметрів нейронних мереж. Основним показником точності опису динаміки реальних явищ за допомогою стохастичних систем (у тому числі стохастичних диференціальних рівнянь) є середня квадратична похибка, яка зазвичай оцінюється на основі методу Монте-Карло або його модифікацій. Таким чином, використання методів Монте-Карло саме по собі зумовлює багатократне моделювання розв'язків стохастичних систем для оцінки

середньої квадратичної похибки, яка є цільовою функцією в даному випадку, а отже змушує зменшувати кількість викликів вхідної функції за допомогою деякого алгоритму пошуку гіперпараметрів складних систем.

Велика кількість сучасних динамічних систем описується за допомогою математичних моделей із врахуванням випадковостей, які в реальному житті з'являються за рахунок впливу великої кількості різних факторів. До таких систем відносяться і дискретні системи, наприклад, динамічні системи із випадковими складовими або випадковими коефіцієнтами; і неперервні системи - часові ряди або стохастичні диференціальні рівняння. Одну із проблем оцінки параметрів динамічних систем із врахуванням випадковостей, а саме проблему оцінки коефіцієнтів неавтономних стохастичних диференціальних рівнянь Ito, буде розглянуто нижче. У роботах [124, 130] проаналізовано проблему мінімізації дисперсії випадкового процесу, що описує динаміку розвитку сукупності клітин $\{x_1, x_2, x_3\}$, динаміка яких, в свою чергу, описується системою стохастичних диференціальних рівнянь

$$d\begin{pmatrix} x_1(t) \\ x_2(t) \\ x_3(t) \end{pmatrix} = \begin{pmatrix} -q-\alpha(t) & 0 & 2\gamma \\ \alpha(t) & -q-\beta & 0 \\ 0 & \beta & -q-\gamma \end{pmatrix}\begin{pmatrix} x_1(t) \\ x_2(t) \\ x_3(t) \end{pmatrix}dt +$$

$$\begin{pmatrix} -\sqrt{\alpha(t)x_1(t)} & 0 & 2\sqrt{\gamma x_3(t)} \\ \sqrt{\alpha(t)x_1(t)} & -\sqrt{\beta x_2(t)} & 0 \\ 0 & -\sqrt{\beta x_2(t)} & -\sqrt{\gamma x_3(t)} \end{pmatrix}\begin{pmatrix} dW_1(t) \\ dW_2(t) \\ dW_3(t) \end{pmatrix} +$$

$$\begin{pmatrix} \sqrt{qx_1(t)} & 0 & 0 \\ 0 & \sqrt{qx_2(t)} & 0 \\ 0 & 0 & \sqrt{qx_3(t)} \end{pmatrix}\begin{pmatrix} dW_4(t) \\ dW_5(t) \\ dW_6(t) \end{pmatrix}, \qquad (2.1)$$

де $W_i(t), (t) \geq 0$ – стандартні вінеровські процеси, що визначаються кореляційною матрицею $R = \left(R_{ij}\right)_{ij=1}^{6}$; β, γ, q– невідомі константи системи, які необхідно оцінити,

$\alpha(t)$ – невідома функція, яка зазвичай належить деякому наперед заданому класу. Наприклад, у роботах [144,145] розглянуто клас поліноміальних функцій, тобто

$$\alpha(t) \in A = \{\alpha_0 + \cdots + \alpha_k t^k\}, k \in N.$$

Більш складні простори функцій подано в роботі [146], де в якості класу функцій $\alpha(t)$ розглянуто раціональні функції

$$\alpha(t) \in A = \left\{\frac{\alpha_0^1 + \cdots + \alpha_k^1 t^k}{\alpha_0^2 + \cdots + \alpha_m^2 t^k}\right\}, k, m \in N$$

Основна проблема при аналізі системи (2.1) полягає в оцінці невідомих параметрів

$$(\beta, \gamma, q, \alpha, R) \in R_+^3 \times A \times \Xi,$$

де Ξ – простір додатньовизначених квадратних матриць розмірності 6×6 із одиничними діагональними елементами, а критерій оптимальності визначається за допомогою наступного функціоналу якості:

$$J(\beta, \gamma, q, \alpha, R) = \sum_{i=1}^{n} c_i \, E|x(t_i) - x_i|^2, \qquad (2.2)$$

причому вагові коефіцієнти будуть залежати від характеру прикладного дослідження. Найпростіший випадок відповідає рівномірному розподілу для коефіцієнтів, тобто $c_i = \frac{1}{n}$, проте можуть зустрічатися і застосування термінальних критеріїв, а саме $c_i = 0, i = 1, \dots, n-1,\ c_n = 1$. Також у формулі (2.2) через $x(t_i)$ позначено розв'язок системи (2.1) у моменти часу t_i, через x_i – відомі значення процесу, який потрібно оцінити зазвичай на основі запропонованої моделі (2.1). У формулі (2.2) важливими складовими є математичні сподівання відхилень $E|x(t_i) - x_i|^2$, тобто одна з мір відхилення від теоретичного значення. Слід зауважити, що за рахунок складного вигляду правої частини системи (2.1) неможливо у загальному випадку знайти аналітичний вигляд $E|x(t_i) - x_i|^2$, як функції від невідомих параметрів $(\beta, \gamma, q, \alpha, R)$. Тому для оцінки $E|x(t_i) - x_i|^2$ найчастіше використовується метод Монте- Карло із модифікаціями [147, 148, 149, 150, 151]. Одним із недоліків методів Монте-Карло є

неперервна лінійна залежність складності алгоритму від кількості симуляцій M, тобто складність методів Монте- Карло буде рівною $O(M)$. З іншого боку, для пошуку оптимального набору параметрів

$$\left(\hat{\beta},\hat{\gamma},\hat{q},\hat{\alpha},\hat{R}\right) = argmin_{(\beta,\gamma,q,\alpha,R)\in R_+^3\times A\times \Xi} J(\beta,\gamma,q,\alpha,R) \quad (2.3)$$

варто визначити метод із невисокою обчислювальною складністю. Оскільки досліджувані параметри розглядаються в неперервній області, то для пошуку оптимальних параметрів найчастіше використовують еволюційні алгоритми та генетичні або еволюційні алгоритми пошуку [152, 153, 154, 155], які, в свою чергу, мають неперервну лінійну залежність від кількості хромосом (особин) P в даному алгоритмі. Якщо узагальнити, то складність алгоритму пошуку оптимальних параметрів $\left(\hat{\beta},\hat{\gamma},\hat{q},\hat{\alpha},\hat{R}\right)$ рівняння (2.1) з генетичним алгоритмом в якості оновлення параметрів та алгоритму Монте-Карло для оцінки $E|x(t_i)-x_i|^2$ буде мати складність $O(M*P*T)$, де T величина розбиття на інтервалі $[0, \max\limits_{1\le i\le n} t_i]$. Також дану методику можна використовувати для задач з точками концентрації, модель яких розглянуто у роботі [156], проте для таких задач слід враховувати використання додаткових обчислювальних потужностей на моделювання реалізацій системи стохастичних диференціальних рівнянь.

Ще однією прикладною задачею, яка потребує покращень алгоритмів пошуку, як було вже зазначено вище, є задачі оцінки параметрів нейронних мереж [96, 157, 158, 159, 160]. Для нейронних мереж невеликої розмірності надають перевагу алгоритмам “грубої сили“, а саме алгоритмам пошуку по сітці, алгоритмам випадкового пошуку по сітці та їх модифікаціям. Проте для нейронних мереж великої розмірності дані тривіальні підходи вимагають великої кількості обчислень, тому використовуються байєсівські, градієнтні та генетичні алгоритми. Особливу увагу серед оптимізаційних алгоритмів оцінки гіперпараметрів нейронних мереж слід приділити еволюційній стратегії з адаптацією коваріаційної матриці [96, 124, 136, 160]. Основна ідея CMA-ES стратегії полягає у виборі нових особин у генетичному

алгоритмі на основі деякого базового розподілу (зазвичай нормального закону розподілу $N(\alpha, \sum)$, середнє значення α дисперсія $\sum$ якого оновлюється на кожній ітерації (епосі) на основі нових обчислених особин з найоптимальнішими значеннями на основі критерію якості (2.2) чи точності нейронної мережі для оптимізації гіперпараметрів нейронних мереж. Використання нормального розподілу не завжди може бути задовільним при оцінці параметрів, оскільки володіє ненульовою імовірністю моделювання хромосом, що виходять за область зміни параметрів, що, в свою чергу, призводить до задачі переходу від неперервного розподілу до деякого дискретного аналізу. Даний перехід буде призводити до збільшення ймовірності вибору граничних точок області дослідження.

Алгоритм CMA-ES або його розширення можуть бути використані як для задачі оптимізації параметрів системи стохастичних диференціальних рівнянь (2.1), так і для пошуку оптимальних гіперпараметрів нейронних мереж. Основним недоліком алгоритму CMA-ES є однопіковість базового розподілу та, як наслідок, велика дисперсія при оцінці параметрів. Дану проблему можна проілюструвати наступним простим прикладом мультимодальної функції

$$f_{T1}(x) = \frac{\sin(\sum_{i=1}^{n} x_i^2)}{\sum_{i=1}^{n} x_i^2} \ , \tag{2.4}$$

де $x = (x_1, \dots, x_n) \in R^n$. На рисунку 2.1 зображено функцію (2.4) у двовимірному просторі, тобто при $n = 2$ для $x \in [-20,20]^2$. Проблема використання еволюційних чи генетичних алгоритмів, зокрема, полягає у визначенні регіону, в якому знаходиться відшукуваний максимум чи мінімум . Крім того, на основі методу CMA-ES коваріаційна матриця при невдалому початковому значенні не буде збігатися до оптимального значення, яке б дозволяло концентруватися навколо оптимуму функції (2.4).

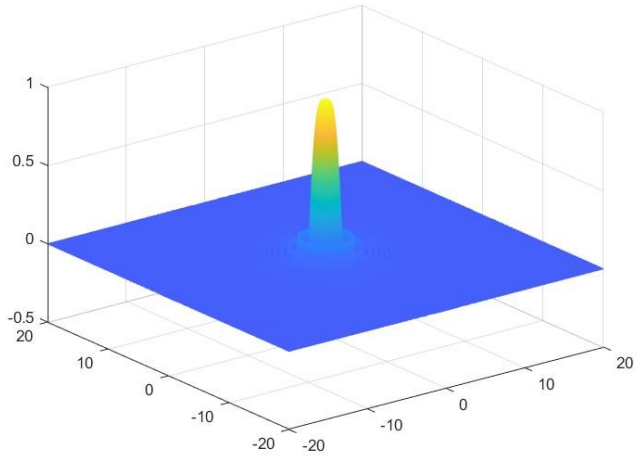


Рис. 2.1. Функція $f(x)$ для $n = 2$ та $x \in [-20,20]^2$

2.1.2. Використання сумішей в розширеному CMA-ES алгоритмі

CMA-ES алгоритм, як і у випадку Байєсівського алгоритму, полягає у перерахунку коваріаційної матриці розподілів гіперпараметрів між епохами еволюційного алгоритму з подальшим вибором параметрів та врахуванням даної матриці. Очевидним недоліком такого методу є те, що припускається однопіковість (унімодальність) щільності розподілу гіперпараметрів (як у нормальному розподілі). Проте на практиці цільова функція (точності, функції втрат чи довільна інша цільова функція) не є однопіковою, що приводить до збільшення області пошуку за рахунок зміни однієї коваріаційної матриці та включення в область пошуку генетичного алгоритму областей зі значеннями цільової функції, що значно відрізняються від локальних екстремумів. Також слід відзначити, що CMA-ES – це стохастичний метод, що відрізняється від класичних градієнтних методів для чисельної оптимізації нелінійних чи неопуклих задач неперервної оптимізації. Ітераційний алгоритм CMA-ES часто використовує багатовимірний розподіл Гаусса $N(m, \Sigma)$, де $m \in R^d, \Sigma \in R^{dxd}$ – додатновизначена симетрична матриця, d – число змінних. CMA-ES на кожній із своїх ітерацій вибирає λ кандидатів-рішень із багатовимірного нормального

розподілу, оцінює значення цільової функції на цих кандидатах (послідовно або паралельно), а потім коригує розподіл вибірки, що використовується для наступної ітерації, щоб надати більшу ймовірність ‘хорошим’ хромосомам, при цьому виводячи із розгляду ‘погані’ хромосоми, які мають найнижчі значення вагових коефіцієнтів [161]. Вектор середніх m та коварійна матриця Σ оновлюються відповідно до ранжування рішень в останньому поколінні – і CMA-ES навчається вибирати рішення з перспективної області.

CMA-ES, як правило, має тенденцію працювати найкраще для складних алгоритмів оцінки функцій; наприклад, у [126] показано, що CMA-ES дав найкращі результати серед більш ніж 100 класичних та сучасних оптимізаторів у широкому діапазоні функцій чорної скриньки. CMA-ES використовувався для налаштування гіперпараметрів і раніше, наприклад, у роботі [162], або автоматичного розпізнавання мови [163], тому використання розширеного варіанту CMA-ES алгоритму, звичайно, сприяє кращому розумінню багатьох задач оптимізації.

Як було зазначено вище, основною проблемою класичного CMA-ES алгоритму є однопіковість базового розподілу (нормальний розподіл $N(\mu, \Sigma)$ у класичному випадку), що призводить до ситуації, коли коваріаційна матриця не збігається до нульової матриці. Для подолання цієї проблеми пропонується використати деяке розширення CMA-ES алгоритму, використовуючи багатопікові (мультимодальні) моделі розподілу вибору нових хромосом. Для цього буде використано поняття суміші (суміші розподілів), зокрема, суміші нормальних розподілів, оскільки вибір цих розподілів є найбільш ілюстративний і легко може бути реалізований на практиці.

Надалі будемо припускати, що щільність суміші належить до одного сімейства розподілів [164], визначається як зважена сума k щільностей компонентів. Щільності компонентів обмежені деяким параметричним класом щільностей, який вважається придатним для наявних даних, обчислювальних цілей та відповідних умов на параметри систем, що оцінюються. Позначимо $p(x; \theta_s)$ – щільність s-го компонента,

де θ_s – параметри компонента; w_{ps} - ваговий коефіцієнт s-го компонента суміші. Ваги повинні бути невід'ємними $w_s \geq 0$ та в сумі давати одиницю $\sum_{s=1}^{k} w_s = 1$.

Ваги w_s також відомі як «пропорції змішування», і їх можна розглядати як ймовірність $p(s)$ того, що вибірка даних буде вилучатися з компонентів суміші s. Тоді щільність суміші компонентів k визначається як:

$$p(x) = \sum_{s=1}^{k} w_s p(x; \theta_s), \tag{2.5}$$

де $\theta = \{\theta_1, \dots, \theta_k, w_1, \dots, w_k\}$ - параметри суміші.

Щільність суміші можна інтерпретувати як моделювання процесу, в якому спочатку вибирається «джерело» s відповідно до поліноміального розподілу $w_1, \dots, w_k$, а потім береться вибірка з відповідної щільності компонентів $p(x; \theta_s)$. Таким чином, можливість вибору джерела s і даних x дорівнює $w_s p(x; \theta_s)$. Тоді як гранична можливість вибору даних x визначається формулою (2.5). Важливою похідною величиною є «апостеріорна ймовірність» компонента суміші, заданого вектором даних, яка використовується для оцінки параметрів суміші (2.5). Апостеріорна ймовірність компонентів суміші визначається за допомогою правила Байєса і має вигляд

$$p(s \mid x) := \frac{w_s p(x; \theta_s)}{p(x)} = \frac{w_s p(x; \theta_s)}{\sum_{\acute{s}} w_{\acute{s}} p(x; \theta_{\acute{s}})} \tag{2.6}$$

Першим кроком під час використання моделі суміші є визначення її архітектури: відповідний клас щільностей компонентів і кількість щільностей компонентів у суміші. Після того, як ці варіанти дизайну зроблені, оцінюються вільні параметри в моделі суміші таким чином, щоб щільність (2.6) якомога точніше наближала щільність реальних даних або ж у випадку використання еволюційного алгоритму розподіл екстремальних значень відповідної цільової функції.

Оцінка параметрів моделі для вхідних даних стає пошуком параметрів максимальної правдоподібності для даних у наборі ймовірнісних моделей, визначених вибраною архітектурою. Логарифмічну функцію правдоподібності для набору даних $X_N = \{x_1, \dots, x_N\}$ можна записати так:

$$\mathcal{L}(X_N, \theta) = \log p\,(X_N; \theta) = \log \prod_{n=1}^{N} p(x_N; \theta) = \sum_{n=1}^{N} p(x_N; \theta) \qquad (2.7)$$

Знайти параметри максимальної правдоподібності для щільності одного компонента легко і часто це можна зробити в аналітичному вигляді. Це стосується, наприклад, компонентів нормальної суміші. Однак, якщо ймовірнісна модель є сумішшю, оцінка часто стає значно складнішою, оскільки логарифмічна правдоподібність як функція параметрів може мати багато локальних екстремумів. Отже, для отримання оцінок параметрів необхідна деяка нетривіальна оптимізація, і тут добре спрацьовує ітераційний EM-алгоритм (expectation-maximization algorithm). EM-алгоритм знаходить параметри в локальних екстремумах логарифмічної функції правдоподібності, заданих деякими початковими значеннями параметрів і, як буде показано нижче, для нормального розподілу можна отримати аналітичний вигляд параметрів μ та Σ для нормального розподілу.

2.1.3. Вибір базового розподілу суміші

Суміш розподілів - це суміш двох або більше ймовірнісних розподілів, зазвичай з одного сімейства. Випадкові змінні беруться з однієї батьківської популяції до створення нового розподілу. Батьківські популяції можуть бути одновимірними або багатовимірними і повинні мати однакову розмірність. Розподіли можуть складатися з різних розподілів (наприклад, нормальний розподіл та розподіл Стьюдента) або з одного й того ж розподілу з різними параметрами. Нові розподіли ймовірностей розглядаються як справжні функції щільності ймовірності і тому можуть використовуватися для знаходження очікуваних значень, оцінок максимальної правдоподібності та інших статистичних даних.

Суміш нормальних розподілів найчастіше використовується для моделювання неперервних даних [165]. Перша причина такої популярності полягає в тому, що

оцінка параметра максимальної правдоподібності може бути виконана в закритій формі і вимагає лише обчислення середнього значення даних і коваріації. Друга причина полягає в тому, що з усіх щільностей із певною дисперсією щільність Гауса має найбільшу ентропію [166].

Можна виділити основні сімейства суміші розподілів.

- Пуасонові суміші [167, 168]

$$P_g(x|\phi) = \int_0^\infty \frac{e^{(-\theta)}\theta^x}{x!} dG(\theta|\phi).$$

- Суміші експоненційних розподілів [169, 170]

$$F(t) = \int_0^\infty (1 - e^{-xt}) dH(x).$$

- Суміші Вейбулла

$$S(x) = \sum_{i=1}^{n} w_i e^{(-b_i x^{c_i})}, x > 0,$$

де $a_i \in R, i = 1, \dots,$ та $\sum_{i=1}^{n} w_i = 1$

- Суміш нормальних розподілів [165]

$$p(x) = \sum_{s=1}^{k} \pi_s p(x|\mu_s; \theta_s), \quad (2.8)$$

де $p(x|\mu_s; \theta_s)$ – щільність багатовимірного нормального розподілу з R^d та параметрами $(\mu_s; \theta_s)$.

2.1.4. EM-алгоритм оцінки параметрів суміші

Оцінку параметрів розподілу суміші легко провести за допомогою EM-алгоритму [171]. Технічно ідея алгоритму полягає в ітераційному визначенні нижньої межі логарифмічної ймовірності та максимізації цієї нижньої межі. Алгоритм на прикладі суміші нормальних розподілів (2.8) виконує Е-крок, оцінюючи, до якого компонента належить кожна точка даних, і М-крок переоцінює параметри на основі цієї оцінки.

Е-крок. Обчислення допоміжних величин:

$$r_{ij} = \frac{w_i p(x_i|\mu_j;\theta_i)}{\sum_{s=1}^{k} w_s p(x_j|\mu_s;\theta_j)}, \tag{2.9}$$

де r_{ij} – ймовірність того, що об'єкт x_i був отриманий з j-ї компоненти суміші при поточному наближенні параметрів π_i, θ_i.

М-крок. Переоцінка нового наближення параметрів суміші:

$$\begin{cases} w_j = \dfrac{1}{n}\sum_{i=1}^{n} r_{ij}\,, \\ \mu_i = \dfrac{\sum_{i=1}^{n} r_{ij} \cdot x_i}{\sum_{i=1}^{n} r_{ij}}, \\ \theta_i = \dfrac{\sum_{i=1}^{n} r_{ij}(x_i - \mu_j)^T (x_i - \mu_j)}{\sum_{i=1}^{n} r_{ij}}. \end{cases} \tag{2.10}$$

Критерій зупинки. Ітерації методу здійснюються до збіжності варіаційної нижньої оцінки на логарифмічну функцію правдоподібності моделі або іншої цільової функції. Розглянемо критерій зупинки, який будується на основі логарифміної функції правдоподібності, що має наступний вигляд

$$\mathcal{L}(w, \mu, \theta, r|X) =$$

$$\sum_{i=1}^{n}\sum_{j=1}^{k} r_{ij}\left(\ln w_j + \ln p\,(x_i|\mu_i;\theta_i) - \sum_{i=1}^{n}\sum_{j=1}^{k} r_{ij}\ln r_{ij}\right). \tag{2.11}$$

Під збіжністю можна розуміти, наприклад, зміну не більше ніж на заздалегідь задану точність $\varepsilon > 0$. Для спрощення критерію зупинки можна використовувати безпосередньо збіжність самих параметрів суміші, тобто збіжність (w, μ, θ) у відповідному нормованому просторі.

Оновлення середнього значення використовує нову вагу змішування, а оновлення коваріаційної матриці – також нове середнє значення μ та ваги w.

Нехай $P(\theta; X_{1:k}, y_{1:k})$ – розподіл гіперпараметрів нейронної мережі на основі значень цільової функції, отриманої на основі k епох, де X_k – значення

гіперпараметрів на k-му кроці, y_k – значення цільової функції на k-му кроці. Тоді алгоритм еволюційної стратегії на основі розширеного CMA можна описати наступними кроками:

1. Визначення області зміни гіперпараметрів (a_0), розмірності суміші (n), кількості генів в генетичному алгоритмі (N), точності методу (ε).
2. Задання випадковим чином початкових значень гіперпараметрів нейронної мережі $\left(w^{(0)}, \mu^{(0)}, \theta^{(0)}\right)$.
3. Вибір N генів X_k відповідно до розподілу (2.8) та обчислень значень цільової функції y_k.
4. Перерахунок параметрів $\left(w^{(k+1)}, \mu^{(k+1)}, \theta^{(k+1)}\right)$ на основі формул (2.10).
5. Якщо задовольняється умова виходу

$$|\mathcal{L}_{k+1} - \mathcal{L}_k| < \varepsilon,$$

то перейти до виконання генетичного алгоритму на основі розподілу гіперпараметрів з розподілом $p(\theta) = P(\theta; X_{1:k}, y_{1:k})$. Якщо

$$|\mathcal{L}_{k+1} - \mathcal{L}_k| \geq \varepsilon,$$

то перейти до кроку 3.

Слід зауважити, що на кроці 3 виконується одна епоха генетичного алгоритму, тому поряд із оптимізацією параметрів w, μ, θ шукається і оптимальне значення гіперпараметів. Таким чином, розроблений алгоритм одночасно із покращенням оптимального значення цільової функції покращує розподіл гіперпараметрів розподілу, ґрунтуючись на оптимальних хромосомах, змодельованих на основі розподілу $P(\theta; X_{1:k}, y_{1:k})$. Один із основних недоліків даного алгоритму полягає у заданні розмірності суміші n, що може призвести до аналогічної проблеми із випадком $n = 1$. Проблему налаштування параметру n буде розглянуто нижче із використанням підходів кластерного аналізу.

2.1.5. Аналіз на основі методу Монте- Карло

Як було відзначено вище, в даній роботі буде здійснено аналіз деяких класичних генетичних алгоритмів та розширеного CMA-ES алгоритму, описаного вище. Основною новизною даного алгоритму є заміна однопікових (одномодальних) класичних розподілів CMA-ES на суміші розподілів з невизначеними коефіцієнтами, проте фіксованою кількістю піків (розмірністю суміші). Як було зазначено вище, основним критерієм якості алгоритму буде кількість викликів цільової функції f без врахування інших характеристик оптимального значення. Вибір цього критерію є визначальним, з нашої точки зору, із врахуванням складності обчислення параметрів складних систем. У роботі [172] проведено аналіз складності обчислень деяких класичних нейронних мереж, включаючи рекурентні нейронні мережі (RNN), згорткові нейронні мережі (CNN) та нейронні мережі, засновані на принципі довгої короткочаної пам'яті (LSTM). Як було відзначено авторами цієї роботи, складність оцінки параметрів всіх вище зазначених нейронних мереж буде мультиплікативною функцією від усіх розмірностей нейронної мережі. Наприклад, складність для LSTM нейронної мережі буде визначатися, як $O\left(n_s N_r n_i n_o n_n n_g\right)$, де n_s – розмірність вхідної послідовності, n_i – кількість факторів в нейронній мережі, n_n – кількість шарів в нейронній мережі, n_f – кількість фільтрів (функцій активацій) в нейронній мережі, n_o – кількість вихідних нейронів, N_r – кількість прихованих одиниць 'резервуара' нейронної мережі. Таким чином, задача обчислення параметрів нейронної мережі при відомих гіперпараметрах вже є сама по собі достатньо трудомісткою задачею з огляду на кількість операцій, потрібних для оцінки параметрів. Тому, на нашу думку, кількість викликів цільової функції є одним із найбільш важливих факторів, які повинні мінімізуватися під час використання метаевристичних алгоритмів.

Основним підходом для порівняння метаевристичних алгоритмів оптимізації в даному підрозділі буде використано метод Монте-Карло, в якому параметри будуть одинаковими для всіх розглянутих алгоритмів. Для проведення порівняльного аналізу

розглянемо декілька класичних еволюційних алгоритмів пошуку оптимального значення для відомих функцій з відомими екстремумами і порівняємо отримані результати із відповідними результатами для розширеного алгоритму CMA-ES, описаного вище. У якості класичних алгоритмів будемо використовувати наступні метаевристичні алгоритми оптимізації:

- ALO (Ant Lion Optimization) – це метаевристичний алгоритм, що імітує механізм полювання мурашиних левів у природі. Алгоритм має п'ять основних процесів у процесі полювання на здобич: установка пасток, упіймання мурах, очікування здобичі, виявлення та затримання здобичі. Математична модель ALO включає такі кроки, як ініціалізація, фаза руху, фаза оновлення, завершення [173]. Рух мурашиних левів моделюється наступним рівнянням:

$$x_{i,j}^{t+1} = x_{i,j}^{t} + \frac{rand()}{|D_{i,j}|} \times \left(x_{best,j} - x_{i,j}^{t}\right),$$

де $x_{i,j}^{t}$ - положення i -го мурашиного лева в j -му вимірі в момент часу t ,

$x_{best,j}$ - найкраща позиція рою в j -му вимірі, $D_{i,j}$ — це діапазон j -го виміру, а $rand()$— випадкове число від 0 до 1. Алгоритм повторюється доти, доки не буде виконано критерій зупинки.

- ABCO (Artificial Bee Colony Optimization) — алгоритм, заснований на поведінці медоносних бджіл, які шукають джерела їжі та повідомляють про її місцезнаходження іншим бджолам. Алгоритм використовує набір правил для моделювання поведінки медоносних бджіл. Модель складається з трьох основних компонентів: зайнятих та безробітних бджіл-годувальників та джерел їжі [174]. Алгоритм ABCO включає наступні етапи: ініціалізація, фаза працевлаштування бджіл, фаза бджіл-спостерігачів, фаза бджіл-розвідників, завершення. Бджола оновлює свій стан на основі рівняння:

$$x_{i,j}^{t+1} = x_{i,j}^{t} + \phi_{ij} \times (x_{i,j}^{t} - x_{k,j}^{t})$$

$$x_{i,j}^{t+1} = x_{i,j}^{t} + \phi_{ij} \times (x_{i,j}^{t} - x_{k,j}^{t}),$$

де $x_{i,j}^t$ — положення i-ї бджоли-глядача в j-му вимірі в момент часу t, ϕ_{ij} — випадкове число між -1 і 1, а $x_{k,j}^t$ — це положення випадково вибраної бджоли, що працює, відмінної від i, в j-му вимірі в момент часу t. Алгоритм завершує роботу при виконанні критерію зупинки, наприклад, при досягненні максимальної кількості ітерацій або пошуку задовільного рішення.

- GA (Genetic Algorithm) заснований на принципах генетики й еволюції та використовує набір правил для моделювання процесу природного відбору, який підтримує популяцію потенційних рішень (окремих особин) та розвиває їх протягом поколінь. Оператори відбору, схрещування та мутації використовуються для відтворення та генетичного розмаїття. популяції P, що складається з N особин, кожна з яких є потенційним розв'язком задачі оптимізації. Населення можна уявити як $P = \{x_1, x_2, \dots, x_N\}$ представлення x_i особи в популяції, що часто кодується як хромосома. Надалі особини відбираються для відтворення з урахуванням їх показників пристосованості, і алгоритм повторюється до того часу, доки не буде виконано критерій зупинки [175].
- PSO (Particle Swarm Optimization) – це алгоритм роєвого інтелекту, який моделює соціальну поведінку зграї птахів чи зграйної риби. Алгоритм починається із сукупності частинок, і кожна частинка являє собою потенційний розв’язок проблеми оптимізації. Частинки рухаються через простір пошуку залежно від своїх власного кращого становища і кращого становища рою. Алгоритм має наступну математичну модель: ініціалізація, оцінка, оновлення швидкості, оновлення позиції, завершення. Оновлення швидкості моделюється рівнянням:

  $v_{i,j}^{t+1} = w \times v_{i,j}^t + c_1 \times rand(\ \ ) \times \left(pbest_{i,j} - x_{i,j}^t\right) + c_2 \times rand(\ \ ) \times \left(gbest_j - x_{i,j}^t\right)$,

  де $v_{i,j}^t$ — швидкість i-ї частки в j-му вимірі в момент часу t, w — інерційна вага, c_1 і c_2 — коефіцієнти прискорення, $rand()$— випадкове число від 0 до 1, $pbest_{i,j}$

— найкраще положення i-ї частки j-м вимірі, а $gbest_j$ — найкраща позиція рою в j-му вимірі. Алгоритм завершує роботу при виконанні критерію зупинки [176].

- CMA-ES – алгоритм є особливим видом стратегії чисельної оптимізації, що належить до сімейства стратегій еволюції (ES). Алгоритм CMA-ES – це стохастичний алгоритм оптимізації без похідних, який широко застосовується до неопуклих і нелінійних завдань оптимізації, особливо у безперервній галузі. Алгоритм має таку математичну модель: ініціалізація; оцінка рішень; вибір рішень-кандидатів; адаптація коваріаційної матриці обраних рішень-кандидатів; генерація нових рішень-кандидатів, що генеруються з адаптованого пошукового розподілу; завершення [177]. Процес адаптації моделюється наступним рівнянням:

  $C^{t+1} = \left(1 - c_1 - c_\mu\right)C^t + c_1 \times p_c \times (y^t - \bar{y}^t)(y^t - \bar{y}^t)^T + c_\mu \times \sum_{i=1}^{\mu} w_i^t\,(y^t - \bar{y}^t)(y^t - \bar{y}^t)^T,$

  де C^t — коваріаційна матриця в момент часу t, c_1 і c_μ — швидкість навчання, p_c — параметр оновлення першого рангу, y^t — найкращий кандидат у момент часу t, $\bar{y}^t$ — середнє значення вибраних рішень-кандидатів у момент часу t, μ — кількість обраних рішень-кандидатів, а w_i^t — вага i-го обраного кандидата в момент часу t.

Основна відмінність між алгоритмами ALO, ABCO, GA, PSO та CMA-ES полягає в тому, як вони моделюють поведінку тварин або природні процеси та способі пошуку оптимального рішення. Ще одна відмінність – PSO немає генетичних операторів, таких як схрещування та мутація, які присутні у GA та CMA-ES. Кожен алгоритм має свої сильні і слабкі сторони, і вибір алгоритму залежить від конкретної задачі оптимізації.

Як було зазначено вище, у подальшому аналізі будуть використані функції із відомими екстремумами. Для порівняння будемо використовувати метод Монте-Карло із $I = 10^4$, симуляціями для наступних функцій, розглянутих в роботі [178]:

$$f_{T1} = \frac{\sin\left(\sum_{i=1}^{n} x_i^2\right)}{\sum_{i=1}^{n} x_i^2}, \tag{2.12}$$

$$f_{T2}(x) = 100(x_2 - x_1^2)^2 + (1 - x_1)^2, x \in R \tag{2.13}$$

$$f_{T3}(x) = \sum_{i=1}^{3} \left(100\left(x_{i+1} - x_i^2\right)^2 - (1 - x_i)^2\right), x \in R^4 \tag{2.14}$$

$$f_{T4}(x) = I_{\{A_1\}}\left(g(x_1) + h(x_1) + j(x_1)\right) + \left(1 - I_{\{A_1\}}\right)x_2^2 + 1000I_{\{A_2\}}\left(g(x_2) + h(x_2) + \right.$$
$$\left. + j(x_2)\right), x \in R^2 , \tag{2.15}$$

де $I_{\{A\}}$– індикатор події A множини A_1 та A_2 задаються наступним співвідношенням

$$A_i = \left\{0.2\left[\frac{x_i}{0.2}\right] - 0.5 < x_i < 0.2\left[\frac{x_i}{0.2}\right] + 0.5\right\},$$

Функції g, h, j визначаються співвідношеннями

$$g(x_i) = 0.15(0.2\left[\frac{x_i}{0.2}\right] - 0.5)^2 I_{\left\{\left[\frac{x_i}{0.2}\right] > 0\right\}},$$

$$h(x_i) = 0.15(0.2\left[\frac{x_i}{0.2}\right] + 0.05)^2 I_{\left\{\left[\frac{x_i}{0.2}\right] < 0\right\}},$$

$$j(x_i) = x_i^2 I_{\left\{\left[\frac{x_i}{0.2}\right] = 0\right\}}.$$

$$f_{T5}(x) = \sum_{t=1}^{20} (y_t - x_1 - x_2 o_{t2} - x_2^2 o_{t3})^2, \tag{2.16}$$

де $y_t, o_{t1}, o_{t2}, o_{t3}$ – фінансові показники, розглянуті в [177];

$$f_{T6}(x) = \sum_{t=1}^{T} \left\{y_t \log\left(F(x_t, b)\right) + (1 - y_t)\log(1 - F(x_t, b))\right\}, \tag{2.17}$$

де $F(x_t, b)$ визначає емпіричну функцію розподілу, що залежить від невідомого параметра $b \in \{0,1\}$,

$$f_{T7}(x) = \sum_{t=1}^{n} (2y_i - 1) sgn(x_i b) \tag{2.18}$$

Більш широкий список тренувальних функцій (30 функцій) проілюстровано в роботі [137], проте основним показником ефективності роботи алгоритму є середньоквадратичне відхилення знайденого оптимального значення функції. Вибір саме такого показника у даній роботі зрозумілий, оскільки для всіх 30 тестових функцій за рахунок простоти обчислення є можливість нехтувати складністю обчислень самої цільової функції, що, як зазначалося вище, неможливо для більш складних систем, таких як нейронні мережі.

Основним показником швидкодії алгоритму у нашому дослідженні буде кількість звернень до функції, для якої шукаємо максимум. Також вище було зазначено, що складність алгоритму напряму буде залежати від розмірності суміші n, що стане основою дискусії в наступних підрозділах даного розділу. Тому для проведення більш чіткого порівняльного аналізу буде розглянуто декілька розмірностей суміші, що дозволить зрозуміти природу впливу даного чинника на швидкість та якість пошуку оптимального значення цільової функції Надалі через CMA1, CMA2, CMA5, CMA10 будемо позначати результати пошуку оптимального значення цільової функції для розширеного CMA-ES алгоритму з 1, 2, 5 та 10 піками відповідно.

Згідно із роботою [137], розглянемо спочатку функцію (2.12) для різних значень розмірності n, і, як буде показано нижче, розмірність вихідної задачі і буде одним із ключових факторів оптимізаційної задачі. Згадані вище генетичні алгоритми було програмно реалізовано за допомогою мови Python з використанням модуля ***cma***. Результати моделювання на основі методу Монте- Карло наведено на рисунку 2.2. Як ми можемо бачити з даного рисунку, з ростом кількості піків для розширеного CMA-ES алгоритму кількість звернень до цільової функції зростає повільніше в порівнянні з іншими алгоритмати для 5 та 10 піків. Ще одним цікавим фактом, який простежується на даному рисунку, є те, що кількість звернень до цільової функції практично не змінюється для класичних алгоритмів при великих розмірностях, а саме $n \geq 20$. Цей факт може бути корисним при дослідженні цільових функцій з розділеними змінними або можливістю використання аналогів EM-алгоритмів для розщеплення змінної та покроковим покращенням за допомогою ітераційних методів [179]. Для малих розмірностей використання великої розмірності суміші є недоцільним, оскільки зумовлює перерахунок параметрів суміші (w, μ, Θ) для точок, які не є локальними екстремумами. У зв'язку із вищесказаним, одержуємо додаткову задачу визначення оптимальної розмірності суміші, яка б 'найкраще' наближала кількість локальних екстремумів цільової функції. Підхід до оцінки локальних

екстремумів може ґрунтуватися на різних припущеннях, проте можна користуватися класичними результатами кластерного аналізу щодо оцінки кількості кластерів для великих наборів даних . Згідно з основними принципами кластерного аналізу, найбільш поширеними є дві оцінки кількості кластерів, а саме

$$k \approx \ln(N)$$

або

$$k \approx N^{\gamma}, \gamma \ll 1,$$

де N – кількість даних, для яких проводиться кластеризація. Для розглянутих нами алгоритмів N будемо ототожнювати із кількістю хромосом, що дає змогу однозначно оцінювати розмірність суміші на основі першого алгоритму, а саме $k \approx \ln(N)$.

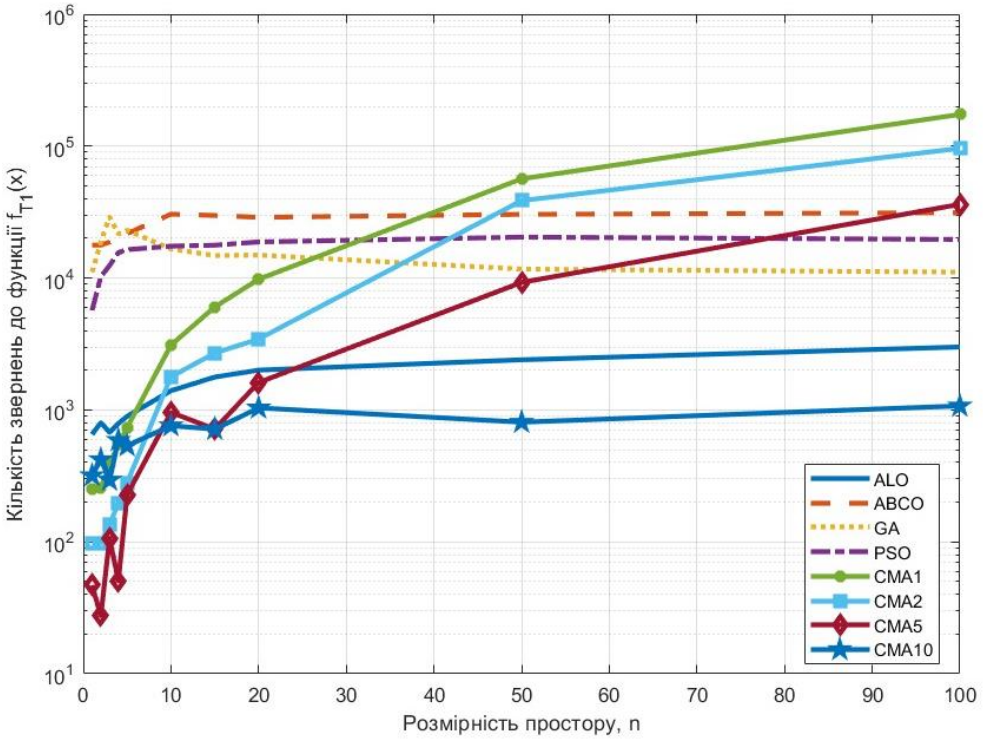


Рис. 2.2 Залежність середніх значень кількості звертань до функції f_{T1} та розмірності простору n

Розглянемо тепер більш детально аналіз оптимізації цільової функції f_{T1} за допомогою оцінки середнього значення та середньоквадратичного відхилення для кількості звернень до f_{T1} – таблиця 2.1. З даної таблиці можна відзначити, що середнє квадратичне відхилення значно нижче для CMA5 та CMA10 для великих значень

розмірності n, що, в свою чергу, підтверджує ефективність розширеного CMA-ES алгоритму. Таким чином підтверджується наше спостереження щодо розмірності початкової оптимізаційної задачі та розмірності суміші в розширеному CMA-ES алгоритмі. Початковий підбір $k \approx \ln(N)$, обговорений вище, для даної задачі видається заниженим, оскільки, згідно з нашим припущенням, оптимальна кількість кластерів повинна бути оцінена як $k \approx 5$, що не підтверджується оціночними значеннями, отриманими в Таблиці. 2.1.

Таблиця. 2.1.

Середні значення кількості звернень до функції f_{T1} із середньоквадратичними відхиленнями (в дужках) при $I = 10^4$ симуляцій

	ALO	**ABCO**	**GA**	**PSO**	**CMA1**	**CMA2**	**CMA5**	**CMA10**
1	653.29 (40.29)	17760.57 (2.86)	11053.35 (407.69)	5670 (289.13)	253.434 (23)	97.85 (28.4)	**46.9** (31.9)	317.69 (52.1)
2	801.12 (43.55)	17770.57 (172.57)	18812.8 (3558.68)	10260 (2740.64)	255.714 (23.05)	98.28 (35.1)	**27.5** (33.7)	420.36 (61.9)
3	676.17 (42.08)	18711.64 (425.82)	29092.2 (10125.89)	12445.0 (4250.23)	390.201 (31.48)	135.15 (38.4)	**106.77** (40.9)	292.2 (72.3)
4	788.62 (48.28)	19875.02 (548.51)	21582.25 (6503.63)	15577.5 (5007.7)	545.424 (62.09)	196.57 (52.4)	**50.47** (51.8)	588.93 (73.4)
5	895.02 (52.92)	21722.72 (1062.06)	23019.37 (7686.432)	16417.5 (5592.08)	735.248 (218.27)	280.97 (73.6)	**227.32** (82.3)	535.98 (93.6)
10	1398.5 (152.4)	30482.31 (7233.62)	16607.22 (4432.16)	17452.5 (8792.08)	3099.11 (575.27)	1783.78 (129.9)	956.94 (145.7)	**759.67** (98.4)

Продовження таблиці 2.1.

15	1775.5 (403.29)	29811.17 (9955.86)	14805.83 (4016.91)	17740.0 (8116.33)	6035.376 (878.96)	2697.28 (152.5)	716.63 (182.1)	**712.67** (129.3)
20	2004 (611.56)	28858.22 (5285.7)	14930.58 (4202.75)	18782.5 (6876.74)	9791.484 (1542.77)	3449.49 (430.87)	1612.28 (287.5)	**1036.53** (256.8)

50	2400.5 (705.3)	30397.71 (8156.89)	11687.08 (1231.79)	20420.0 (8866)	56512.825 (12220.98)	38626.37 (1002.4)	9267.20 (503.4)	**807.64** (383.7)
100	3000.8 (806.23)	31289.91 (9665.79)	11083.29 (608.27)	19605 (9676.14)	173977.32 (76227.7)	96187.69 (1985.4)	36216.1 (1498.1)	**1072.23** (789.3)

Ще одним важливим фактом, на який варто звернути увагу в Табл. 2.1, є значення середньоквадратичних відхилень. Для розмірностей цільової функції $n \geq 10$ CMA10 показує оптимальніші результати із меншими значеннями середньоквадратичних відхилень. Даний факт знову ж таки вказує на правильність припущень щодо зв'язку розмірності вихідної задачі та розмірності суміші.

Проаналізуємо також відповідні значення функцій $f_{T2} - f_{T7,}$ наведеними в таблиці 2.2. Як ми можемо бачити з даної таблиці, зберігається тенденція щодо вибору кількості піків для розширеного CMA-ES алгоритму, а саме для задач більшої розмірності кількість піків повинна бути більшою. Дані властивості наштовхують на існування взаємозалежності між розмірністю початкової задачі та кількістю вибраних піків. Слід зауважити, що лише для однієї функції f_{T3} класичний алгоритм показав кращий результат, ніж розроблений нами алгоритм. Це вказує на існування додаткових припущень на цільову функцію, які можуть 'порушити' правильність роботи розробленого алгоритму.

Таблиця. 2.2.

Середні значення кількості звернень до функцій $f_{T2} - f_{T7}$ із середньоквадратичними відхиленнями (в дужках) при $I = 10^4$ симуляцій.

	ALO	ABCO	GA	PSO	CMA1	CMA2	CMA5	CMA10
f_{T2}	1053.2 (321.24)	26868.59 (9302.66)	16740.95 (5551.51)	11557.5 (4412.91)	587.4 (87.9)	**478.1** (83.1)	543.1 (93.4)	567.4 (132.1)
f_{T3}	**2012.3** (615.24)	22473.09 (4393.99)	27562.27 (13830.33)	14448.25 (6771.35)	11933.4 (148.07)	8560.23 (200.2)	7932.23 (200.2)	9678 (298.45)
f_{T4}	749.875 (57)	17802.31 (5.74)	12287.37 (740.11)	7875.25 (1620.49)	135.73 (32.44)	**111.34** (43.12)	189.86 (86.31)	231.81 (92.84)
f_{T5}	1290.7 (462.86)	38657.57 (7813.46)	25345.70 (8129.27)	11658.5 (4513.03)	519.49 (42.84)	456.11 (42.29)	387.83 (51.7)	**322.4** (53.11)
f_{T6}	575 (3.78)	17850.4 (3.52)	10729 (2.56)	5500 (7.95)	20.46 (12.63)	22.19 (22.84)	20.32 (13.65)	**19.28** (12.56)
f_{T7}	575 (2.1)	17850.6 (2.22)	24631.14 (7196.34)	5500 (9.67)	20.79 (12.83)	21.75 (18.12)	19.4 (22.31)	**18.32** (25.63)

Таким чином, у даному підрозділі проведено порівняльний аналіз метаевристичних алгоритмів оптимізації оцінки параметрів складних систем; запропоновано розширений алгоритм еволюційної стратегії з адаптацією коваріаційної матриці для оцінки параметрів складних систем. Крім того, розширений CMA-ES базується на ідеї використання сумішей розподілів із невизначеною величиною розмірності суміші та з відомим базовим розподілом. Дослідження показують, що з ростом кількості піків для розширеного CMA-ES алгоритму кількість звернень до цільової функції спадає і таким чином підтверджує ефективність запропонованого розширеного алгоритму CMA-ES. Однак для малої розмірності n вибір великої кількості піків є недоцільним. Дані властивості наштовхують на існування взаємозалежності між розмірністю початкової задачі та кількістю вибраних піків, що буде предметом дослідження наступних підрозділів роботи

2.2. Самоадаптивний CMA-ES алгоритм

У даному підрозділі роботи буде розглянуто один із самоадаптивних алгоритмів підбору параметрів складних систем, прикладами яких є нейронні мережі. Як було зазначено вище, використання класичних метаевристичних алгоритмів є недоцільним для складних систем, оскільки зумовлює критичне збільшення обчислень цільової функції. Розглянемо класичне означення самоадаптивного алгоритму, згідно якого самоадаптивні алгоритми – це алгоритми, які змінюють свою поведінку під час виконання на основі доступної інформації та заздалегідь визначених механізмів зміни параметрів [138] . Ці алгоритми широко використовуються в різних галузях, включаючи машинне навчання, оптимізацію та стиснення даних. Самоадаптивність алгоритму у даному випадку буде ґрунтуватися на підборі кількості піків у суміші (розмірності суміші) розподілів у розширеному CMA-ES алгоритмі за умови нормального базового розподілу. Зауважимо, що використання саме нормального розподілу дозволить найбільш наочно вказати переваги та недоліки обраного нами алгоритму.

Отже, розглянемо покращений самоадаптивний алгоритм CMA-ES, з акцентом на параметр, що визначає кількість піків у суміші нормальних розподілів. В алгоритмі враховуються методи налаштування даного оптимального значення, що використовується при виборі номерів кластерів в алгоритмах кластеризації CURE, BIRCH тощо. Очевидно, що наведене обґрунтування цього підходу може бути поширене на суміші з іншим базовим розподілом, кожен з яких характеризується скінченним числом піків у суміші розподілів. Це означає самоадаптивність та застосовність алгоритму до ширшого спектру сценаріїв, що включають різні характеристики розподілу.

Немає сумніву у тому, що запропонований садоадаптивний алгоритм налаштування параметрів, що базується на CMA-ES алгоритмі, може бути розширений і на інші генетичні та еволюційні алгоритми, які містять відбір додаткових хромосом (індивідів) при переході між ітераційними епохами алгоритму. Ще однією особливістю запропонованого алгоритму є використання теоретичних

основ кластерного аналізу для оцінки кількості піків у розподілі хромосом. Цей підхід широко використовується у новітніх самоадаптивних алгоритмах для визначення початкових параметрів (гіперпараметрів) складних систем.

У роботах [97, 139, 173, 175, 176, 178] розглянуто різні генетичні алгоритми пошуку оптимального значення для задачі оптимізації

$$f(x) \to extr\ , \tag{2.19}$$

де функція $f: R^n \to R$, причому припускається, що задача оптимізації є безумовною $(x \in R^n)$. Важливу роль у генетичних алгоритмах відіграє метод відбору нових хромосом $x_i \in R^n$. Зрозуміло, що складність алгоритму буде залежати від розподілу хромосом на кожній із ітерацій (епох) еволюційного алгоритму. Надалі щільність розподілу нових "особин" на m–ій ітерації будемо позначати через $p_{new}^{(m)}(x), x \in R^n$. Проблему вибору початкового розподілу $p_{new}^{(0)}(x), x \in R^n$ та переобчислення $p_{new}^{(m)}(x), x \in R^n, m > 0$ детально описано в роботі [180], де увага також сконцентрована на підборі нових хромосом.

Особливу увагу в дисертаційному дослідженні будемо приділяти CMA-ES алгоритму в роботах [96, 97, 130, 133, 139]. Даний алгоритм ґрунтується на припущенні, що $p_{new}^{(m)}(x)$ визначається як багатовимірний нормальний розподіл, тобто

$$p_{new}^{(m)}(x; \mu_m, \sum_m) = (2\pi)^{-\frac{n}{2}} (\det(\sum_m))^{-\frac{1}{2}} e^{-\frac{1}{2}(x-\mu_m)' \sum_m^{-1} (x-\mu_m)},$$

де μ_m – середнє значення розподілу, $\sum_m$ – відповідна коваріаційна матриця. Основна увага при застосуванні CMA-ES алгоритму приділяється саме аналізу коваріаційної матриці $\sum_n$, яка відображає розкид даних від свого середнього значення. Великі значення елементів коваріаційної матриці $\sum_m$ вказують на великий розкид нових хромосом, і в даному випадку нормальний розподіл може бути замінений рівномірним розподілом на деякій обмеженій підмножині з R^n.

2.2.1. Мотиваційний приклад

Розглянемо один приклад використання класичного CMA-ES алгоритму для розв'язання задачі (1) та його недоліки за наявності специфічних умов функції f. Розглянемо наступну оптимізаційну задачу

$$f(x) = \alpha e^{-x\prime x} + (1-\alpha)e^{-(x-\mu)(x-\mu)}, x \in R^2\ , \qquad (2.20)$$

де $a \in [0,1]$, $\mu = (1,1)$. Оптимальні значення безумовної задачі оптимізації (2.19) для функції (2.20) будуть мати вигляд

$$x_{opt} = (\beta, \beta), \beta \in [0,1],$$

причому $\beta = 0$ при $\alpha = 1$ та $\beta = 1$ при $\alpha = 0$. У загальному випадку β задовольняє рівняння

$$\alpha\beta - (1-\alpha)(1-\beta)e^{4\beta-2} = 0.$$

Залежність β та $\alpha \in [0.5,1]$ можемо бачити на рисунку 2.3. Використовуючи симетрію можемо продовжити розв'язок на $\alpha \in [0,0.5]$.

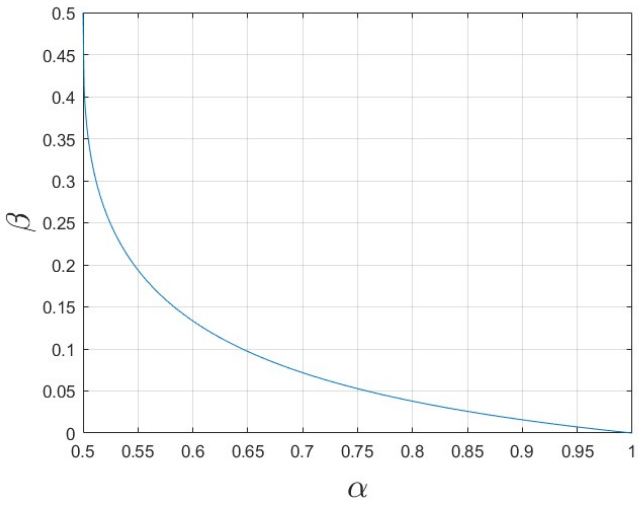


Рис. 2.3. Взаємозв'язок α та β для задачі (2.18) та функції (2.19)

Розглянемо тепер випадок використання еволюційного алгоритму CMA-ES для задачі оптимізації (2.19) з цільовою функцією (2.20) [188]. Використовуючи міркування, аналогічні до міркувань щодо оптимального розв'язку β, робимо висновок, що коваріаційна матриця відбору нових хромосом у CMA-ES алгоритмі буде мати вигляд

$$\sum_m = \begin{pmatrix} \sigma_1^2(m) & \rho\sigma_1(m)\sigma_2(m) \\ \rho\sigma_1(m)\sigma_2(m) & \sigma_2^2(m) \end{pmatrix},$$

де $\rho \in (0,1)$, а значення параметрів $\sigma_1(m), \sigma_2(m) > 0$ будуть залежати від параметра α. Нас буде цікавити значення $\alpha \approx \frac{1}{2}$. У цьому випадку функція $f(x)$ (2.19) буде мати приблизно два піки одного рівня (рис. 2.4). З даного рисунка видно, що для випадку $\alpha = \frac{1}{2}$ буде мати місце рівність $\sigma_1(m) = \sigma_2(m)$ та буде справедлива нерівність

$$\sigma_1(m) = \sigma_2(m) > \sigma_c > 0, \tag{2.21}$$

де σ_c буде визначати нижню межу для середньоквадратичного відхилення $\sigma_1(m)$, $\sigma_2(m)$ на кожній ітерації. Слід зауважити, що оцінка середнього значення μ_m в CMA-ES алгоритмі буде близькою до середини між піками, тобто $\mu_m \approx \left(\frac{1}{2}, \frac{1}{2}\right)$ для достатньо великих n. У загальному ж випадку μ_m буде залежати від параметра α та буде мати місце наближення $\mu_m \approx (\alpha, 1-\alpha)$. Значення граничної величини σ_c повинно бути достатньо великим для того, щоб з близькою до 1 ймовірністю на основі нормального розподілу $p_{new}^{(m)}(x; \mu_m, \sum_m)$ покривалися два піки функції f, тобто (0,0) та (1,1). Врахувавши даний факт, приходимо до висновку, що $\sigma_1(m)$ та $\sigma_2(m)$ будуть лінійними функціями від відстані між піками. У випадку наявності багатьох локальних екстремумів у цільовій функції в R^2 буде мати місце наступне співвідношення :

$$\sigma_1(m) = O(M), \sigma_2(m) = O(M),$$

де M визначає максимальну відстань між локальними екстремумами функції f, що є близькими до глобального екстремуму, тобто

$$M = \max_{x: f(x) \approx max f(x)} |x - argmax\, f(x)|.$$

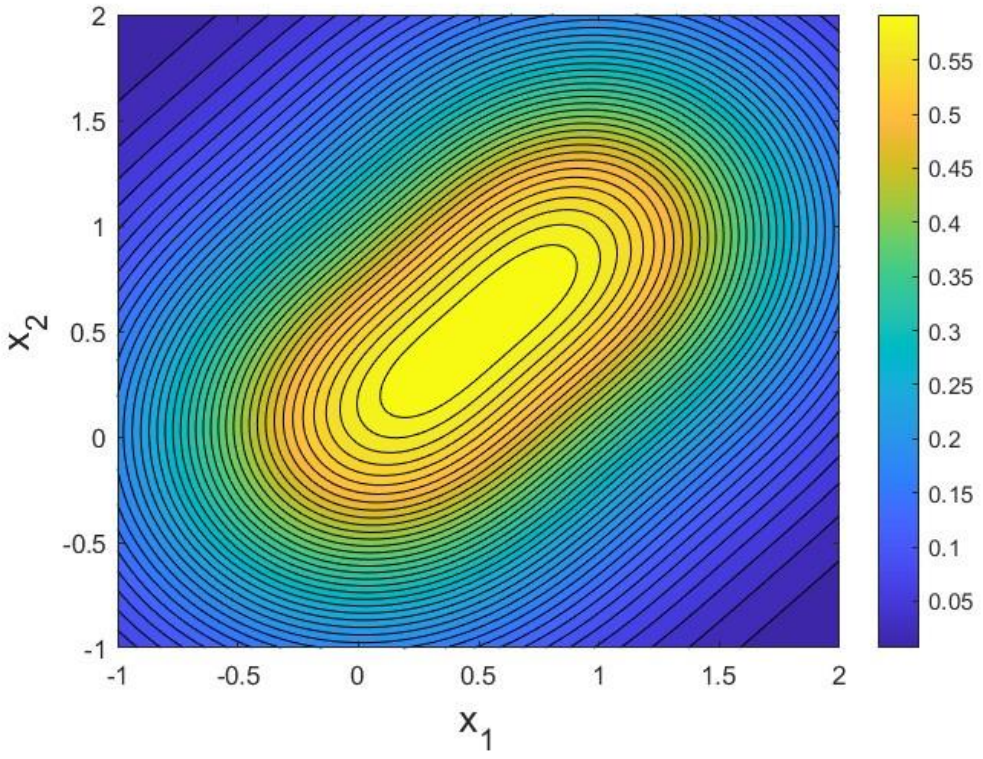


Рис. 2.4. Пошуковий ландшафт для функції (2) при $\alpha = \frac{1}{2}$

Виходячи із наведених вище міркувань, приходимо до висновку, що класичний CMA-ES алгоритм із щільністю вибору на кожній ітерації $p_{new}^{(m)}(x; \mu_m, \sum_m)$ володіє недоліком для цільових функцій із більше ніж одним локальним екстремумом приблизно одного і того ж рівня. Тобто у випадку використання CMA-ES алгоритму даний факт зумовить суттєве збільшення кількісті обчислень цільової функції f, що в деяких випадках має критичне значення. Для подолання цієї проблеми авторами роботи [181] було розроблено розширений CMA-ES алгоритм, який дозволяє враховувати локальні екстремуми цільової функції та локалізовувати кожен із них, враховуючи деякі особливості даного екстремуму. У роботі [181] було запропоновано заміну стандартного нормального розподілу на суміш нормальних розподілів, що визначається наступним чином:

$$p_{new}^{(m)}\left(x; \mu_m, \sum_m, w_m, k(m)\right) =$$

$$= (2\pi)^{-\frac{n}{2}} \sum_{i=1}^{k(m)} w_m^i (\det(\sum_{m,i}))^{-\frac{1}{2}} e^{-\frac{1}{2}(x-\mu_{m,i})' \sum_{m,i}^{-1}(x-\mu_{m,i})}, \qquad (2.22)$$

де $\mu_{m,i}$ та $\sum_{m,i}$ – параметри нормального розподілу в елементі суміші,

$w_m = (w_m^1, \dots, w_m^{k(m)})$ – вагові коефіцієнти суміші, які задовольняють наступні умови:

$$\begin{cases} w_m^i > 0, i = 1, \dots, k(m), \\ \sum_{i=1}^{k(m)} w_m^i = 1, \end{cases}$$

де $k(m)$ – кількість піків у суміші (2.21). Введемо до розгляду наступні позначення:

- N – загальна кількість "особин" (хромосом), що визначені в розширеному CMA-ES алгоритмі. Дане значення не змінюється при зміні ітерації n.
- $f_i^{(m)} = f(x_i)$ – значення цільової функції для хромосоми $x_i, i = 1, \dots, N$ на m-й ітерації;
- найбільш ймовірна щільність, за якою моделюється ген x_i, що визначається співвідношенням

$$a_i^{(m)} = \operatorname{argmax}\begin{pmatrix} \left(\det(\sum_{m,1})\right)^{-\frac{1}{2}} e^{-\frac{1}{2}(x-\mu_{m,1})' \sum_{m,1}^{-1}(x-\mu_{m,1})}, \dots, \\ \left(\det(\sum_{m,k(m)})\right)^{-\frac{1}{2}} e^{-\frac{1}{2}(x-\mu_{m,k(m)})' \sum_{m,k(m)}^{-1}(x-\mu_{m,k(m)})} \end{pmatrix},$$

Визначення даного показника обчислюється на основі ймовірнісної кластеризації із використанням сумішей розподілів [181]:

- Відсоток хромосом, що формуються i-м розподілом в суміші, тобто

$$p_i^{(m)} = \frac{\#\left\{a_j^{(m)} = i, j = 1, \dots, N\right\}}{N}, i = 1, \dots, k(m)$$

Нагадаємо розширений CMA-ES алгоритм, який можна знайти в роботі [181], який спрямований на знаходження оптимального значення оптимізаційної задачі :

1. Визначення області зміни гіперпараметрів системи, розмірності суміші $k(n) = k = const$, кількості генів в генетичному алгоритмі N, точності методу ε, додаткового "параметру сталості" n_{const}.

2. $n = 1$. Задання випадковим чином початкових значень параметрів суміші $(w_n, \mu_n, \sum_n)$.

3. Вибір N генів згідно з розподілом (2.22) з параметрами $(w_m, \mu_m, \sum_m)$ та обчислень значень цільової функції $f_i^{(m)}$.

4. Перерахунок параметрів $(w_{m+1}, \mu_{m+1}, \sum_{m+1})$ на основі формул ЕМ алгоритму та екстремальних значень $f_i^{(n)}$. Визначення найкращої хромосоми, що оптимізує задачу (1) та відповідного значення функції

$$F_m = \max_{i=1,\dots,N} f_i^{(m)}.$$

5. Видалення хромосом з мінімальними значеннями $f_i^{(m)}$ та їх заміщення новими хромосомами на основі суміші (2.21) з параметрами $(w_{m+1}, \mu_{m+1}, \sum_{m+1})$.

6. Використання мутації та схрещування хромосом.

7. Якщо задовольняється умова виходу

$$\left|F_n - F_{n-n_{const}}\right| < \varepsilon,$$

то оптимальний розв'язок знайдено з максимальним значенням функції F_n та хромосомою, що відповідає даному оптимальному значенню.

Якщо $\left|F_m - F_{m-n_{const}}\right| \geq \varepsilon$, то перейти до кроку 2.

Особливу увагу приділимо саме оцінці параметра $k(m)$, за допомогою якого вдається локалізувати оцінку локальних мінімумів функції. Основним завданням у цьому розділі буде визначення умов, за яких збільшуватиметься чи зменшуватиметься кількість піків $k(m)$ на кожній ітерації.

Враховуючи дані показники, розглянемо алгоритм створення додаткового піку у суміші, тобто збільшення $k(m)$, та алгоритм видалення піку, тобто зменшення $k(m)$ на одиницю. На прикладі, наведеному вище, можна стверджувати, що $k(m)$ повинно бути збільшеним у тому випадку, коли для відповідної коваріаційної матриці $\sum_{m,i}$ у суміші (2.21) не відбувається затиснення до локального екстремуму, тобто

$$\lim_{n\to\infty} \det(\sum_{n,i}) > 0$$

У цьому випадку пік суміші (2.22), що відповідає параметрам $(\mu_{m,i}, \sum_{m.i})$, розбиватиметься на два піки, причому розбиття буде проводитися за допомогою класичного ЕМ алгоритму, описаного в роботі [182]. У випадку ж видалення піку або зменшенням значення величини $k(m)$ будемо користуватися методологією алгоритмів кластеризації великих даних CURE, BIRCH [183, 184, 185, 186, 187]. Згідно з даними алгоритмами кластерами є достатньо великі підгрупи множин, кількість елементів у яких приблизно рівна $\frac{N}{k(n)}$ або більша даного числа. В іншому випадку множина не враховується як повноцінний кластер і вважається множиною приєднання. За аналогією будемо зменшувати значення $k(m)$ у тому випадку, якщо $p_i^{(m)} \ll \frac{N}{k(m)}$. Використовуючи дані алгоритми зменшення та збільшення $k(m)$ [189], отримаємо наступний самоадаптивний розширений алгоритм CMA-ES:

1. Визначення області зміни гіперпараметрів системи, початкову розмірність суміші $k(1) = k = O(\ln(N))$, кількості генів в генетичному алгоритмі N , точності методу ε, додаткового "параметру сталості" n_{const}.

2. $m = 1$. Задання випадковим чином початкових значень параметрів суміші $(w_m, \mu_m, \sum_m)$.

3. Вибір N генів згідно з розподілом (2.21) з параметрами$(w_m, \mu_m, \sum_m)$ та обчисленнями значень цільової функції $f_i^{(m)}$.

4. Перерахунок параметрів $(w_{m+1}, \mu_{m+1}, \sum_{m+1})$ на основі формул ЕМ алгоритму та екстремальних значень $f_i^{(m)}$. Визначення найкращої хромосоми, що оптимізує задачу (2.19), та відповідного значення функції

$$F_m = \max_{i=1,\dots,N} f_i^{(m)}.$$

Видалення хромосом з мінімальними значеннями $f_i^{(m)}$ та їх заміщення новими хромосомами на основі суміші (2.22) з параметрами $(w_{m+1}, \mu_{m+1}, \sum_{m+1})$.

5. Використання мутації та схрещування хромосом.

6. Якщо $det(\sum_{m,i})$ не змінюються протягом n_{const} ітерацій, то збільшуємо $k(m)$ на одиницю за рахунок створення суміші з $m = 2$ з нормального розподілу на основі хромосом із множини $X_i = \left\{a_j^{(m)} = i, j = 1, \dots, N\right\}$. Переходимо до наступного кроку.

7. Якщо $p_i^{(m)} \ll \frac{N}{k(m)}$ для деякого i, то видаляємо хромосоми, що відповідають даному піку, тобто $X_i = \left\{a_j^{(n)} = i, j = 1, \dots, N\right\}$, та зменшуємо $k(m)$ на одиницю.

8. Якщо задовольняється умова виходу

$$\left|F_m - F_{m-n_{const}}\right| < \varepsilon,$$

то оптимальний розв'язок знайдено з максимальним значенням функції F_m та хромосомою, що відповідає даному оптимальному значенню.

Якщо $\left|F_m - F_{m-n_{const}}\right| \geq \varepsilon$, то переходимо до кроку 2.

2.2.2. Самоадаптивний алгоритм на обмежених областях

Попередні приклади, розглянуті в підрозділах 2.1 та 2.2 були сконцентровані на задачах пошуку оптимального значення цільової функції $f(x)$ без обмежень на область пошуку функції. Проте переважна більшість прикладних задач, включаючи оптимізації гіперпараметрів нейронних мереж, будуть містити ряд обмежень, які, в свою чергу, накладають додаткові обмеження на базові розподіли, на яких будуються суміші.

Розглянемо загальну оптимізаційну задачу

$$f(x) \to extr \tag{2.22}$$

за умови

$$x \in X \subset R^n. \tag{2.23}$$

Додаткове обмеження (2.23) значно ускладнює оптимізаційну задачу (2.22). Проте в термінах самоадаптивного алгоритму, розглянутого вище, є можливість спростити процес пошуку оптимального значення. Для цього, по-перше, слід розглянути лише ті розподіли $p(x;\theta)$, носієм яких є множина $X \subset R^n$, де θ – параметр розподілу. По-друге, слід розглянути лише однопікові розподіли, тобто такі, для яких існує точка $x_m(\theta) \in X$, така, що

$$p(x_m(\theta);\theta) > p(x;\theta), x_m(\theta) \neq x \in X.$$

Зауважимо, що нормальний розподіл задовольняє описану вище умову, якщо припустити, що $x_m(\theta) = \mu$, де $\theta = (\mu, \Sigma)$ – параметри нормального розподілу із середнім μ та коваріаційною матрицею Σ. Аналогічно на множині $X = \{0,1,2,\dots\}$ може бути розглянуто розподіл Пуассона із параметром $\theta = \lambda$. У цьому випадку $x_m(\theta) = \lfloor\lambda\rfloor$ або $x_m(\theta) = \lfloor\lambda + 1\rfloor$.

2.2.3. Переваги і недоліки самоадаптивного CMA-ES

Переваги розробленого алгоритму були перераховані вище, і основний з них полягає у перерахунку розмірності суміші на основі найкращих хромосом на кожній ітерації. Проблемою для даного алгоритму є наявність великої кількості екстремумів, різнозначних за значеннями. У даному випадку кроки 6 та 7 останнього алгоритму будуть продукувати або велику кількість нових піків, або зменшувати кількість піків за рахунок виконання умови $p_i^{(m)} \ll \frac{N}{k(m)}$. Прикладом функції, для якої розроблений алгоритм працюватиме аналогічно класичним алгоритмам і не буде показувати кращі результати, є функція

$$f(x,y) = \begin{cases} 0, x^2 + y^2 \geq 1, \\ \sin\left(\pi(x^2 + y^2)\right), x^2 + y^2 < 1. \end{cases}$$

Дана функція має нескінченну кількість піків при $x^2 + y^2 = \frac{1}{2}$, а отже, розроблений самоадаптивний алгоритм буде мати приблизну таку ж обчислювальну складність, як класичні алгоритми та класичний CMA-ES.

Висновки до розділу II

У розділі II розглянуто проблему мультимодальності в алгоритмі CMA-ES і для подолання цього недоліку запропоновано розширенний CMA-ES, який використовує суміш нормальних розподілів. Це дозволяє алгоритму враховувати множинні піки цільової функції та знаходити кращі рішення. Описано чотири методи відбору хромосом, що враховують мультимодальність: логарифмічний ріст, прирост логарифмів, натуральне впорядкування та степеневий ріст. Оцінка параметрів суміші нормальних розподілів виконана за допомогою EM-алгоритму (expectation-maximization algorithm). Цей алгоритм знаходить параметри в локальних екстремумах логарифмічної функції правдоподібності.

Використання суміші нормальних розподілів в CMA-ES алгоритмі може значно покращити його ефективність при вирішенні складних задач оптимізації з множинними локальними екстремумами.

Проведено порівняльний аналіз розширеного CMA-ES з класичними генетичними алгоритмами ALO, ABCO, GA, PSO. Для порівняння використовується метод Монте-Карло з 1000 симуляціями для 5 тестових функцій. Порівняння з класичними алгоритмами показує, що чим більше піків у суміші, тим менше звернень до цільової функції потрібно алгоритму CMA-ES. Також розмірність задачі впливає на оптимальну кількість піків. Розширений CMA-ES демонструє кращі результати за середнім та стандартним відхиленням кількості звернень до цільової функції.

Також, на відміну від класичних алгоритмів, розширений CMA-ES підбирає кількість піків у суміші розподілів під час виконання, що дозволяє уникнути зайвих обчислень. Адаптивність базується на методах визначення кількості кластерів, відомих з кластерного аналізу. Загалом, запропонований підхід є перспективним для ефективної оптимізації складних систем.

В алгоритмі враховані методи підбору кількості кластерів, застосовані у алгоритмах кластеризації CURE та BIRCH. Немає сумніву у тому, що дана логіка

може бути розвинена і на суміші із іншим базовим розподілом, які мають скінченну кількість піків. Отже, розширений CMA-ES є ефективним алгоритмом для складних систем, а розмірність суміші слід обирати з огляду на розмірність задачі.

РОЗДІЛ ІІІ.
ВИКОРИСТАННЯ САМОАДАПТИВНИХ АЛГОРИТМІВ В ЗАДАЧАХ ПОШУКУ ОПТИМАЛЬНОГО КЕРУВАННЯ

Даний розділ присвячено деяким питанням теорії керування для стохастичних диференціальних рівнянь із марковськими перемиканнями. Основою даного розділу є твердження, що стосуються існування оптимального керування, причому оптимальність розуміється в термінах оптимізації деякого функціоналу якості. Розвиток дослідження оптимальних керування розпочинається з класичних робіт по аналізу звичайних диференціальних рівнянь [190, 191], де класичний підхід будується на оптимізації функціоналу якості із врахуванням правої частини рівняння. Даний підхід було розвинено в наступних роботах, увага в яких приділяється стохастичним диференціальним рівнянням (СДР) Ito без післядії

$$dx(t) = a\big(t, x(t), u(t)\big)dt + b\big(t, x(t), u(t)\big)dw(t), \qquad (3.1)$$

де $w(t), t \geq 0$ – стандартний вінерівський процес, a, b – вимірні функції своїх аргументів, для яких виконуються умови існування та єдиності розв'язку [192, 193, 194, 195,196]. Основна ідея даних робіт полягає у використанні інфінітезимального оператора (генератора), що відповідає попередньому СДР та визначається співвідношенням

$$A^u f = (\nabla f)^T f + \frac{1}{2} tr\big(J(f)(bb^T)\big), \qquad (3.2)$$

де функція $f \in C^2$. Розглянемо основане твердження, що визначає достатні умови існування оптимального керування для СДР (3.13) з інфінітезимальним оператором (3.2) та функціоналом якості

$$J(t, x; u) = E\left\{\int_t^T q(x(t), u(t))dt + r(x(T))|x(0) = x\right\}. \qquad (3.3)$$

При знаходженні оптимального керування основним підходом вважається використання так званого рівнянням Гамільтона – Якобі – Беллмана (Hamilton–Jacobi–Bellman, HJB).

Теорема 3.1. Нехай виконуються наступні умови:

- Коефіцієнти СДР (3.1) задовольняють умовам теореми існування та єдиності;
- Існує непорожня множина керувань U.

Тоді оптимальне керування $u^{opt} \in U$ задовольняє наступне співвідношення

$$\frac{\partial V(t,x)}{\partial t} + \min_{u \in U}\{A^u V(t,x) + q(x,u)\} = 0, (t,x) \in (0,T) \times R^n \quad (3.4)$$

з крайовою умовою

$$V(T,x) = r(x).$$

де функція $V(t,x)$ визначає значення функціоналу якості при оптимальному керуванні

$$V(t,x) = \min_{u \in U} J(t,x;u),$$

причому $V(t,x)$ задовольняють наступні властивості

- для довільного керування $u \in U$ виконується співвідношення

$$V(t,x) \leq J(t,x;u);$$

- Якщо існує функція $k: [0,T] \times R^n \to R$, така що має місце співвідношення

$$A^{k(t,x)} V(t,x) + q\big(x, k(t,x)\big) = \min_{u \in U}\{A^u V(t,x) + q(x,u)\},$$

то має місце ефект зворотного зв'язку для керування та розв'язку

$$u\big(t, x(t)\big) = k\big(t, x(t)\big). \quad (3.5)$$

Таким чином, дане твердження дозволяє не лише говорити про існування оптимального керування що задовольняє рівняння (3.4), а і спрощувати його пошук на основі додаткових припущень про вигляд керування на основі (3.5). Як буде показано нижче, особлива ефективність теорема 3.1 буде мати у випадку спеціальних випадків функцій $q(x,u)$ та $r(x)$. У випадку так званої лінійно – квадратичної задачі оптимізації, при якій функціонал якості визначається співвідношенням

$$J(t,x;u) =$$

$$= E\left\{\int_t^T \left(x^T(t)Q(t)x(t) + u^T(t)R(t)u(t)\right)dt + x^T(T)Gx(T)|x(0) = x\right\}, (3.6)$$

де $Q(t), R(t),$ – додатньо визначені матричні функції, G – додатньовизначена матриця.

Подальші дослідження в напрямку досліджень оптимального керування можна умовно розбити на два основних напрямки. Перший із даних напрямків ґрунтується на розширенні або зміні властивостей випадкового процесу $w(t)$, що визначає випадковість у СДР (3.13) за рахунок наявності доданку $b\left(t,x(t),u(t)\right)dw(t)$. Розглянемо декілька класичних результатів, що стосуються теорії керування. У роботах [197, 198, 199] основна увага присвячена дослідженню синтезу оптимального керування поряд із розглядом фрактального (дробового) броунівського руху [200], який визначається наступним чином.

Означення 3.1. Фрактальним броунівським рухом (Fractional Brownian motion, fBm) називається центрований Гауссівський процес $w(t) = B^H(t), t \geq 0$ з коваріаційною функцією

$$E\left(B^H(t)B^H(s)\right) = \frac{1}{2}(t^{2H} + s^{2H} - |t-s|^{2H}),$$

де параметр $H \in (0,1)$ називається індексом Херста (Hurst index).

Випадок $H = \frac{1}{2}$ відповідає стандартному вінерівському процесу. У класичних роботах Леві [201] фрактальний броунівський рух визначається через інтегральне співвідношення

$$B^H(t) = \frac{1}{\Gamma\left(\frac{1}{2}+H\right)}\int_0^t (t-s)^{\frac{1}{2}+H}dw(t).$$

Розглянемо одне твердження [202] для лінійних СДР із фрактальним броунівських рухом, що визначаються через співвідношення

$$dx(t) = \left(a(t)x(t) + b(t)u(t)\right)dt + c(t)x(t)dB^H(t), x(t), u(t) \in R^1. \quad (3.7)$$

Тоді має місце наступне твердження для $H > \frac{1}{2}$.

Теорема 3.2. Припустимо, що виконуються наступні умови:

- Коефіцієнти СДР (3.7) задовольняють умовам теореми існування та єдиності;
- $Q(t) > 0, R(t) > \delta > 0, H > 0$;
- Існує непорожня множина керувань U.

Тоді оптимальне керування задається співвідношенням

$$u(t) = -R^{-1}(t)b(t)p(t)x(t), \tag{3.8}$$

де функція $p(t)$ визначається із співвідношення

$$\begin{cases} \dfrac{dp(t)}{dt} + 2p(t)\left(a(t) + c(t)\displaystyle\int_0^t \phi(t,s)c(s)ds\right) + \\ +Q(t) - R(t)b^2(t)p^2(t) = 0, \\ p(T) = G, \end{cases} \tag{3.9}$$

функція $\phi(t,s)$ визначається співвідношенням

$$\phi(t,s) = H(2H-1)|s-t|^{2H-2}, s,t \in [0,\infty).$$

Вигляд (3.8) являться загальним виглядом керування для лінійно – квадратичної задачі пошуку оптимального керування. Слід зауважити, що при $H \to \frac{1}{2}$ функція пряму до 0 при $s \neq t$ і отримане рівняння в (3.9) співпадає із класичним результатом для СДР Іто при $H = \frac{1}{2}$, тобто функція $p(t)$ задовольняє диференціальне рівняння

$$\frac{dp(t)}{dt} + 2p(t)a(t) + Q(t) - R(t)b^2(t)p^2(t) = 0.$$

Отже можна побачити неперевну залежність керування від параметра H фрактального броунівського руху $B^H(t)$. Загальніші теореми, які ґрунтуються на розгляді марковських процесів, розглянуті в роботі [203]. Приклади використання семімартингалів та відповідні задачі пошуку оптимального керування розглянуті в роботах [204, 205, 206]. Синтез оптимального керування на основі інтегралів за семімартингалами розглянуті в роботі [205], в якій значна увага приділена саме

алгоритмам синтезу оптимального керування на основі наближення неперервних семімартингалів дискретними версіями.

Другим основним напрямком узагальнення СДР (3.1) є узагальнення коефіцієнтів та структури розв'язку за рахунок додавання випадкових зовнішніх чинників по відношенню до вінерівського процесу $w(t), t \geq 0$. Дані випадковості можна трактувати як випадкові коефіцієнти рівняння або наявність зовнішніх збурень у вигляді переключень. Вплив обох цих чинників на структуру оптимального керування розглянуто в роботах Ясинського В.К., Лукашіва Т.О [207, 208] та роботах Лукашіва Т.О., Малика І.В. [209]. Авторами даних робіт вдалося розширити використання другого методу Ляпунова на СДР із зовнішніми збуреннями та випадковими коефіцієнтами та відповідними узагальненнями для рівняння Гамільтона – Якобі – Беллмана. Так, наприклад у роботі [210] вдалося встановити достатні умови асимптотичної стійкості для СДР із випадковими коефіцінтами та зовнішніми збуреннями у вигляді марковського процесу перемикань. Розглянемо один результат, присвячений стійкості стохастичних диференціальних рівнянь випадкової структури наступного вигляду

$$dx(t) = \alpha(t)a(t, \xi(t), x_t)dt + b(t, \xi(t), x_t)dw(t), t \in R_+ \backslash K \qquad (3.10)$$

з переключеннями

$$x(t_k) = x(t_k -) + g\big(t_k, \xi(t_k -), \eta_k, x_{t_k -}\big), t_k \in K \qquad (3.11)$$

та початковою умовою

$$x_0 = \phi \in C([-h, 0]), \qquad (3.12)$$

де $\xi(t), t \geq 0$ – марковський процес переключень в просторі Y, що задається щільністю переходу $p: R_+ \times Y \times Y \to R_+$; $\eta_k, k \geq 0$ – дискретний ланцюг Маркова задана на вимірному просторі H, що задається матрицею перехідних ймовірностей P; a, b, g – вимірні функції своїх аргументів агрумінтів; $\alpha(t)$ – неперервний випадковий процес, причому випадкові процеси $\xi(t), w(t), \alpha(t), \eta_k$ є незалежними в сукупності; $x_t = \{x(t-\theta), \theta \in [0, h]\}$ - . Тоді має місце наступне твердження.

Теорема 3.3. Припустимо, що виконуються наступні умови:

- Коефіцієнти задачі (3.10) – (3.12) задовольняють умови існування та єдиності;
- Моменти переключення t_k здовольняють співвідношення

$$|t_k - t_{k-1}| > \Delta > 0;$$

- Функції a, b, g задовольняють узагальнене глобальну умову Ліпшиця

$$|a(t,y,\phi_1) - a(t,y,\phi_2)|^2 + |b(t,y,\phi_1) - b(t,y,\phi_2)|^2 +$$
$$|g(t,y,h,\phi_1) - g(t,y,h,\phi_2)| \le L \sup_{\theta\in[0,h]} |\phi_1(\theta) - \phi_2(\theta)|^2$$

для довільного $y \in Y, h \in H, t \ge 0, \phi_i \in C([-h,0])$;

- Існує послідовність функціоналів Ляпунова – Красовського v_k та u_k, для яких виконується співвідношення

$$Lv_k \le -u_k,$$

де L – інфінітезимальний оператор (генератор) випадкового процесу $x(t)$, заданого задачею (3.10) – (3.12).

Тоді розв'язок задачі (3.10) – (3.12) асимптотично стійкий, тобто виконуються співвідношення

- Для довільних $\varepsilon_1, \varepsilon_2 > 0$ існує $\delta > 0$ таке що за умови $\sup_{\theta\in[0,h]} |\phi(\theta)| < \delta$ виконується умова

$$P\left(\sup_{t\ge 0}|x(t)| > \varepsilon_1\right) < \varepsilon_2;$$
$$\lim_{T\to\infty} P\left(\sup_{t\ge T}|x(t)| > \varepsilon_1\right) = 0.$$

Таким чином, використовуючи другий метод Ляпунова і його модифікацію Ляпунова – Красовського, вдалося знайти достатні умови асимптотичної стійкості, що є спорідненою задачею із задачею пошуку оптимального керування за умови збіжності інтегралу по процесу

$$J(u) = \int_0^\infty E|x^u(t)|^2 dt.$$

3.1. Постановка задачі.

Перейдемо тепер безпосередньо до основного математичного об'єкту даного розділу, а саме стохастичного диференціального рівняння із марковськими збуреннями та випадковими переключеннями. На ймовірнісному базисі $(\Omega, \mathfrak{F}, F, \mathbf{P})$ розглянемо стохастичну динамічну систему випадкової структури, задану стохастичним диференціальним рівнянням Ito з пуассоновими збуреннями наступного вигляду:

$$dx(t) = a(t-, \xi(t-), x(t-), u(t-))dt + \\ + b(t-, \xi(t-), x(t-), u(t-))dw(t) + \\ + \int_{\mathbb{R}^m} \big(c(t-, \xi(t-), x(t-), u(t-), z)\big)\tilde{\nu}(dz, dt), t \in \mathbb{R}_+ \backslash K, \quad (3.13)$$

з марковськими перемиканнями

$$\Delta x(t)|_{t=t_k} = g\big(t_k-, \xi(t_k-), \eta_k, x(t_k-)\big), t_k \in K = \{t_n \Uparrow\} \quad (3.14)$$

для $\lim\limits_{n\to+\infty} t_n = +\infty$ та початковими умовами

$$x(0) = x_0 \in \mathbb{R}^m, \xi(0) = y \in \mathbf{Y}, \eta_0 = h \in \mathbf{H}. \quad (3.15)$$

Тут $\xi(t), t \geq 0$ є однорідним неперервним марковським процесом зі скінченною кількістю станів $\mathbf{Y} := \{y_1, \dots, y_N\}$ і генератором Q; $\{\eta_k, k \geq 0\}$ ланцюг Маркова зі значеннями у просторі $\mathbf{H}$, і матрицею перехідних ймовірностей $\mathbb{P}_H$; $x: [0, +\infty) \times \Omega \to \mathbb{R}^m$; $w(t)$ є m-вимірним стандартним вінеровим процесом; $\tilde{\nu}(dz, dt) = \nu(dz, dt) - \mathbb{E}\nu(dz, dt)$ є центрованою мірою Пуассона; процеси w, ν, ξ і η незалежні в сукупності. Позначимо через

$$\mathfrak{F}_{t_k} = \sigma(\xi(s), w(s), \nu(s,*), \eta_e, s \leq t_k, t_e \leq t_k)$$

мінімальну σ-алгебра, відносно якої $\xi(t)$ вимірна для всіх $t \in [0, t_k]$ і η_n для $n \leq k$.

Процес $x(t), t \geq 0$ є $c\grave{a}dl\grave{a}g$; керування $u(t) := u(t, x(t)): [0, T] \times \mathbb{R}^m \to \mathbb{R}^m$ є m- вимірною функцією з класу допустимих керувань U.

Вимірні за сукупністю змінних відображення $a\colon \mathbb{R}_+ \times \mathbf{Y} \times \mathbb{R}^n \times \mathbb{R}^n \to \mathbb{R}^n$, $b\colon \mathbb{R}_+ \times \mathbf{Y} \times \mathbb{R}^n \times \mathbb{R}^n \to \mathbb{R}^n \times \mathbb{R}^n$, $c\colon \mathbb{R}_+ \times \mathbf{Y} \times \mathbb{R}^n \times \mathbb{R}^n \times \mathbb{R}^n \to \mathbb{R}^n$ і функція $g\colon \mathbb{R}_+ \times \mathbf{Y} \times \mathbf{H} \times \mathbb{R}^n \to \mathbb{R}^n$ задовольняють умову Ліпшиця

$$|a(t,y,x_1,u) - a(t,y,x_2,u)| + |b(t,y,x_1,u) - b(t,y,x_2,u)| +$$
$$+ \int_{\mathbb{R}^m} |c(t,y,x_1,u,z) - c(t,y,x_2,u,z)|\Pi(dz) +$$
$$+|g(t,y,h,x_1) - g(t,y,h,x_2)| \le L|x_1 - x_2|,$$

де $\Pi(dz)$ визначається зі співвідношення $\mathbb{E}\nu(dz,dt) = \Pi(dz)dt$, $L > 0$, $x_1, x_2 \in \mathbb{R}^m$ для $\forall t \ge 0, y \in \mathbf{Y}, h \in \mathbf{H}$, і стан

$$|a(t,y,0,u)| + |b(t,y,0,u)| + \int_{\mathbb{R}^m} |c(t,y,0,u,z)|\Pi(dz) +$$
$$+|g(t,y,h,0)| \le C < \infty,$$

Визначені вище умови щодо відображень a, b, c та g гарантують існування та єдиність сильного розв'язку рівнянь (3.13)–(3.15) з точністю до стохастичної еквівалентності.

Введемо до розгляду сумісну перехідну ймовірність випадкового процесу $(\xi(t_{k+1}), \eta_{k+1}, x(t_{k+1}))$

$$\mathbf{P}_k((y,h,x), \Gamma \times G \times \mathbf{C}) :=$$
$$:= P(\xi(t_{k+1}), \eta_{k+1}, x(t_{k+1}) \in \Gamma \times G \times \mathbf{C} | (\xi(t_k), \eta_k, x(t_k)) = (y,h,x))$$

Слід відзначити, що з використанням марковської властивості розв'язку $x(t)$, дана перехідна ймовірність задає перехідну ймовірність ланцюга Маркова $(\xi(t_k), \eta_k, x(t_k))$, що визначає розв'язок рівнянь (3.13)–(3.15).

3.2. Синтез оптимального керування

Задача оптимальної керування для СДР Іто (3.13) з перемиканнями (3.14) полягає в побудові такого керування $u(t, x(t))$, для якого незбурений рух задачі (3.13) – (3.15) $x(t) \equiv 0$ буде в цілому стійкий за ймовірністю. Слід зазначити, що для існуваннння тривіального розв'язку задачі Коші (3.13) – (3.15) слід вимагати додаткові умови на коефіцінти рівнянь (3.13) та (3.14), а саме

$$a(t,y,0,u) = b(t,y,0,u) = c(t,y,0,u,z) = g(t,y,h,0) = 0$$

для довільних $t \in R_+, y \in Y, u \in R^m, h \in H$.

Надалі по аналогії із теоремою 3.1 будемо припускати, що керування u визначатиметься за принципом повного зворотного зв'язку. Крім того, умова неперервності $u(t)$ по t буде справедливою для довільних значень із наступної області

$$t \geq 0, x \in \mathbb{R}^n, y \in \mathbf{Y}, h \in \mathbf{H}. \quad (3.16)$$

для кожного фіксованого $\xi(t) = y \in \mathbf{Y}$ та $\eta_k = h \in \mathbf{H}$.

Також будемо припускати, що відома структура системи на момент часу $t \geq 0$, яка залежить від ланцюга Маркова η_k ($k \geq 0$ відповідає моменту часу $t_k \in K$).

Очевидним є той факт, що існує нескінченна кількість керувань, множину яких будемо позначати через U. Єдине керування слід вибирати з умови якості процесу, що виражається як умови мінімізації функціонала якості

$$J^u(y, h, x_0) :=$$

$$:= \sum_{k=0}^{\infty} \int_{t_k}^{\infty} E\{W(t, x(t), u(t)) | \xi(0) = y, \eta_0 = h, x(0) = x_0\} dt, \quad (3.17)$$

де $W(t, x, u) \geq 0$ є невід'ємною функцією, визначеною в області $t \geq 0, x \in \mathbb{R}^n$, $u \in \mathbb{R}^m$.

Алгоритм обчислення функціонала (3.17) для заданого керування $u(t, x)$ виглядає наступним чином.

- Знайти траєкторії $x(t)$ із СДР (3.13) при $u(t) \equiv u(t, y, h, x(t))$, наприклад, методом Ейлера-Маруями;
- Підставити $x(t), \xi(t), u(t) = u(t, x(t))$ у функціонал (3.17);
- Обчислити значення функції (3.17) методом статистичного моделювання (Монте-Карло);

Проблема вибору керування $u(t)$, що визначає оцінку J^u і якість процесу $x(t)$ як сильного розв'язку СДР (3.13), пов'язана з особливостями задачі, і можна виділити наступні чотири умови:

1. Умови мінімізації функціонала (3.17) повинні забезпечувати досить швидке затухання сильного розв'язку $x(t)$ СДР (3.13) у середньому з високою ймовірністю;

2. Значення інтегралу у функціоналі якості (3.17) повинен мати високу точність, що буде впливати на оцінку оптимального керування $u(t)$;

3. Функціонал $W(t,x,u)$ має бути таким, щоб можна було побудувати розв'язок задачі стабілізації.

Зауваження 1. Для лінійного СДР (3.13) у багатьох випадках квадратична форма за змінними x,u має вигляд

$$W(t,x,u) = x^T M(t)x + u^T D(t)u,$$

де $M(t)$ – симетрична невід'ємно визначена матриця розміру $n \times n$ і $D(t)$ є додатно визначеною матрицею розміру $m \times m$ для всіх $t \geq 0$.

Зауваження 2. Значення J^u у випадку квадратичної форми в середньому досить добре оцінює якість перехідного процесу. Наявність доданка $u^T Du$ та умови мінімуму одночасно обмежують величину керуючої дії $u \in \mathbb{R}^m$ та росту розв'язку за умови наявності доданку $x^T M(t)x$.

Зауваження 3. Якщо умова стрибка фазової траєкторії є лінійною, то розв'язок задачі стабілізації належить до класу лінійних на фазовому просторі $x \in \mathbb{R}^n$ для керувань $u(t,x)$. Такі задачі називаються задачами лінійно-квадратичної стабілізації.

Означення 1. Керування $u^0(t)$, яке задовольняє умову

$$J^{u_0}(y,h,x_0) = \min J^u(y,h,x_0),$$

де мінімум слід шукати для всіх допустимих керувань, що залежать від змінних t і x при фіксованих $\xi(0) = y \in \mathbf{Y}$ і $\eta_0 = h \in \mathbf{H}$. Керування u_0 будемо називати оптимальним у сенсі оптимальної стабілізації сильного розв'язку $x \in \mathbb{R}^n$ системи (3.13)-(3.15) та оптимізації функціоналу якості $J^u(y,h,x_0)$.

Теорема 3.4. Нехай для системи (3.13)-(3.15) існує скалярна функція $v^0(t_k,y,h,x)$ і m-вимірна функціЯ $u^0(t,y,h,x) \in \mathbb{R}^m$, що визначені в області (3.16), для яких виконуються умови:

1. Послідовність функції $v_k^0(y,h,x) \equiv v^0(t_k,y,h,x)$ є функціоналом Ляпунова;
2. Послідовність m-вимірних керувань

$$u_k^0(y,h,x) \equiv u^0(t_k,y,h,x) \in \mathbb{R}^r;$$

вимірні за усіма аргументами для $0 \le t_k < t_{k+1}, k \ge 0$;

3. Послідовність функцій з (3.17) при $x \in \mathbb{R}^n$ є додатно визначеною, тобто для $\forall t \in [t_k, t_{k+1}), k \ge 0$,

$$W(t,x,u_k^0(y,h,x)) > 0;$$

4. Послідовність нескінченно малих операторів $(lv_k^0)|_{u_k^0}$, обчислених для $u_k^0 \equiv u^0(y,h,x)$, задовольняє наступну умову для

$$(lv_k^0)|_{u_k^0} = -W(t,x,u_k^0), \forall t \in [t_k, t_{k+1});$$

5. Значення $(lv_k^0) + W(t,x,u)$ досягає мінімуму при $u = u^0, k \ge 0$, тобто

$$(lv_k^0)|_{u_k^0} + W(t,x,u_k^0) = \min_{u \in \mathbb{R}^r}\{(lv_k^0)|_u + W(t,x,u)\} = 0. \qquad (3.17)$$

6. Ряд

$$\sum_{k=0}^{\infty} \int_{t_k}^{\infty} \mathbf{E}\{W(t,x(t),u(t))|x(t_{k-1})\}dt < \infty \qquad (3.18)$$

є збіжним.

Тоді існу оптимальне керування $u_k^0 \equiv u^0(t_k,y,h,x), k \ge 0$, яке стабілізує розв'язок задачі Коші (3.13)-(3.15). Крім того, для даного оптимального керування буде мати місце наступна рівність

$$v^0(y,h,x_0) \equiv \sum_{k=0}^{\infty} \int_{t_k}^{\infty} \mathbf{E}\{W(t,x(t),u(t))|x(t_{k-1})\}dt =$$

$$= \min_{u \in \mathbb{R}^r} \sum_{k=0}^{\infty} \int_{t_k}^{\infty} \mathbf{E}\{W(t,x(t),u(t))|x_(t_k)\}dt \equiv I_{u^0}(y,h,x_0). \qquad (3.19)$$

Доведення теореми 3.4.

I. Стійкість за ймовірністю в цілому динамічної системи випадкової структури (3.13)-(3.15) для $u \equiv u^0(t_k,x), k \ge 0$ із означення стійкості в цілому та теореми 1

[210], оскільки функціонали $v^0(y,h,x)$ для будь-яких $t \in [t_k, t_{k+1}), k \geq 0$ задовольняють умови цієї теореми.

II. Рівність (3.19), очевидно, є також наслідком Теореми 1 [210].

III. Розглянемо доведення властивостей оптимального керування від супротивного. Припустимо, що стабілізація сильного розв'язку динамічної системи випадкової структури (3.13)-(3.15) відбувається з керуванням $u^0(t_k, x)$, $t_k \leq t < t_{k+1}, k \geq 0$.

Проте існу керування $u^*(t_k, x) \neq u^0(t_k, x)$, яке при підстановці в (3.13), (3.14) реалізує розв'язок $x^*(t)$ з початковими умовами (3.15), такий що справедлива нерівність

$$I_{u^*}(y,h,x_0) \leq I_{u^0}(y,h,x_0). \tag{3.20}$$

Виконання умов 1-6 Теореми 3.4 призведе до наступної нерівності:

$$(lv_k^0)|_{u^*} \geq -W\big(t,x,u^*(t,y,h,x)\big). \tag{3.21}$$

Усереднюючи (3.21) за випадковими величинами $\{x^*(t), \xi(t), \eta_k\}$ на інтервалах $[t_k, t_{k+1}), k \geq 0$ та інтегруючи по t від 0 до T, отримуємо наступні нерівності

$$\begin{gathered}
\mathbf{E}\left\{v^0\left(t_1,\xi(t_1),\eta_{k_1},x^*(t_1)\right)\middle|\, y_1,\eta_{k_1},x^*(t_1)\right\} - v^0(y,h,x_0) \geq \\
\geq -\int_0^{t_1} \mathbf{E}\big\{W\big(t,x^*(t),u^*(t)\big)|x_0\big\}dt, \\
\mathbf{E}\{v^0(t_2,\xi(t_2),\eta_{k_2},x^*(t_2))|y_1,\eta_{k_1},x^*(t_1)\} - \\
-\{v^0(t_1,\xi(t_1),\eta_{k_1},x^*(t_1))|y,h,x_0\} \geq \\
\geq -\int_{t_1}^{t_2} \mathbf{E}\big\{W\big(t,x^*(t),u^*(t)\big)|x^*(t_1)\big\}dt,
\end{gathered} \tag{3.22}$$

...

$$\begin{gathered}
\mathbf{E}\left\{v^0\left(t_n,\xi(t_n),\eta_{k_n},x^*(t_n)\right)|y_{n-1},\eta_{k_{n-1}},x^*(t_{n-1})\right\} - \\
-\{v^0(t_{n-1},\xi(t_{n-1}),\eta_{k_{n-1}},x^*(t_{n-1}))|y_{n-2},\eta_{k_{n-2}},x^*(t_{n-2})\} \geq \\
\geq -\int_{t_{n-1}}^{t_n} \mathbf{E}\big\{W\big(t,x^*(t),u^*(t)\big)|\, x^*(t_{n-1})\big\}.
\end{gathered} \tag{3.23}$$

Враховуючи мартингальну властивість функцій Ляпунова $v^0(t,\xi(t),h,x^*(t))$,

отримуємо n рівностей із ймовірністю одиниця:

$$\mathbf{E}\{v^0(t_k, \xi(t_k), \eta_k, x^*(t_k))|y_{k-1}, \eta_{k-1}, x^*(t_{k-1})\} =$$

$$= v^0(t_{k-1}, \xi(t_{k-1}), \eta_{k-1}, x^*(t_{k-1})), k = \overline{1,n}.$$

Підставляючи розв'язок $x^*(t)$ у нерівності (3.22)-(3.23), отримуємо нерівність

$$\mathbf{E}\{v^0(t_n, \xi(t_n), \eta_{k_n}, x^*(t_n))|t_{n-1}, \xi(t_{n-1}), \eta_{k_{n-1}}, x^*(t_{n-1})\} - v^0(y, h, x_0) \geq$$

$$\geq -\sum_{k=0}^{n} \int_{t_k}^{t_{k+1}} \mathbf{E}\{W(t, x^*(t), u^*(t))|x^*(t_{k-1})\}dt \geq$$

$$\geq -\sum_{k=0}^{\infty} \int_{t_k}^{\infty} \mathbf{E}\{W(t, x^*(t), u^*(t))|x^*(t_{k-1})\}dt. \qquad (3.24)$$

Відповідно до припущення (3.20) випливає, що при $t_n \to \infty$ інтеграли в правій частині (3.24) збігаються, і з урахуванням збіжності ряду (3.19) (умова 6) отримуємо нерівність:

$$v^0(y, h, x_0) = I_{u^0}(y, h, x_0) \leq$$

$$\leq \sum_{k=0}^{\infty} \int_{t_k}^{\infty} \mathbf{E}\left\{\frac{W(t, x^*(t), u^*(t))}{x^*(t_{k-1})}\right\} dt = I_{u^*}(y, h, x_0). \qquad (3.25)$$

Дійсно, зі збіжності ряду в останній нерівності випливає, що підінтегральні вирази прямують до нуля при $t \to \infty$. Таким чином,

$$\lim_{n\to\infty} \mathbf{E}\{v^0(t_n, y_n, \eta_{k_n}, x^*(t_n)\} = 0.$$

Зауважимо, що має сенс розглядати природні випадки, коли з умови

$$\mathbf{E}\{W\} \underset{t\to\infty}{\to} 0$$

випливає умова $\mathbf{E}\{v^0\} \underset{t\to\infty}{\to} 0$.

Отже, нерівність (3.25) суперечить нерівності (3.20). Ця суперечність доводить твердження про оптимальність керування $u^0(t_k, x), k \geq 0$.

У випадках, коли марковський процес зі скінченною кількістю станів $\xi(t_k)$ допускає наступний розклад умовної ймовірності:

$$\mathbf{P}\{\omega: \xi(t + \Delta t) = y_j|\xi(t) = y_i, y_i \neq y_j\} =$$

$$= q_{ij}(t)\Delta t + o(\Delta t), i, j = \overline{1,N}, \qquad (3.26)$$

отримуємо рівняння, які мають задовольняти оптимальні функції Ляпунова $v_k^0(y,h,x)$, та оптимальне керування $u_k^0(t,x), \forall t \in [t_k, t_{k+1})$.

Зазначимо, що слабкий інфінітезимальний оператор (генератор) процесу (y,h,x) має вигляд

$$(lv_k)(y,h,x) = \frac{\partial v_k(y,h,x)}{\partial t} + (\nabla v_k(y,h,x), a(t,y,x,u)) +$$
$$+\frac{1}{2} Sp(b^T(t,y,x,u) \cdot \nabla^2 v_k(y,h,x) \cdot b(t,y,x,u)) +$$
$$+ \int_{\mathbb{R}^m} [v_k(y,h,x + c(t,y,x,u,z)) - v_k(y,h,x) - (\nabla v_k(y,h,x))^T \times$$
$$\times c(t,y,x,u,z)]\Pi(dz) +$$
$$+ \sum_{j\neq i}^{N} [\int_{\mathbb{R}^m} v_j(t,x) p_{ij}(t,z|x)dz - v_i(t,x)]q_{ij}, \qquad (3.27)$$

де $(\cdot,\cdot)$ є скалярним добутком, $\nabla v_k = \left(\frac{\partial v_k}{\partial x_1}, \dots, \frac{\partial v_k}{\partial x_m}\right)^T$, $\nabla^2 v_k = \left[\frac{\partial^2 v_k}{\partial x_i \partial x_j}\right]_{i,j=1}^{m}$, $k \geq 0$, "T" позначає транспонування, Sp – слід матриці, $p_{ij}(t,z|x)$ – умовна щільність ймовірності

$$P(x(\tau) \in [z, z+dz] | x(\tau - 0) = x) = p_{ij}(\tau, z|x)dz + o(dz)$$

за умови $\xi(\tau - 0) = y_i, \xi(\tau) = y_j$.

Беручи до уваги формулу (3.27), перше рівняння для v^0 можна отримати, замінивши ліву частину (3.18) на вираз для інфінітезимального оператора $(lv_k^0)\big|_{u^*}$.

Далі шукане рівняння в точках(t_k, y_j, η_k, x) має форму

$$\frac{\partial v_k^0}{\partial t} + \left(\left(\frac{\partial v_k^0}{\partial x}\right)^T \cdot a(t,y,x,u)\right) + \frac{1}{2} Sp\left(\left(b^T(t,y_i,x) \cdot \frac{\partial^2 v_k^0}{\partial x^2} \cdot b(t,y_i,x)\right)\right) +$$
$$+ \int_{\mathbb{R}^m} [v_k^0(\cdot,\cdot,x + c(t,y,x,u,z)) - v_k^0 - (\frac{\partial v_k^0}{\partial x})^T \cdot c(t,y,x,u,z)]\Pi(dz) +$$
$$+ \sum_{j\neq i}^{l} \left[\int_{-\infty}^{+\infty} v_j^0(y_j,h,x_j) p_{ij}(t,z|x)dz - v_i^0(y_i,h,x)\right] q_{ij}(t)dt +$$
$$+ W(t,x,u) = 0. \qquad (3.28)$$

Друге рівняння для оптимального керування $u_k^0(t,y,h,x)$ можна отримати з

(3.28) диференціюванням по змінній u, оскільки $u = u^0$ забезпечує мінімум лівої частини (3.28)

$$\left[\left(\frac{\partial v^0}{\partial x}\right)^T \cdot \left(\frac{\partial a}{\partial u}\right) + \left(\frac{\partial W}{\partial u}\right)^T\right]\Big|_{u=u_k^0} = 0, \qquad (3.29)$$

де $\frac{\partial a}{\partial u}$ – $n \times m$-матриця Якобі, складена з елементів

$$\left\{\frac{\partial a_n}{\partial u_s}, n = \overline{1,m}, s = \overline{1,r}\right\}; \left(\frac{\partial W}{\partial u}\right) \equiv \left(\frac{\partial W}{\partial u_1}, \dots, \frac{\partial W}{\partial u_r}\right), k \geq 0.$$

Таким чином, задача оптимальної стабілізації, згідно Теореми 3.4, полягає в розв'язуванні складної нелінійної системи рівнянь (3.26) з частинними похідними для визначення невідомих функцій Ляпунова $v_{ik}^0 \equiv v_k^0(y,h,x), i = \overline{1,l}, k \geq 0$.

Слід зазначити, що ця система отримана шляхом 'усунення' керування $u_k^0 = u^0(t,y,h,x)$ з рівнянь (3.28) і (3.29).

Розв'язати таку систему досить складно, тому далі будемо розглядати лінійні стохастичні системи, для яких можна побудувати зручні схеми знаходження оптимального керування. Як було зазначено в Теоремах 3.2 та 3.3 пошук оптимального керування у випадку лінійних систем та квадратичних функціоналів якості зводиться до розв'язання деякого допоміжного диференціального або інтегро-диференціального рівняння.

3.3. Стабілізація лінійних систем

Розглянемо випадковий процес $x(t)$, що визначенається лінійним СДР Ito з марковськими перемиканнями та пуассонівськими збуреннями

$$dx(t) = [A(t-,\xi(t-))x(t-) + B(t-,\xi(t-))u(t-)]dt + \sigma(t-,\xi(t-))x(t-)dw(t) + \\ + \int_{\mathbb{R}^m} \mathrm{C}(t-,\xi(t-),u(t-),z)x(t-)\tilde{\nu}(dz,dt), t \in \mathbb{R}_+ \backslash K, \qquad (3.30)$$

де з марковські перемикання задаються співвідношенням

$$\Delta x(t)|_{t=t_k} = g\big(t_k-, \xi(t_k-), \eta_k, x(t_k-)\big), t_k \in K = \{t_n \Uparrow\} \quad (3.31)$$

для $\lim\limits_{n\to+\infty} t_n = +\infty$, причому розв'язок задовольняє початкову умову

$$x(0) = x_0 \in \mathbb{R}^n, \xi(0) = y \in \mathbf{Y}, \eta_0 = h \in \mathbf{H}. \quad (3.32)$$

Тут A, B, σ і C — кусково-неперервні інтегровні матриці-функції відповідної розмірності.

Припустимо, що стрибки фазового вектора $x \in \mathbb{R}^n$ в момент $t = t^*$ зміни структури системи внаслідок переходу $\xi(t^*-) = y_i$ у $\xi(t^*) = y_j \neq y_i$ є лінійними та задаються наступним співвідношенням

$$x(t^*) = K_{ij}x(t^*-) + \sum_{s=1}^{N} \xi_s Q_s x(t^*-), \quad (3.33)$$

де $\xi_s := \xi_s(\omega)$ – незалежні випадкові величини, для яких $\mathbf{E}\xi_s = 0, \mathbf{E}\xi_s^2 = 1$, K_{ij} і Q_s – задані $(m \times m)$-матриці.

Зазначимо, що рівність (3.33) може замінити загальні умови стрибка:

- у випадку невипадкових стрибків $Q_s = 0$, тобто

$$x(t^*) = K_{ij}x(t^*-);$$

- у випадку неперервної зміна фазового вектора $Q_s = 0$, $K_{ij} = A_{ij} = I$ (одинична $m \times m$ матриця).

Якість перехідного процесу оцінюватимемо квадратичним функціоналом

$$J^u(y, h, x_0) := $$

$$:= \sum_{k=0}^{\infty} \int_{t_k}^{\infty} \mathbf{E}\{x^T(t)M(t)x(t) + u^T(t)D(t)u(t)|y, h, x_0\}dt, \quad (3.34)$$

де $M(t) \geq 0, D(t) > 0$ є симетричними додатно визначеними матрицями розмірності $(n \times n)$ та $(m \times m)$ відповідно.

Відповідно до теореми 3.4, необхідно знайти оптимальні функції Ляпунова $v_k^0(y, h, x)$ і керування $u_k^0(t, x)$ для $\forall t \in [t_k, t_{k+1}), t_k \in K, k = 0,1,2,\ldots$

Оптимальні функції Ляпунова аналогічно до класичних результатах наведених в теоремах 3.2 та 3.3 будемо шукати в наступному вигляді

$$v_k^0(y, h, x) = x^T G(t, y, h)x,$$

де $G(t,y,h)$ додатно визначена симетрична матриця розміру $n \times n$.

Тут і далі, неперервний ланцюг Маркова $\xi(t)$ має скінченне число станів $\mathbf{Y} = \{y_1, y_2, \ldots, y_l\}$, а $\eta_k, k \geq 0$ описує ланцюг Маркова зі значеннями h_k в метричному просторі $\mathbf{H}$ і з ймовірністю переходу на k-му етапі $\mathbf{P}_k(h,G)$. Введемо до розгляду наступні позначення:

$$A_i(t) := A(t,y_i),$$

$$B_i(t) := B(t,y_i),$$

$$\sigma_i(t) := \sigma(t,y_i),$$

$$C_i(t,z) := C(t,y_i,z),$$

$$G_{ik}(t) := G(t,y_i,h_k),$$

$$v_{ik} := v(y_i,h_k,x).$$

Підставимо функціонал (3.35) у рівняння (3.28) і (3.30) для того, щоб оптимізувати функцію Ляпунова $v_k^0(y,h,x)$ і знайти оптимальне керування $u_k^0(t,x)$ для $\forall t \in [t_k, t_{k+1})$. Враховуючи вигляд слабкого інфінітезимального оператора (3.27), отримаємо:

$$x^T(t)\frac{dG_{ik}(t)}{dt}x(t) + 2[A_i(t)x(t) + B_i(t)u(t)]G_{ik}(t)x(t) +$$

$$+Sp\left(x^T(t)\sigma_i^T(t)G_{ik}(t)\sigma_i(t)x(t)\right) +$$

$$\int_{\mathbb{R}^m} x^T(t)C_i^T(t,z)G_{ik}(t)C_i(t,z)x(t)\Pi(dz) +$$

$$+x^T(t)\sum_{j\neq i}^{N}\left[K_{ij}^T G_{ik}(t)K_{ij} + \sum_{s=0}^{l} Q_s^T G_{ik}(t)Q_s - G_{ik}(t)\right]q_{ij}x(t) +$$

$$+x^T(t)M_{ik}(t)x(t) + u^T(t)D_{ik}(t)u(t) = 0, \qquad (3.36)$$

$$2x^T(t)G_{ik}(t)B_i(t) + 2u^T(t)D_{ik}(t) = 0. \quad (3.37)$$

Зауважимо, що частинна похідна по u оператора (lv) дорівнює нулю, що підтверджує гіпотезу про побудову оптимального керування, що не залежить від перемикання (3.31) та (3.30).

З (3.37) знаходимо оптимальне керування для $\xi(t) = y_i$, при перемиканні (3.31) $\eta_k = h_k, k \geq 0$,

$$u_{ik}^0(t,x) = -D_{ik}^{-1}(t)B_i^T(t)G_{ik}(t)x(t). \quad (3.38)$$

Враховуючи матричну рівність

$$2x^T(t)G_{ik}(t)A_i(t)x = x^T(t)(G_{ik}(t)A_i(t) + A_i^T(t)G_{ik}(t))x(t),$$

та виключивши u_{ik}^0 (3.36) і, прирівнявши до нуля отриману матрицю квадратичної форми, можна отримати систему матричних диференціальних рівнянь типу Ріккаті для знаходження матриць $G_{ik}(t)$, де $i = 1,2,\dots,l,\ k \geq 0$, що відповідає інтервалу $[t_k, t_{k+1})$:

$$\frac{dG_{ik}(t)}{dt} + G_{ik}(t)A_i(t) - B_i(t)D_{ik}^{-1}(t)B_i^T(t)G_{ik}(t)) +$$
$$+Sp(\sigma_i^T(t)G_{ik}(t)\sigma_i(t)) + \int_{\mathbb{R}^m} C_i^T(t,z)G_{ik}(t)C_i(t,z)\Pi(dz) +$$
$$+\sum_{j\neq i}^{N}\left[K_{ij}^T G_{ik}(t)K_{ij} + \sum_{s=0}^{l} Q_s^T G_{ik}(t)Q_s - G_{ik}(t)\right]q_{ij} +$$
$$+M_{ik}(t) = 0, \quad (3.39)$$

$$\lim_{t\to\infty} G_{ik}(t) = 0, i = \overline{1,N}, k \geq 0. \quad (3.40)$$

Таким чином, ми отримуємо твердження, яке є наслідком Теореми 3.4 для лінійних систем із квадратичним функціоналом якості за припущення, що керування $u(t)$ лінійно залежить від процесу $x(t)$.

Теорема 3.4. Нехай система матричних рівнянь (3.39) та (3.40) має додатно визначені розв'язки порядку $(m \times m)$

$$G_{1k}(t) > 0, G_{2k}(t) > 0, \dots, G_{lk}(t) > 0.$$

Тоді керування (3.38) дає розв'язки задачі оптимальної стабілізації системи (3.30)-(3.32) з умовою стрибка (3.33) та критерієм оптимальності (3.34).

Таким чином, використовуючи метод Ляпунова – Красовського для аналізу асимптотичної стійкості стохастичних диференціальних рівнянь Ito із перемиканнями (3.13) – (3.15), вдалося розробити алгоритм для синтезу оптимального керування. Слід зауважити, що для лінійних систем рівняння (3.36) та (3.37) відповідають аналогічному рівнянню системи (3.9) для визначеного оптимального керування диференціального рівняння із фрактальним броунівським рухом.

3.4. Модельний приклад

Розглянемо модельні приклади, в яких будемо припускати автономність систем, тобто незалежність коефіцінтів СДР (3.13) або (3.30) від часу t. Проте будемо припускати, що наявна опосередкована залежність від часу через неперервний ланцюг Маркова з неперервним часом $\xi(t)$, тобто випадковий процес $x(t)$ задовольняє СДР Ito

$$dx(t) =$$

$$[A(\xi(t))x + B(\xi(t))u]dt + \sigma(\xi(t))xdw(t) + C(\xi(t))xdN(t), t \in \mathbb{R}_+ \backslash K,$$

з марківським перемиканням (3.31) та початковими умовами (3.32). Тут $x \in \mathbb{R}^n$, $u \in \mathbb{R}^m$, $A(y)$, $B(y)$, $\sigma(y)$, $C(y)$ — відомі матричні функції, визначені на множині $\mathbf{Y} = \{y_1, y_2, \ldots, y_k\}$ можливих значень ланцюга Маркова ξ. $N(t), t \geq 0$ – пуассонівський процес з інтенсивністю λ.

У разі стрибків фазового вектора (3.33) і квадратичного функціоналу якості (3.34) система (3.39) і (3.40) для знаходження невідомих матриць $G_{ik}, i = \overline{1,N}, k \geq 0$ набуде вигляду

$$G_{ik}A_i + A_i^T G_{ik} - B_i D_{ik}^{-1} B_i^T G_{ik} + \sigma_i^T G_{ik}\sigma_i +$$

$$+\lambda C_i^T G_{ik} C_i +$$

$$+\sum_{j\neq i}^{N}\left[K_{ij}^{T}G_{ik}K_{ij}+\sum_{s=0}^{l}Q_{s}^{T}G_{ik}Q_{s}-G_{ik}\right]q_{ij}+$$

$$+M_{ik}=0, i=\overline{1,N}, k\geq 0. \quad (3.41)$$

Зауваження 4. Будь-яку диференціальну систему, записану в нормальній формі (наприклад, систему (3.30), де явно вказана залежність x від t), можна звести до автономної системи, збільшивши кількість невідомих функцій (координати) на одиницю.

Метод малого параметра для розв'язання задачі оптимальної стабілізації.

Алгоритмічне вирішення задачі оптимальної стабілізації лінійної автономної системи випадкової структури ((3.35), (3.31) та (3.32)) досягається введенням малого параметра. Є два способи введення малого параметра:

Випадок I. Імовірністі переходу $y_i \to y_j$ ланцюгів Маркова ξ малі, тобто інтенсивності переходу q_{ij} через малий параметр ($\varepsilon > 0$) можна подати у вигляді

$$q_{ij}=\varepsilon r_{ij}. \quad (3.42)$$

Випадок II. Малі стрибки фазового вектора $x(t) \in R^n$, тобто матриці K_{ij} і Q_s (3.33), слід подати у вигляді

$$K_{ij}=I+\varepsilon K_{ij}; Q_s=\varepsilon Q_s. \quad (3.43)$$

У цих випадках будемо шукати оптимальну функцію Ляпунова $v_k^0(y,x,h), k \geq 0$, у вигляді збіжного степеневого ряду

$$v_k^0(y,h,x)=x^T\sum_{r=0}^{\infty}\varepsilon^r G^{(r)}(y,h)x. \quad (3.44)$$

Згідно з (3.38) оптимальне керування u^0 слід шукати у вигляді збіжного ряду

$$u_k^0(y,h,x)=-[D^{-1}(y)B^T(y)\sum_{r=0}^{\infty}\varepsilon^r G^{(r)}(y,h)]x.$$

Випадок I. Підставимо ряд (3.44) і (3.45) з урахуванням (3.42) у (3.36):

$$G_{ik}A_i+(A_i)^T G_{ik}-B_i D_{ik}^{-1}B_i^T G_{ik}+\sigma_i^T G_{ik}\sigma_i+$$

$$+\lambda C_i^T G_{ik}C_i+$$

$$+\sum_{j\neq i}^{l}K_{ij}^T G_{ik}K_{ij}+\sum_{s=1}^{N}Q_s^T(G_{ik}Q_s-G_{ik})\varepsilon r_{ij}+C_{ik}=0; i=\overline{1,l}, k\geq 0.$$

Прирівнявши коефіцієнти при однакових степенях $\varepsilon > 0$, отримаємо:

$$A_i^T G_i^{(0)} + G_{ik}^{(0)} A_i - B_i D_{ik}^{-1} B_{ik}^T G_{ik}^{(0)} +$$

$$+\sigma_{ik}^T G_{ik}^{(0)} \sigma_{ik} + \lambda C_i^T G_{ik}^{(0)} C_i + M_{ik} = 0, i = \overline{1,l}, k \geq 0, \qquad (3.46)$$

$$\tilde{A}_{ik}^T G_{ik}^{(r)} + G_{ik}^{(r)} \tilde{A}_{ik} + \sigma_i^T G_{ik}^{(r)} \sigma_i + \lambda C_i^T G_{ik}^{(0)} C_i =$$

$$= -\sum_{j \neq i}^{l} (K_{ik}^T G_{ik}^{(r-1)} K_{ij} + \sum_{s=1}^{N} Q_s^T G_{ik}^{(r-1)} Q_s - G_{ik}^{(r-1)}) r_{ij} +$$

$$+ \sum_{q=1}^{r-1} B_i D_{ik}^{-1} B_i^T G_{ik}^{(r-q)}, \qquad (3.47)$$

$$r > 1; \tilde{A}_{ik} \equiv A_i - B_i D_{ik}^{-1} B_i^T G_{ik}^{(0)}, i = \overline{1,l}, k \geq 0.$$

Зауважимо, що система (3.46) складається з незалежних матричних рівнянь, які при фіксованих $i = 1,2,\ldots,l$ дають розв'язок задачі оптимальної стабілізації системи

$$dx(t) = \big(A_i x(t) + B_i u(t)\big)dt + \sigma_i x(t) dw(t) + C_i x(t) dN(t), \qquad (3.48)$$

з критерієм якості

$$J^u(y,h,x_0) = \sum_{k=0}^{\infty} \int_{t_k}^{\infty} E\{x^T(t) M_{ik} x(t) + u^T(t) D_{ik} u(t) | x_0\} dt,$$

$$i = \overline{1,l}, k \geq 0, M_{ik} > 0, D_{ik} > 0.$$

Необхідною та достатньою умовою розв’язку системи (3.47) є існування лінійного допустимого керування для системи (3.48), що забезпечує експоненційну стійкість у середньому квадратичному незбуреного руху цієї системи.

Припустимо, що система матричних квадратних рівнянь (3.46) має єдиний додатно визначений розв’язок $G_{ik}^{(0)} > 0, i = \overline{1,l}, k \geq 0$.

Рівняння (3.47) для знаходження $G_{ik}^{(r)} > 0, r \geq 1, k \geq 0$ є лінійним, тому воно має єдиний розв’язок для фіксованого $i = \overline{1,l}, k \geq 0, r \geq 1$ та будь-яких матриць, що знаходяться у правій частині (3.47).

Дійсно, система

$$dx(t) = \tilde{A}_{ik} x(t) dt + \sigma_i x(t) dw(t) + C_i x(t) dN(t)$$

отримується із системи (3.48) з оптимальним керуванням

$$u_k^0 = -D_{ik}^{-1} B_{ik}^T G_{ik}^{(0)} x(t),$$

що забезпечує експоненційну стійкість у середньому квадратичному. Тоді існу єдиний розв'язок системи (3.47). Зауважимо, що в лінійному випадку для автономних систем асимптотична стійкість еквівалентна експоненційній стійкості. Розглянемо теорему, що випливає з результатів цієї роботи.

Теорема 3.6. Якщо сильний розв'язок $x(t)$ системи (3.48) експоненціально стійкий у середньому квадратичному, то існують функції Ляпунова $v_k(y,h,x), k \geq 0$, які задовольняють умови:

$$c_1\|x\|^2 \leq v_k(y,h,x) \leq c_1\|x\|;$$

$$\frac{dE[v_k]}{dt} \leq -c_3\|x\|^2.$$

Таким чином, система матричних рівнянь (3.46) і (3.47) дозволяє послідовно знаходити коефіцієнти $G_{ik}^{(r)} > 0$ відповідного ряду (3.44), починаючи з додатного розв'язку $G_{ik}^{(0)} > 0, i = \overline{1,l}, k \geq 0$.

Наступним кроком є доведення збіжності ряду (3.44). Без втрати загальності ми спрощуємо позначення, фіксуючи $k \geq 0$. Нехай $L_r := \max\limits_{\substack{i=\overline{1,l},\\ k\geq 0}} \left\| G_i^{(r)} \right\|$. Тоді з (3.47) випливає, що існує константа $c > 0$, та для будь-якого $r > 0$ справедлива наступна оцінка

$$L_r \leq c\left[\sum_{q=1}^{r-1} L_q L_{r-q} + L_{r-1}\right]. \qquad (3.49)$$

Далі скористаємося методом мажорантних рядів. Розглянемо квадратне рівняння

$$\rho^2 + (a+\varepsilon)\rho + b = 0, \qquad (3.50)$$

де коефіцієнти a і b обрані так, що розклад в степеневий ряд одного з коренів цього розв'язку було мажорантним рядом (3.44).

Отримамо

$$\rho_{1,2} = -\frac{a+\varepsilon}{2} \pm \sqrt{\frac{(a+\varepsilon)^2}{4} - b} = \sum_{r=0}^{\infty} \varepsilon^r \rho_r. \quad (3.51)$$

Підставимо (3.51) в (3.50) і прирівняємо коефіцієнти при рівних степенях ε. Тоді ми отримаємо вираз для ρ_r через $\rho_0, \ldots, \rho_{r-1}$:

$$\rho_r = -\frac{1}{2\rho_0 + a}\left[\sum_{q=1}^{r-1} \rho_q \rho_{r-q} + \rho_{r-1}\right], \quad (3.52)$$

де ρ_0 слід знайти з рівняння

$$\rho_0^2 + a\rho_0 + b = 0.$$

Порівнюючи (3.49) і (3.52), знаходимо, що ряд (3.51) буде основним рядом для визначення коефіцінтів (3.44), якщо врахувати

$$c = -\frac{1}{2\rho_0 + a} > 0; \rho_0 = L_0 > 0.$$

Таким чином, значення коефіцієнтів a і b у рівнянні (3.50) є

$$a = -\left[\frac{1}{c} + 2L_0\right] < 0;$$
$$b = \frac{L_0}{c} + L_0^2 > 0.$$

Використовуючи відомі a і b (3.51), знаходимо, що мажорантний ряд для (3.44) буде розкладом одного з коренів (3.50). Даний корінь визначає зі співвідношення

$$\rho_0 = L_0 = -\frac{a}{2} - \sqrt{\frac{a^2}{4} - b}.$$

Збіжність ряду (3.44) при $v_k^0(y, h, x)$ випливає з очевидної нерівності

$$\left\|\sum_{r=0}^{\infty} \varepsilon^r G^{(r)}(y, h)\right\| \leq \sum_{r=0}^{\infty} L_r \varepsilon^r.$$

Таким чином, ми довели твердження, яке дозволяє будувати послідовне наближення функції G:

Теорема 5:

- Для $\forall i = \overline{1, l}, k \geq 0$ система (3.48) має лінійне допустиме керування;
- Інтенсивності переходу q_{ij} однорідного ланцюга Маркова ξ задовольняють умову (3.42).

Тоді:

- Існує єдиний розв'язок задачі оптимальної стабілізації системи (3.35), (3.31) та (3.32) з умовою стрибка (3.33) фазового вектора $x \in R^m$;
- Оптимальне керування $u_k^0(y,h,x)$ та відповідна функція Ляпунова $v_k^0(y,h,x)$, що визначаються збіжними рядами (3.44) і (3.45), знаходяться з відповідних систем (3.46) і (3.47).

Випадок II. Підставимо ряд $G_{ik} = \sum_{r=0}^{\infty} \varepsilon^r G_{ik}^{(r)}$ у (3.36) і прирівняємо коефіцієнти при тих самих степенях ε. Тоді з урахуванням (3.43) отримаємо наступні рівняння:

$$G_{ik}^{(0)} A_i + A_i^T G_{ik}^{(0)} + \sigma_i^T G_{ik}^{(0)} \sigma_i - B_i D_{ik}^{-1} B_i^T G_{ik}^{(0)} + \\ + \lambda C_i^T G_{ik} C_i + \sum_{j \neq i}^{l} \left(G_{jk}^{(0)} - G_{ik}^{(0)} \right) q_{ij} + M_{ik} = 0, k \geq 0, \tag{3.53}$$

$$G_{ik}^{(r)} \tilde{A}_{ik} + \tilde{A}_{ik}^T G_{ik}^{(r)} + \sigma_i^T G_{ik}^{(r)} \sigma_i + \lambda C_i^T G_{ik} C_i + + \\ \sum_{j \neq i}^{l} \left(G_{jk}^{(r)} - G_{ik}^{(r)} \right) q_{ij} = \Phi_{ik}^{(r)}, \tag{3.54}$$

де $i = \overline{1,l} k \geq 0$, $\tilde{A}_{ik} = A_i - B_i D_{ik}^{-1} B_i^T G_{ik}^{(0)}$,

$$\Phi_{ik}^{(r)} = \sum_{q=1}^{r-1} B_i D_{ik}^{-1} B_i^T G_{ik}^{(r-q)} - \\ - \sum_{j \neq i}^{l} (K_{ij}^T G_{jk}^{(r-1)} + G_{jk}^{(r-1)} K_{ij} + K_{ij}^T G_{jk}^{(r-2)} K_{ij} + \sum_{s=1}^{N} Q_s^T G_{jk}^{(r-2)} Q_s) q_{ij}.$$

Виходячи з наведених вище рівнянь, справедливе твердження щодо вигляду матриць G:

Теорема 3.5.

Припустимо, що

- Система матричних рівнянь (3.53) має єдиний додатно визначений розв'язок $G_{ik}^{(0)} > 0, i = \overline{1,l}; k \geq 0$;
- Стрибки фазового вектора $x \in R^n$ задовольняють умову (3.43).

Тоді лінійно-квадратична задача оптимальної стабілізації (3.35), (3.31) та (3.32) мінімізації функціонала (3.34) має єдиний розв'язок, який задано у вигляді збіжних

рядів (3.44) та (3.45), а матриці $G_{ik}^{(r)}, i = \overline{1,l}; r \geq 1, k \geq 0$ є єдиним розв'язком лінійних матричних рівнянь (3.54).

Модельні приклади

Модельний приклад 1. Щоб проілюструвати наведені вище теоретичні результати, розглянемо приклад із такими параметрами:

- Неперервний ланцюг Маркова $\xi(t), t \geq 0$ задається генератором

$$Q = \begin{pmatrix} -7 & 7 \\ 3 & -3 \end{pmatrix};$$

- $t_k = k, k \geq 1;$
- Значення функції g у моменти часу t_k залежать лише від значення x:

$$g(t, \xi, \eta, x) = K_{ij}x = \alpha x, Q_s = 0;$$

де $\alpha \in [-1,1]$. Наприклад, нижче ми використовуємо $\alpha = 0.2;$

- Інтенсивність пуассонівського процесу становить $\lambda = 0.2;$
- Значення матриць $A(\xi)$ для $\xi(t) \in \{1,2\}$ є

$$A_1 = \begin{pmatrix} -2 & 1 & 1 \\ 3 & -3 & 0 \\ 0 & 6 & -2 \end{pmatrix}, A_2 = \begin{pmatrix} -4 & 8 & 0 \\ 0 & 1 & 2 \\ 3 & -2 & -1 \end{pmatrix}.$$

- Значення матриць $B(\xi)$ для $\xi(t) \in \{1,2\}$ є

$$B_1 = \begin{pmatrix} 4 & 2 \\ 0 & -2 \\ 1 & 1 \end{pmatrix}, B_2 = \begin{pmatrix} 0 & 1 \\ 4 & -3 \\ -1 & 1 \end{pmatrix};$$

- Значення матриць $\sigma(\xi)$ для $\xi(t) \in \{1,2\}$ є

$$\sigma_1 = \begin{pmatrix} 1 & 0 & 0 \\ -1 & 1 & 1 \\ -1 & -1 & 1 \end{pmatrix}, \sigma_2 = \begin{pmatrix} 1 & -1 & -1 \\ 0 & 1 & -1 \\ 0 & 0 & 1 \end{pmatrix}.$$

- Значення матриць $C(\xi)$ для $\xi(t) \in \{1,2\}$ є

$$C_1 = \begin{pmatrix} 0.3 & 0.2 & 0.1 \\ -0.2 & 0.4 & 0.8 \\ 0.1 & 0.1 & 0.2 \end{pmatrix}, C_2 = \begin{pmatrix} -0.5 & 0.1 & -0.2 \\ 0.4 & 0.5 & -1.1 \\ -0.7 & -0.6 & 0.3 \end{pmatrix};$$

• Параметри керування є

$$M_{ik} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 3 \end{pmatrix}, D_{ik} = \begin{pmatrix} 10 & 0 \\ 0 & 40 \end{pmatrix}.$$

Для простоти вважатимемо, що випадкові величини η_k є константами, а розв'язок $x(t)$ і оптимальне керування $u(t)$ залежить тільки від випадкового процесу $\xi(t)$.

Основна задача оптимального керування – це знаходження розв'язку рівняння Ріккаті (3.41). Існує кілька основних підходів до пошуку наближеного розв'язку цього рівняння. Однак у прикладі ми використовували метод оптимізації рою частинок, який дозволяє відносно швидко знайти розв'язок рівняння (3.41). Результатом знаходження розв'язку цього рівняння будуть матриці

$$G_1 = \begin{pmatrix} 0.2044 & 0.0943 & 0.0043 \\ 0.1258 & 0.3605 & 0.165 \\ 0.0139 & 0.1575 & 0.3146 \end{pmatrix}, G_2 = \begin{pmatrix} 0.1268 & 0.5538 & -0.0962 \\ 0.2533 & 2.8729 & 0.2214 \\ 0.1096 & 0.2075 & 2.0079 \end{pmatrix}.$$

Обидва розв'язки додатно визначені, тому за Теоремою 3.6 існує оптимальне керування, яке стабілізує систему (3.48) і визначається як

$$u(t)_{\xi(t)=i} = -D^{-1}B^T G_i x(t)$$

для $i \in \{1,2\}$. Дві реалізації розв'язку $x(t)$ та відповідного керування $u(t)$ показані на рисунку 3.1.

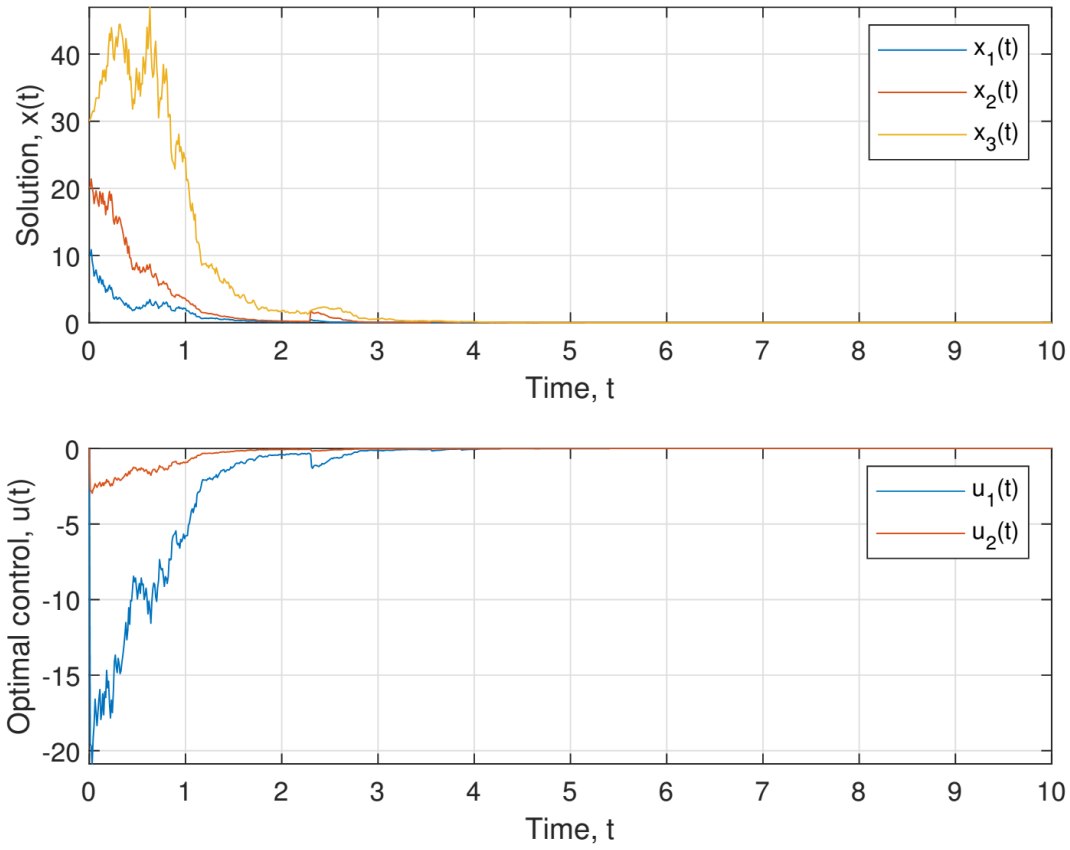


Рис. 3.1. Реалізації розв'язку та оптимального керування з початковими умовами $x = (10, 20, 30)$.

Як ми можемо бачити з наведених вище прикладів, результуюче оптимальне керування $u(t)$ стабілізує систему, а отже, мінімізує функціонал $J^u(y, h, x_0)$. Крім того, розглядаючи вигляд матриці D з (3.34), можна побачити, що $u_2(t)$ близьке до 0, оскільки

$$u^T(t)Du(t) = 100u_1^2(t) + 1600u_2^2(t).$$

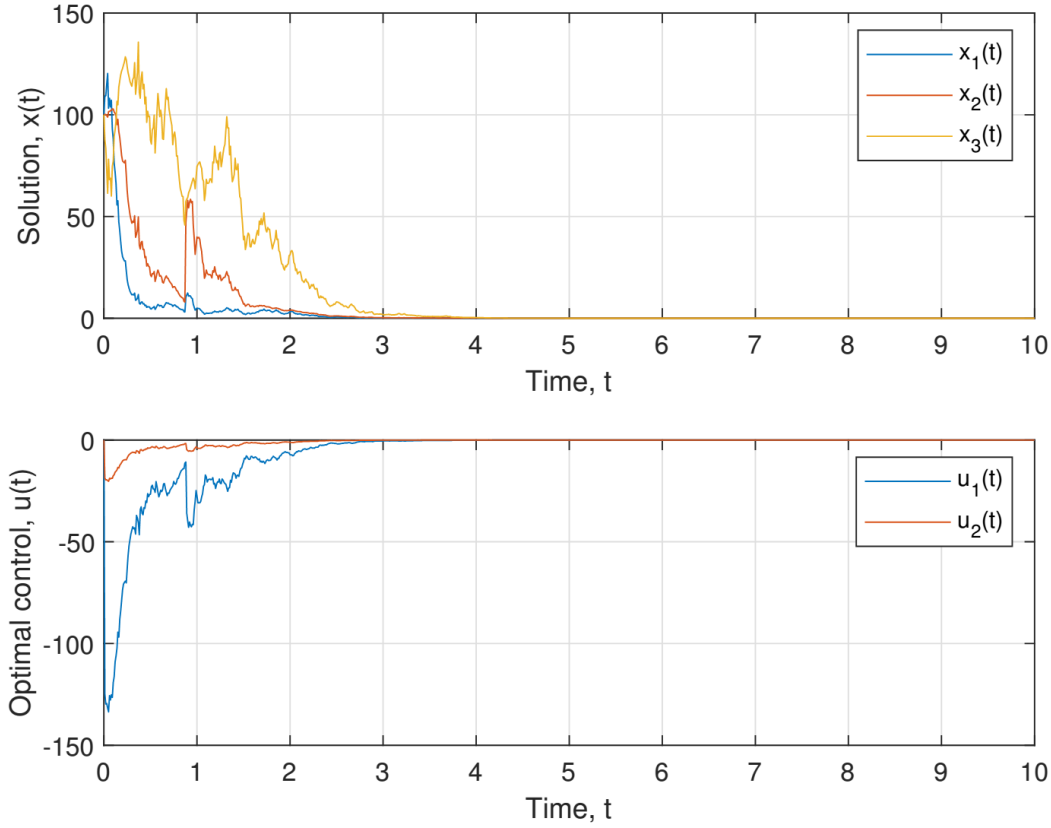


Рис. 3. 2. Реалізації розв'язку та оптимального керування з початковими умовами (100, 100, 100).

Таким чином, знайдене оптимальне керування узгоджується із заданим функціоналом якості $J^u(y, h, x_0)$.

Аналіз розв'язку показав наявність оптимального керування для довільного $\alpha \in [-1,1]$. Випадок $|\alpha| < 1$ відповідає ситуації стиснення, оскільки в цьому випадку розв'язок стискається на коефіцієнт α на кожному кроці в точці t_k. Випадок $|\alpha| = 1$ не є стискаючим, але існування оптимального керування можна знайти на основі Теореми 3.6. Випадок $|\alpha| > 1$ виходить за рамки даного дослідження, оскільки в цьому випадку розв'язок рівняння (3.50) або не існує, або не є додатно визначеним.

Модельний приклад 2. Розглянемо приклад використання розширеного CMA-ES алгоритму для оптимізації лінійно-квадратичної пробдеми пошуку оптимального керування. Для цього визначимо лінійну систему різницевих рівнянь наступним чином

$$x(t+1) = Ax(t) + Bu(t), t = 0, \dots, T-1$$

де

$$A = \begin{pmatrix} 1 & 2 & 4 \\ 0 & 4 & 4 \\ 1 & 3 & 5 \end{pmatrix}, B = \begin{pmatrix} -1 & 2 \\ 1 & 1 \\ -1 & 4 \end{pmatrix}.$$

Функціонал якості для даного рівняння буде мати вигляд

$$J^u = \sum_{t=1}^{T-1} \left(x^T(t) Q x(t) + u^T(t) R u(t) \right) + x^T(T) H x(T),$$

де

$$Q = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}, R = \begin{pmatrix} 0.2 & 0 \\ 0 & 0.2 \end{pmatrix}, H = \begin{pmatrix} 5 & 1 & 0 \\ 0 & 4 & 1 \\ 1 & 0 & 10 \end{pmatrix}.$$

Умови на матриці виконуються, оскільки матриці Q, R, H є додатньовизначеними. Крім того, вибиремо нульове початкове значення для різницевого рівняння $x(0) = (0,0,0)^T$. У цьому випадку оптимальне керування u^0 та відповідний процес x^0 будуть нульовими

$$u(t) = \begin{pmatrix} 0 \\ 0 \end{pmatrix}, x(t) = \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix}, t \geq 1.$$

При цьому функціонал якості для даного оптимального значення буде рівний 0, що і забезпечу глобальний мінімум невід'много функціоналу якості J^u.

Поряд із цим розглянемо моделювання оптимального керування на основі евристичних алгоритмів, описаних у Розділі 2 цієї роботи. Ще одним важливим параметром функціоналу якості є горизонт дослідження системи T. Для детальнішого дослідження евриистичних алгоритмів, розглянемо три значення даного параметру

$$T = 5, 20, 50.$$

При цьому оптимізаціна задача буде проводитися в просторі R^{Tm}, оскільки будемо шукати оптимальне керування

$$u = \left(u_0 = \begin{pmatrix} u_{0,1} \\ u_{0,2} \end{pmatrix}, u_1 = \begin{pmatrix} u_{1,1} \\ u_{1,2} \end{pmatrix}, \dots, u_{T-1} = \begin{pmatrix} u_{T-1,1} \\ u_{T-1,2} \end{pmatrix} \right).$$

Для аналізу задачі будемо розглядати допустимі керування, які знаходяться на просторі

$$U = \left\{ u_t = \begin{pmatrix} u_{t,1} \\ u_{t,2} \end{pmatrix}, \left| u_{t,i} \right| \leq 20, t = 0, \dots, T-1; i = 1,2 \right\}.$$

Для всіх евристичних алгоритмів розглянемо також наступні параметри:

- $N_{ch} = 50$ – кількість хромосом;
- $Iter_{MAX} = 1000$ – максимальна кількість ітерацій;
- $\varepsilon = 0.001$ – абсолютна точність наближення;
- $N_{const} = 20$ – кількість ітерацій для перевірки зміни мінімального значення. Два останні параметри використовуються для аналізу виходу з алгоритму за допомогою перевірки умови

$$\left| F_n - F_{n-N_{const}} \right| < \varepsilon,$$

де

$$F_n = \min_{i=1,\dots,N_{ch}} J^{u_i}(T).$$

У випадку якщо остання нерівність виконується, то відбувається вихід з алгоритму пошуку, оскільки зміна оптимально значення цільової функції є несуттєвою на протязі N_{const} ітерацій (епох) алгоритму.

Крім того для аналізу середньої кількості викликів функції будемо використовувати метод Монте – Карло з $Iter = 10^3$ ітерацій.

Як ми можемо бачити з таблиці 1, розширений CMA-ES алгоритм з розмірністю суміші $K = 5$ дозволяє отримувати найкращий результат з точки зору кількості викликів функціоналу якості (цільової функції) J^u. Для усіх випадків горизонту

дослідження $T = 5,10,50$ даний метод показав найменшу (в середньому) кількість викликів функції та найменше середнє квадратичне відхилення серед викликів у методі Монте Карло. Також слід зауважити, що середньоквадратичні відхилення у всіх трьох методах стабілізується, що випливає із аналізу випадків $T = 10$ та $T = 50$. Також при аналізі пошуку оптимального керування за допомогою п'яти алгоритмів (PSO, GA, ALO, CMA-ES, CMA-ES5) було встановлено, що алгоритми PSO, GA дозволяють знайти субоптимальні керування з меншими значеннями функціоналу якості J^u ніж відповідні субоптимальні керування для алгоритмів ALO, CMA-ES, CMA-ES5. Даний факт знову ж таки наштовхує на думку про важливість розробки самоадаптивних евристичних алгоритмів для аналізу складних систем.

Таблиця 3. 1.

Порівняльний аналіз п'яти евристичних алгоритмів

	PSO	GA	ALO	CMA-ES	CMA-ES5
$T = 5$	11097 (9636)	898060 (815952)	1583 (448)	2027 (490)	**934** **(295)**
$T = 10$	27475 (10064)	1456302 (956834)	3201 (521)	3325 (530)	**1902** **(431)**
$T = 50$	48983 (12921)	2817847 (1104562)	6542 (786)	5975 (831)	**4805** **(721)**

Модельний приклад 3. Розглянемо оптимізацію системи, визначеної параметрами в модельному прикладі 1. Проте тут зосередимо увагу на пошуку оптимального керування на основі евристичних алгоритмів. Для аналізу розглянемо випадок

$$T = 10.$$

Оскільки побудова лінійних керувань грунтується на формулі (3.38), то основною проблемою в даному прикладі буде оцінка матриць G_1 та G_2, точні значення яких визначені в модельному прикладі 1

$$G_1 = \begin{pmatrix} 0.2044 & 0.0943 & 0.0043 \\ 0.1258 & 0.3605 & 0.165 \\ 0.0139 & 0.1575 & 0.3146 \end{pmatrix},$$

$$G_2 = \begin{pmatrix} 0.1268 & 0.5538 & -0.0962 \\ 0.2533 & 2.8729 & 0.2214 \\ 0.1096 & 0.2075 & 2.0079 \end{pmatrix}. \quad (3.55)$$

Зауважимо, що на основі теореми 3.5, матриці G_1 та G_2 повинні бути додатновизначені. Тому для моделювання додатно визначених матриць будемо використовувати розподіл Вішарта (Wishart distribution) з параметрами (m, n, V) та зі щільністю

$$f(X) = \frac{|X|^{\frac{m-n-1}{2}} e^{-\frac{Sp(V^{-1}X)}{2}}}{2^{\frac{mn}{2}} |V|^{\frac{n}{2}} \Gamma_n\left(\frac{m}{2}\right)},$$

де $\Gamma_n(a)$ – багатовимірний гамма-розподіл, $|X|$ - визначник матриці X. Крім того, у якості параметру розподілу V будемо використовувати випадкові діагональні матриці із додатніми алементами на діагоналі, а точніше

$$V_{ii} \sim Unif(0,10).$$

Параметр m буде рівномірно вибиратися з множини $\{n, \dots, 2n\}$. Використовуючи дані параметри моделювання матриць, отримали результати близькі до результатів, обговорених в модельному прикладі 2. Знову ж таки, за допомогою методу Монте Карло вдалося встановити що CMA-ES5 алгоритм показав кращі результати, проте точність оцінки матриць (3.55) є дещо вищою для методів PSO та GA.

Висновки до розділу III

У даному розділі отримано достатні умови існування розв'язку задачі оптимальної стабілізації динамічних систем зі стрибками. Розглянуто випадок лінійної системи з квадратичним функціоналом якості, причому в даному випадку знайдено замкнену фому оптимального керування. Показано, що синтез оптимального керування, яке стабілізує систему за ймовірністю, спрощує проблему вирішення рівнянь Ріккаті. Крім того, для лінійної автономної системи обґрунтовано метод розв'язання задачі оптимальної стабілізації з використанням малого параметра. Отримані розв'язання можуть бути використані для опису фондового ринку в економіці, біологічних системах, включаючи моделі відповіді на лікування раку, та інших складних динамічних системах.

Практичним застосування результатів даного розділу є модельні приклади, в яких проведено порівняння класичних евристичних алгоритмів та розширеного CMA-ES алгоритму. Як показали результати моделювання на основі методу Монте Карло, новий алгоритм володіє більшою швидкодію в порівнянні із класичними алгоритмами, проте точність субоптимальних значень є нижчою ніж для PSO та GA. Третій модельний приклад показав аналогічні результати для оцінки матриць G_1 та G_2, які відіграють ключову роль в задачах оптимізації лінійних систем із квадратичним функціоналом якості.

Підсумовуючи все вище сказане, до основних теоретичних та практичних здобутків розділу ІІІ можна віднести наступне:

- В розділі проведено аналіз ключових джерел, які спрямовані на розв'язання оптимізаціних задаач;
- Знайдено достатні умови стабілізації СДР із випадковими збуреннями та марковськими перемиканнями в термінах функцій Ляпунова;
- Знайдено явний вигляд оптимального керування для лінійних автономних систем із квадратичним функціоналом якості;
- Проведено аналіз класичних евристичних алгоритмів та зроблено відповідні висновки щодо їх переваг та недоліків, причому аналіз проведено як для

моделювання безпосередньо керування, так і для моделювання додатно визначених матриць G_i, $i = 1,2$ (3.55).

.

ОСНОВНІ РЕЗУЛЬТАТИ І ВИСНОВКИ

Основна увага дисертаційної роботи присвячена побудові самоадаптивних алгоритмів на основі сумішей розподілів та алгоритму еволюційної стратегії з адаптацією коваріаційної матриці (CMA-ES) для оцінки параметрів складних систем. Даний алгоритм дозволяє знаходити швидше розв'язок оптимізаційної задачі $f(x) \to extr$, де функція f: $R^d \to R$, причому припускається, що розглянута задача оптимізації є безумовною, тобто $x \in R^d$. Також у дослідженні було поставлено задачу розробки самоадаптивного алгоритму оптимізації розміру суміші для розширеного CMA-ES алгоритму. Розв'язання цієї задачі було здійснено на основі побудови самоадаптивного алгоритму на прикладі класичного CMA-ES.

Основними результатами роботи є наступні:

1. Запропоновано розширення CMA-ES алгоритму за припущення багатопіковості розподілу хромосом в генетичному алгоритмі. Розроблено алгоритм для оцінки гіперпараметрів складних систем на основі розширеного CMA-ES алгоритму.

2. Запропоновано самоадаптивний алгоритм оптимізації розміру суміші для розширеного CMA-ES алгоритму, при якому мінімізується кількість звернень до цільової функції і таким чином підтверджує ефективність запропонованого алгоритму.

3. На прикладі використання методу Монте-Карло показано, що запропонований алгоритм дає кращі результати для вибраної оптимізаційної задачі у порівнянні з рядом класичних еволюційних алгоритмів.

4. Використано розширений CMA-ES алгоритм для конструювання оптимального керування в задачах синтезу оптимального керування для системи диференціальних.

5. Розглянуто задачі оптимізації стохастичних динамічних систем випадкої структури з марковськими переключеннями.

6. Встановлено ряд переваг та неділіків використанні розширеного CMA-ES алгоритму, основною перевагою якого являться суттєве зменшення кількості звернень до цільової функції, а основним недоліком якого являться деяке зменшення точності субоптимальних значень для цільвої функції.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Michalewicz Z. Genetic Algorithms + Data Structures = Evolution Programs. Berlin, Heidelberg : Springer Berlin Heidelberg, 1996. URL: https://doi.org/10.1007/978-3-662-03315-9.
2. Handbook of Evolutionary Computation / ed. by T. Baeck, D. B. Fogel, Z. Michalewicz. CRC Press, 1997. URL: https://doi.org/10.1201/9780367802486.
3. Bäck T., Schwefel H.-P. An Overview of Evolutionary Algorithms for Parameter Optimization. Evolutionary Computation. 1993. Vol. 1. Iss. 1. P. 1–23. URL: https://doi.org/10.1162/evco.1993.1.1.1.
4. Schwefel H.-P., Rudolph G. Contemporary evolution strategies. Advances in Artificial Life. Berlin, Heidelberg, 1995. P. 891–907. URL: httP.s://doi.org/10.1007/3-540-59496-5_351.
5. Genetic Programming: An Introduction : On the Automatic Evolution of Computer Programs and Its Applications (The Morgan Kaufmann Series in Artificial Intelligence) / R. E. Keller et al. Morgan Kaufmann Publishers, 1997. P. 450.
6. Slowik A., Kwasnicka H. Evolutionary algorithms and their applications to engineering problems. Neural Computing and Applications. 2020. Vol. 32. Iss. 16. P. 12363–12379. URL: https://doi.org/10.1007/s00521-020-04832-8.
7. Abraham A., Corchado E., Corchado J. M. Hybrid learning machines. Neurocomputing. 2009. Vol. 72. Iss. 13-15. P. 2729–2730. URL: https://doi.org/10.1016/j.neucom.2009.02.017.
8. A novel two-stage hybrid swarm intelligence optimization algorithm and application / W. Deng et al. Soft Computing. 2012. Vol. 16. Iss. 10. P. 1707–1722. URL: https://doi.org/10.1007/s00500-012-0855-z.
9. Fazel Zarandi M. H., Gamasaee R. Type-2 fuzzy hybrid expert system for prediction of tardiness in scheduling of steel continuous casting process. Soft Computing. 2012. Vol. 16. Iss. 8. P. 1287–1302. URL: https://doi.org/10.1007/s00500-012-0812-x.

10. A hybrid particle swarm optimization algorithm for high-dimensional problems / D. Jia et al. Computers & Industrial Engineering. 2011. Vol. 61. Iss. 4. P. 1117–1122. URL: https://doi.org/10.1016/j.cie.2011.06.024.
11. Khashei M., Bijari M., Hejazi S. R. Combining seasonal ARIMA models with computational intelligence techniques for time series forecasting. Soft Computing. 2012. Vol. 16. Iss. 6. P. 1091–1105. URL: https://doi.org/10.1007/s00500-012-0805-9.
12. A Hybrid GA-PSO Approach Based on Similarity for Various Types of Economic Dispatch Problems / U. Guvenc et al. Elektronika ir Elektrotechnika. 1970. Vol. 108. Iss. 2. P. 109–114. URL: https://doi.org/10.5755/j01.eee.108.2.155.
13. Evaluating cardiac health through semantic soft computing techniques / G. Acampora et all. Soft Computing. 2011. Vol. 16. Iss. 7. P. 1165–1181. URL:https://doi.org/10.1007/s00500-011-0792-2.
14. Electric energy forecasting in crude oil processing using Support Vector Machines and Particle Swarm Optimization / M. Petrujkic et al. 2008 9th Symposium on Neural Network Applications in Electrical Engineering (NEUREL 2008), Belgrade. 2008. URL: https://doi.org/10.1109/neurel.2008.4685568.
15. Sadeghi B. H. M. A BP-neural network predictor model for plastic injection molding process. Journal of Materials Processing Technology. 2000. Vol. 103. Iss. 3. P. 411–416. URL: https://doi.org/10.1016/s0924-0136(00)00498-2.
16. Global optimization of absorption chiller system by genetic algorithm and neural network / T. T. Chow et al. Energy and Buildings. 2002. Vol. 34. Iss. 1. P. 103–109. URL: https://doi.org/10.1016/s0378-7788(01)00085-8.
17. Cook D. F., Ragsdale C. T., Major R. L. Combining a neural network with a genetic algorithm for process parameter optimization. Engineering Applications of Artificial Intelligence. 2000. Vol. 13. Iss. 4. P. 391–396. URL: https://doi.org/10.1016/s0952-1976(00)00021-x.

18. Woll S. L. B., Cooper D. J. Pattern-based closed-loop quality control for the injection molding process. Polymer Engineering & Science. 1997. Vol. 37. Iss. 5. P. 801–812. URL: https://doi.org/10.1002/pen.11723 .
19. Chen C. R., Ramaswamy H. S. Modeling and optimization of variable retort temperature (VRT) thermal processing using coupled neural networks and genetic algorithms. Journal of Food Engineering. 2002. Vol. 53. Iss. 3. P. 209–220. URL: https://doi.org/10.1016/s0260-8774(01)00159-5.
20. Schaffer J. D., Whitley D., Eshelman L. J. Combinations of genetic algorithms and neural networks: a survey of the state of the art. [Proceedings] COGANN-92: International Workshop on Combinations of Genetic Algorithms and Neural Networks, м. Baltimore, MD, USA. URL: https://doi.org/10.1109/cogann.1992.273950.
21. Nagata Y., Chu K. H. Optimization of a fermentation medium using neural networks and genetic algorithms Biotechnology Letters. 2003. Vol. 25. Iss. 21. P. 1837–1842. URL: https://doi.org/10.1023/a:1026225526558.
22. Estivill-Castro V. The design of competitive algorithms via genetic algorithms. ICCI'93: 5th International Conference on Computing and Information, м. Sudbury, Ont., Canada. URL: https://doi.org/10.1109/icci.1993.315358.
23. Zhang G., Eddy Patuwo B., Y. Hu M. Forecasting with artificial neural networks: International Journal of Forecasting. 1998. Vol. 14. Iss. 1. P. 35–62. URL: https://doi.org/10.1016/s0169-2070(97)00044-7 .
24. Finnie G. R., Wittig G. E., Desharnais J.-M. A comparison of software effort estimation techniques: Using function points with neural networks, case-based reasoning and regression models. Journal of Systems and Software. 1997. Vol. 39. Iss. 3. P. 281–289. URL: https://doi.org/10.1016/s0164-1212(97)00055-1.
25. Boozarjomehry R. B., Svrcek W. Y. Automatic design of neural network structures. Computers & Chemical Engineering. 2001. Vol. 25. Iss. 7-8. P. 1075–1088. URL: https://doi.org/10.1016/s0098-1354(01)00680-9.

26. Fischer M. M., Leung Y. A genetic-algorithms based evolutionary computational neural network for modelling spatial interaction dataNeural network for modelling spatial interaction data. The Annals of Regional Science. 1998. Vol. 32, Iss. 3. P. 437–458. URL: https://doi.org/10.1007/s001680050082 .
27. Predicting the exchange traded fund DIA with a combination of genetic algorithms and neural networks / M. Versace et al. Expert Systems with Applications. 2004. Vol. 27. Iss. 3. P. 417–425. URL: https://doi.org/10.1016/j.eswa.2004.05.018.
28. Cheng J., Li Q. S. Reliability analysis of structures using artificial neural network based genetic algorithms. Computer Methods in Applied Mechanics and Engineering. 2008. Vol. 197. Iss. 45-48. P. 3742–3750. URL: https://doi.org/10.1016/j.cma.2008.02.026.
29. Structure optimization of neural network for dynamic system modeling using multi-objective genetic algorithm / S. M. R. Loghmanian et al. Neural Computing and Applications. 2011. Vol. 21. Iss. 6. P. 1281–1295. URL: https://doi.org/10.1007/s00521-011-0560-3.
30. Shapiro A. F. The merging of neural networks, fuzzy logic, and genetic algorithms. Insurance: Mathematics and Economics. 2002. Vol. 31. Iss. 1. P. 115–131. URL: https://doi.org/10.1016/s0167-6687(02)00124-5.
31. Evaluating the process of a genetic algorithm to improve the back-propagation network: A Monte Carlo study / C.-Y. Huang et al. Expert Systems with Applications. 2009. Vol. 36. Iss. 2. P. 1459–1465. URL: https://doi.org/10.1016/j.eswa.2007.11.055 .
32. Venkatesan D., Kannan K., Saravanan R. A genetic algorithm-based artificial neural network model for the optimization of machining processes. Neural Computing and Applications. 2008. Vol. 18. Iss. 2. P. 135–140. URL: https://doi.org/10.1007/s00521-007-0166-y.

33. Training feedforward networks using genetic algorithms/ Montana D.J., Davis L.D. IJCAI'89: Proceedings of the 11th international joint conference on Artificial intelligence. Conference Series. 1989. Vol. 1. P. 762 – 767.
34. Xiao H., Tian Y. Prediction of mine coal layer spontaneous combustion danger based on genetic algorithm and BP neural networks. Procedia Engineering. 2011. Vol. 26. P. 139–146. URL: https://doi.org/10.1016/j.proeng.2011.11.2151.
35. Optimal Sizing of Energy Storage System in Solar Energy Electric Vehicle Using Genetic Algorithm and Neural Network / S. Zhou et al. Advanced Intelligent Computing Theories and Applications. With Aspects of Artificial Intelligence. Berlin, Heidelberg, 2007. P. 720–729. URL: https://doi.org/10.1007/978-3-540-74205-0_76.
36. Sexton R. S., Dorsey R. E., Sikander N. A. Simultaneous optimization of neural network function and architecture algorithm. Decision Support Systems. 2004. Vol. 36. Iss. 3. P. 283–296. URL: https://doi.org/10.1016/s0167-9236(02)00147-1.
37. Alsultanny Y. A., Aqel M. M. Pattern recognition using multilayer neural-genetic algorithm. Neurocomputing. 2003. Vol. 51. P. 237–247. URL: https://doi.org/10.1016/s0925-2312(02)00619-7.
38. Guyer D., Yang X. Use of genetic artificial neural networks and spectral imaging for defect detection on cherries. Computers and Electronics in Agriculture. 2000. Vol. 29, Iss. 3. P. 179–194. URL: https://doi.org/10.1016/s0168-1699(00)00146-0.
39. Furtuna R., Curteanu S., Leon F. An elitist non-dominated sorting genetic algorithm enhanced with a neural network applied to the multi-objective optimization of a polysiloxane synthesis process. Engineering Applications of Artificial Intelligence. 2011. Vol. 24. Iss. 5. P. 772–785. URL: https://doi.org/10.1016/j.engappai.2011.02.004.
40. Samanta B. Gear fault detection using artificial neural networks and support vector machines with genetic algorithms. Mechanical Systems and Signal Processing. 2004. Vol. 18. Iss. 3. P. 625–644. URL: https://doi.org/10.1016/s0888-3270(03)00020-7.

41. Samanta B. Artificial neural networks and genetic algorithms for gear fault detection. Mechanical Systems and Signal Processing. 2004. Vol. 18. Iss. 5. P. 1273–1282. URL: https://doi.org/10.1016/j.ymssp.2003.11.003.
42. Jack L. B., Nandi A. K. Genetic algorithms for feature selection in machine condition monitoring with vibration signals. IEE Proceedings - Vision, Image, and Signal Processing. 2000. Vol. 147, Iss. 3. P. 205. URL: https://doi.org/10.1049/ip-vis:20000325 .
43. Failure diagnosis and nonlinear observer. Application to a hydraulic process / H. Hammouri et al. Journal of the Franklin Institute. 2002. Vol. 339. Iss. 4-5. P. 455–478. URL: https://doi.org/10.1016/s0016-0032(02)00027-3.
44. Xin Yao. Evolving artificial neural networks. Proceedings of the IEEE. 1999. Vol. 87. Iss. 9. P. 1423–1447. URL: https://doi.org/10.1109/5.784219.
45. Altun A. A. A combination of Genetic Algorithm, Particle Swarm Optimization and Neural Network for palmprint recognition. Neural Computing and Applications. 2012. Vol. 22. Iss. 1. P. 27–33. URL: https://doi.org/10.1007/s00521-011-0800-6.
46. Cooperative binary-real coded genetic algorithms for generating and adapting artificial neural networks / D. Barrios et al. Neural Computing & Applications. 2003. Vol. 12. Iss. 2. P. 49–60. URL: https://doi.org/10.1007/s00521-003-0364-1.
47. Delgado M., Pegalajar M. C. A multiobjective genetic algorithm for obtaining the optimal size of a recurrent neural network for grammatical inference. Pattern Recognition. 2005. Vol. 38. Iss. 9. P. 1444–1456. URL: https://doi.org/10.1016/j.patcog.2004.03.026.
48. Arifovic J., Gençay R. Using genetic algorithms to select architecture of a feedforward artificial neural network. Physica A: Statistical Mechanics and its Applications. 2001. Vol. 289. Iss. 3-4. P. 574–594. URL: https://doi.org/10.1016/s0378-4371(00)00479-9.
49. Karimi H., Yousefi F. Application of artificial neural network–genetic algorithm (ANN–GA) to correlation of density in nanofluids. Fluid Phase Equilibria. 2012. Vol. 336. P. 79–83. URL: https://doi.org/10.1016/j.fluid.2012.08.019.

50. Neural network model incorporating a genetic algorithm in estimating construction costs / G.-H. Kim et al. Building and Environment. 2004. Vol. 39. Iss. 11. P. 1333–1340. URL: https://doi.org/10.1016/j.buildenv.2004.03.009.
51. Kim H.-j., Shin K.-s., Park K. Time Delay Neural Networks and Genetic Algorithms for Detecting Temporal Patterns in Stock Markets. Lecture Notes in Computer Science. Berlin, Heidelberg, 2005. P. 1247–1255. URL: https://doi.org/10.1007/11539087_164.
52. Kim H.-j., Shin K.-s. A hybrid approach based on neural networks and genetic algorithms for detecting temporal patterns in stock markets. Applied Soft Computing. 2007. Vol. 7. Iss. 2. P. 569–576. URL: https://doi.org/10.1016/j.asoc.2006.03.004.
53. Ferentinos K. P. Biological engineering applications of feedforward neural networks designed and parameterized by genetic algorithms. Neural Networks. 2005. Vol. 18. Iss. 7. P. 934–950. URL: https://doi.org/10.1016/j.neunet.2005.03.010.
54. Liang Y.-H. Combining seasonal time series ARIMA method and neural networks with genetic algorithms for predicting the production value of the mechanical industry in Taiwan. Neural Computing and Applications. 2008. Vol. 18. Iss. 7. P. 833–841. URL: https://doi.org/10.1007/s00521-008-0216-0.
55. Malek H., Ebadzadeh M. M., Rahmati M. Three new fuzzy neural networks learning algorithms based on clustering, training error and genetic algorithm. Applied Intelligence. 2011. Vol. 37. Iss. 2. P. 280–289. URL: https://doi.org/10.1007/s10489-011-0327-7.
56. Melin P., Castillo O. Introduction to Pattern Recognition with Intelligent Systems. Hybrid Intelligent Systems for Pattern Recognition Using Soft Computing. Berlin, Heidelberg, 2005. P. 1–5. URL: https://doi.org/10.1007/978-3-540-32378-5_1.
57. Chen Y.-C., Su C.-T., Yang T. Rule extraction from support vector machines by genetic algorithms. Neural Computing and Applications. 2012. Vol. 23. Iss. 3-4. P. 729–739. URL: https://doi.org/10.1007/s00521-012-0985-3.

58. Zhang G. P. Neural networks for classification: a survey. IEEE Transactions on Systems, Man and Cybernetics, Part C (Applications and Reviews). 2000. Vol. 30. Iss. 4. P. 451–462. URL: https://doi.org/10.1109/5326.897072.
59. Oreski S., Oreski D., Oreski G. Hybrid system with genetic algorithm and artificial neural networks and its application to retail credit risk assessment. Expert Systems with Applications. 2012. Vol. 39. Iss. 16. P. 12605–12617. URL: https://doi.org/10.1016/j.eswa.2012.05.023.
60. Emission control in palm oil mills using artificial neural network and genetic algorithm / A. L. Ahmad et al. Computers & Chemical Engineering. 2004. Vol. 28. Iss. 12. P. 2709–2715. URL: https://doi.org/10.1016/j.compchemeng.2004.07.034.
61. Boehm O., Hardoon D. R., Manevitz L. M. Classifying cognitive states of brain activity via one-class neural networks with feature selection by genetic algorithms. International Journal of Machine Learning and Cybernetics. 2011. Vol. 2. Iss. 3. P. 125–134. URL: https://doi.org/10.1007/s13042-011-0030-3.
62. Tong D. L., Schierz A. C. Hybrid genetic algorithm-neural network: Feature extraction for unpreprocessed microarray data. Artificial Intelligence in Medicine. 2011. Vol. 53. Iss. 1. P. 47–56. URL: https://doi.org/10.1016/j.artmed.2011.06.008.
63. Classification of seismic signals at Villarrica volcano (Chile) using neural networks and genetic algorithms / G. Curilem et al. Journal of Volcanology and Geothermal Research. 2009. Vol. 180. Iss. 1. P. 1–8. URL: https://doi.org/10.1016/j.jvolgeores.2008.12.002.
64. Dieterle F., Kieser B., Gauglitz G. Genetic algorithms and neural networks for the quantitative analysis of ternary mixtures using surface plasmon resonance. Chemometrics and Intelligent Laboratory Systems. 2003. Vol. 65. Iss. 1. P. 67–81. URL: https://doi.org/10.1016/s0169-7439(02)00104-1.
65. Kim K. Application of a hybrid genetic algorithm and neural network approach in activity-based costing. Expert Systems with Applications. 2003. Vol. 24. Iss. 1. P. 73–77. URL: https://doi.org/10.1016/s0957-4174(02)00084-2.

66. Koc A. B., Heinemann P. H., Ziegler G. R. Optimization of Whole Milk Powder Processing Variables with Neural Networks and Genetic Algorithms. Food and Bioproducts Processing. 2007. Vol. 85. Iss. 4. P. 336–343. URL: https://doi.org/10.1205/fbp07074.
67. Potočnik P., Grabec I. Empirical modeling of antibiotic fermentation process using neural networks and genetic algorithms. Mathematics and Computers in Simulation. 1999. Vol. 49. Iss. 4-5. P. 363–379. URL: https://doi.org/10.1016/s0378-4754(99)00045-2.
68. Prakasham R. S., Sathish T., Brahmaiah P. Imperative role of neural networks coupled genetic algorithm on optimization of biohydrogen yield. International Journal of Hydrogen Energy. 2011. Vol. 36. Iss. 7. P. 4332–4339. URL: https://doi.org/10.1016/j.ijhydene.2011.01.031.
69. Takeda F., Nishikage T., Omatu S. Banknote recognition by means of optimized masks, neural networks and genetic algorithms. Engineering Applications of Artificial Intelligence. 1999. Vol. 12. Iss. 2. P. 175–184. URL: https://doi.org/10.1016/s0952-1976(98)00061-x.
70. Kim K.-j., Lee W. B. Stock market prediction using artificial neural networks with optimal feature transformation. Neural Computing and Applications. 2004. Vol. 13. Iss. 3. P. 255–260. URL: https://doi.org/10.1007/s00521-004-0428-x.
71. J Holland J. H. Adaptation in Natural and Artificial Systems. The MIT Press, 1992. URL: https://doi.org/10.7551/mitpress/1090.001.0001.
72. Sakawa M. Genetic Algorithms and Fuzzy Multiobjective Optimization. Boston, MA : Springer US, 2002. URL: https://doi.org/10.1007/978-1-4615-1519-7.
73. Coley D. A. An Introduction to Genetic Algorithms for Scientists and Engineers. World Scientific Publishing Company, 1997. P. 227.
74. E G. D. Genetic algorithms in search, optimization, and machine learning. Reading, Mass: Addison-Wesley Pub. Co., 1989. P 412.

75. The enhanced genetic algorithms for the optimization design/ Guo P., Wang X., Han Y. 3rd International Conference on Biomedical Engineering and Informatics (BMEI), м. Yantai, China, 2010. URL: https://doi.org/10.1109/bmei.2010.5639829.
76. Hamdan M. A heterogeneous framework for the global parallelization of genetic algorithms, Arab J. Inf. Tech. 2008. Vol. 5. Iss. 2. P.192 – 199.
77. Sivanandam S. N., Deepa S. N. Evolutionary Computation. Introduction to Genetic Algorithms. Berlin, Heidelberg. P. 1–13. URL: https://doi.org/10.1007/978-3-540-73190-0_1
78. Searson D. Non – linear PLS using genetic programming, PhD thesis submitted to school of chemical engineering and advance materials, University of Newcastle. 2005. P. 242.
79. Mitchell M. An introduction to genetic algorithms. Cambridge, Mass : MIT Press, 1996. P 205.
80. Goldberg D. E. Simple genetic algorithms and the minimal deceptive problem / In Davis L.D. (ed.): Genetic Algorithms and Simulated Annealing. Morgan Kaufmann. 1987.
81. Vose D.M., Liepins G.E. Punctuated equilibria in genetic search, Compl. Syst. 1991. Vol. 5. P. 31 – 44.
82. A simulated annealing−like convergence theory for the simple genetic algorithm/ Davis T.E., Principe J.C. Proceedings of the Fourth International Conference on Genetic Algorithms.1991.
83. Whitley D. An Executable Model of a Simple Genetic Algorithm. Foundations of Genetic Algorithms. 1993. P. 45–62. URL: https://doi.org/10.1016/b978-0-08-094832-4.50009-x.
84. Laboudi Z., Chikhi S. Comparison of genetic algorithms and quantum genetic algorithms, Int. Arab J. Inf. Tech. 2012. Vol. 9. Iss. 3. P. 243 – 249.

85. Conti E. Improving exploration in evolution strategies for deep reinforcement learning via a population of novelty-seeking agents. NIPS, 2018. P. 18. URL: https://doi.org/10.48550/arXiv.1712.06560.
86. Hansen, N. The CMA Evolution Strategy: A Comparing Review. CoLab Computational Laboratory. ETH Zurich. 2014. URL: https://doi.org/10.1007/3-540-32494-1_4.
87. On the adaptation of arbitrary normal mutation distributions in evolution strategies: The generating set adaptation / Hansen N. Proceedings of the Sixth International Conference on Genetic Algorithms. Pittsburgh. 1995. P. 57-64.
88. Adapting arbitrary normal mutation distributions in evolution strategies: the covariance matrix adaptation / Hansen N., Ostermeier A. IEEE International Conference on Evolutionary Computation. Japan. 1996. URL: https://doi.org/10.1109/icec.1996.542381.
89. Gagganapalli S. R. Implementation and Evaluation of CMA-ES Algorithm. Master's Thesis, North Dakota State University, Fargo, ND, USA. 2015. P. 67.
90. Hansen N., Müller S. D., Koumoutsakos P. Reducing the Time Complexity of the Derandomized Evolution Strategy with Covariance Matrix Adaptation (CMA-ES). Evolutionary Computation. 2003. Vol. 11. Iss. 1. P. 1–18. URL: https://doi.org/10.1162/106365603321828970.
91. Hansen N. The CMA Evolution Strategy: A Tutorial Nikolaus Hansen. 2011. P. 39. URL: https://doi.org/10.48550/arXiv.1604.00772.
92. Comparing results of 31 algorithms from the black-box optimization benchmarking BBOB-2009 / N. Hansen et al. the 12th annual conference comp. New York, USA. 2010. P. 1689–1696. URL: https://doi.org/10.1145/1830761.1830790.
93. Jin J., Yang C., Zhang Y. An Improved CMA-ES for Solving Large Scale Optimization Problem. Lecture Notes in Computer Science. Cham. 2020. P. 386–396. URL: https://doi.org/10.1007/978-3-030-53956-6_34.

94. Warm Starting CMA-ES for Hyperparameter Optimization / M. Nomura et al. Proceedings of the AAAI Conference on Artificial Intelligence. 2021. Vol. 35. Iss. 10. P. 9188–9196. URL: https://doi.org/10.1609/aaai.v35i10.17109.
95. Efficient multi-objective CMA-ES algorithm assisted by knowledge-extraction-based variable-fidelity surrogate model / Z. Li et al. Chinese Journal of Aeronautics. 2022. URL: https://doi.org/10.1016/j.cja.2022.09.020.
96. CMA-ES for Hyperparameter Optimization of Deep Neural Networks/ Loshchilov I., Hutter F. In International Conference on Learning Representations. Workshop Track. 2016. URL: https://doi.org/10.48550/arXiv.1604.07269
97. Dang V.-H., Vien N. A., Chung T. A covariance matrix adaptation evolution strategy in reproducing kernel Hilbert space. Genetic Programming and Evolvable Machines. 2019. Vol. 20. Iss. 4. P. 479–501. URL: https://doi.org/10.1007/s10710-019-09357-1.
98. Ros R., Hansen N. A Simple Modification in CMA-ES Achieving Linear Time and Space Complexity. Parallel Problem Solving from Nature – PPSN X. Berlin, Heidelberg. 2008. P. 296–305. URL: https://doi.org/10.1007/978-3-540-87700-4_30.
99. Loshchilov I., Schoenauer M., Sebag M. Self-adaptive surrogate-assisted covariance matrix adaptation evolution strategy. the fourteenth international conference. Philadelphia, Pennsylvania, USA. 2012. URL: https://doi.org/10.1145/2330163.2330210.
100. A competitive variable-fidelity surrogate-assisted CMA-ES algorithm using data mining techniques / Z. Li et al. Aerospace Science and Technology. 2021. Vol. 119. P. 107084. URL: https://doi.org/10.1016/j.ast.2021.107084.
101. PSA-CMA-ES/ Nishida K., Akimoto Y. GECCO '18: Genetic and Evolutionary Computation Conference, Japan. New York, NY, USA. 2018. URL: https://doi.org/10.1145/3205455.3205467.
102. Cooperative Coevolutionary CMA-ES with Landscape-Aware Grouping in Noisy Environments / Y. Wu et al. IEEE Transactions on Evolutionary Computation. 2022. P. 1. URL: https://doi.org/10.1109/tevc.2022.3180224.

103. Hybrid of PSO and CMA-ES for Global Optimization / P. Xu et al. IEEE Congress on Evolutionary Computation (CEC). Wellington, New Zealand. 2019. URL: https://doi.org/10.1109/cec.2019.8789912.

104. Efficient multi-objective CMA-ES algorithm assisted by knowledge-extraction-based variable-fidelity surrogate model / Z. Li et al. Chinese Journal of Aeronautics. 2022. URL: https://doi.org/10.1016/j.cja.2022.09.020.

105. Tamilselvi S. Introduction to Evolutionary Algorithms. 1994. Berlin, Heidelberg: Springer; P. 80-94. DOI:http://dx.doi.org/10.5772/intechopen.104198

106. Back T., Hammel U., Schwefel H. P. Evolutionary computation: comments on the history and current state. IEEE Transactions on Evolutionary Computation. 1997. Vol. 1. Iss. 1. P. 3–17. URL: https://doi.org/10.1109/4235.585888.

107. Bäck T., Rudolph G., Schwefel H.-P. Evolutionary programming and evolution strategies: Similarities and differences. In Proceedings of the Second Annual Conference on Evolutionary Programming, Citeseer. 1993.

108. Storn, Rainer and Price, Kenneth. Differential Evolution - A Simple and Efficient Adaptive Scheme for Global Optimization over Continuous Spaces.Technical Report. ICSI. 1995. P. 16.

109. Eiben A. E., Smith J. E. Popular Evolutionary Algorithm Variants. Natural Computing Series. Berlin, Heidelberg. 2015. P. 99–116. URL: https://doi.org/10.1007/978-3-662-44874-8_6.

110. Koza J. R. Genetic programming: On the programming of computers by means of natural selection. Cambridge, Mass : MIT Press, 1992. P. 819.

111. Sigaud O., Wilson S. W. Learning classifier systems: a survey. Soft Computing. 2007. Vol. 11. Iss. 11. P. 1065–1078. URL: https://doi.org/10.1007/s00500-007-0164-0.

112. Evolutionary Algorithms / T. Bartz-Beielstein et al. Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery. 2014. Vol. 4. Iss. 3. P. 178–195. URL: https://doi.org/10.1002/widm.1124.

113. Nissen V. A Brief Introduction to Evolutionary Algorithms from the Perspective of Management Science. Innovative Research Methodologies in Management. Cham. 2017. P. 165–210. URL: https://doi.org/10.1007/978-3-319-64394-6_8.
114. Hansen N., Ostermeier A. Completely Derandomized Self-Adaptation in Evolution Strategies. Evolutionary Computation. 2001. Vol. 9. Iss. 2. P. 159–195. URL: https://doi.org/10.1162/106365601750190398.
115. Xin Yao, Yong Liu, Guangming Lin. Evolutionary programming made faster. IEEE Transactions on Evolutionary Computation. 1999. Vol. 3. Iss. 2. P. 82–102. URL: https://doi.org/10.1109/4235.771163 .
116. Evolutionary Algorithms for Parameter Optimization–Thirty Years Later / T. H. W. Bäck et al. Evolutionary Computation. 2023. Vol. 31. Iss. 2. P. 81–122. URL: https://doi.org/10.1162/evco_a_00325.
117. Partial reinforcement optimizer: An evolutionary optimization algorithm / A. Taheri et al. Expert Systems with Applications. 2023. P. 122070. URL: https://doi.org/10.1016/j.eswa.2023.122070.
118. Evolutionary algorithms for hyperparameter optimization in machine learning for application in high energy physics / L. Tani et al. The European Physical Journal C. 2021. Vol. 81. Iss. 2. URL: https://doi.org/10.1140/epjc/s10052-021-08950-y.
119. Vincent A. M., Jidesh P. An improved hyperparameter optimization framework for AutoML systems using evolutionary algorithms. Scientific Reports. 2023. Vol. 13. Iss. 1. URL: https://doi.org/10.1038/s41598-023-32027-3.
120. Evolutionary Strategies for Parameter Optimization in Deep Learning Models / Shalini S. et al. International Journal of Intelligent Systems and Applications in Engineering. IJISAE, 2024. Vol. 12. Iss. 2s, P. 371–378.
121. Літвінчук Ю. А. Порівняльний аналіз оптимізації гіперпараметрів нейронних мереж. *Прикладна математика та інформаційні технології.* Міжнар. наук. конф. присвяченої 60-річчю кафедри прикладної математики та інформаційних технологій (м. Чернівці, 22-24 вересня 2022 р.): Чернівецький нац. ун-т, 2022. С.

181-183. URL: https://archer.chnu.edu.ua/jspui/bitstream/123456789/5858/1/AMIT2022-Materials.pdf

122. Малик І. В., Літвінчук Ю. А. Алгоритм CMA-ES для оптимізації гіперпараметрів нейронної мережі. *Молодіжна наука заради миру та розвитку.* Міжнар. наук.-практ. конф. присвячена Всесвітньому дню науки 9–11 листопада 2022 р. Чернівці, Україна. 2022. С. 647-649.
URL: https://archer.chnu.edu.ua/xmlui/handle/123456789/6979

123.

124. Bharadiya J. Convolutional Neural Networks for Image Classification. International Journal of Innovative Research in Science Engineering and Technology. 2023. Vol. 8, Iss. 5. P. 673–677. URL: https://doi.org/10.5281/zenodo.7952031.

125. Amari S.-i. Backpropagation and stochastic gradient descent method. Neurocomputing. 1993. Vol. 5, no. 4-5. P. 185–196. URL: https://doi.org/10.1016/0925-2312(93)90006-o.

126. Bottou, Léon.Large-Scale Machine Learning with Stochastic Gradient Descent. Proc. of COMPSTAT.2010. URL: DOI: 10.1007/978-3-7908-2604-3_16.

127. Hardt, Moritz & Recht, Benjamin & Singer, Yoram. Train faster, generalize better: Stability of stochastic gradient descent. 2015. P.10. URL: https://doi.org/10.48550/arXiv.1509.01240

128. Belete D. M., Huchaiah M. D. Grid search in hyperparameter optimization of machine learning models for prediction of HIV/AIDS test results. International Journal of Computers and Applications. 2021. P. 1–12. URL: https://doi.org/10.1080/1206212x.2021.1974663.

129. Liang H., Tamang A. Z., Weihua S. X. Stochastic Information Management in Smart Grid. Communications Surveys & Tutorials, IEEE. 2014. Vol. 16. P. 1746-1770. URL: 10.1109/SURV.2014.020614.00115.

130. Hamster C., van Heijster P. Waves in a Stochastic Cell Motility Model. Bulletin of Mathematical Biology. 2023. Vol. 85, Iss. 8. URL: https://doi.org/10.1007/s11538-023-01164-1.
131. Vastola J. J., Holmes W. R. Chemical Langevin equation: A path-integral view of Gillespie's derivation. Physical Review E. 2020. Vol. 101, Iss. 3. URL: https://doi.org/10.1103/physreve.101.032417.
132. Hansen N., Müller S. D., Koumoutsakos P. Reducing the Time Complexity of the Derandomized Evolution Strategy with Covariance Matrix Adaptation (CMA-ES). Evolutionary Computation. 2003. Vol. 11, Iss. 1. P. 1–18. URL: https://doi.org/10.1162/106365603321828970.
133. Hansen N., Ros R., Mauny N., Schoenauer M., Auger A. Impacts of invariance in search: When CMA-ES and PSO face ill-conditioned and non-separable problems. Applied Soft Computing. 2011. Vol. 11, Iss. 8. P. 5755–5769. URL: https://doi.org/10.1016/j.asoc.2011.03.
134. Voß T., Hansen N., Igel C. Improved step size adaptation for the MO-CMA-ES. GECCO '10: Proceedings of the 12th annual conference on Genetic and evolutionary computation. 2010 P. 487–494. URL: https://doi.org/10.1145/1830483.1830573.
135. Loshchilov I. A Computationally Efficient Limited Memory CMA-ES for Large Scale Optimization. GECCO '14: Proceedings of the 2014 Annual Conference on Genetic and Evolutionary ComputationJuly. 2014. P. 397–404. URL: https://doi.org/10.1145/2576768.2598294.
136. Arabas J., Jagodzinski D. Toward a Matrix-Free Covariance Matrix Adaptation Evolution Strategy. IEEE Transactions on Evolutionary Computation. 2020. Vol. 24, Iss. 1. P. 84–98. URL: https://doi.org/10.1109/tevc.2019.2907266.
137. A covariance matrix adaptation evolution strategy variant and its engineering application / Y. Liang et al. Applied Soft Computing. 2019. Vol. 83. P. 105680. URL: https://doi.org/10.1016/j.asoc.2019.105680.

138. Малик І.В., Літвінчук Ю.А. Адаптивний метод навчання обмеженої машини Больцмана з алгоритмом генерації та знищення нейронів. *Інформаційні технології: наука, техніка, технологія, освіта, здоров'я*: тези доп. XXVIII Міжнар. наук.-практ. конф. MicroCAD-2020. Ч. IV. Харків. 2020. С. 174. URL:https://science.kpi.kharkov.ua/wpcontent/uploads/2020/10/Tezi_chastina_4_2020.pdf

139. Sakamoto, N., Akimoto, Y. Modified box constraint handling for the covariance matrix adaptation evolution strategy. GECCO '17: Proceedings of the Genetic and Evolutionary Computation Conference Companion. 2017.P. 183–184. URL: https://doi.org/10.1145/3067695.3075986

140. Cross Entropy Covariance Matrix Adaptation Evolution Strategy for Solving the Bi-Level Bidding Optimization Problem in Local Energy Markets / D. Dabhi et al. Energies. 2022. Vol. 15. Iss. 13. P. 4838. URL: https://doi.org/10.3390/en15134838.

141. Ahrari A., Deb K., Preuss M. Multimodal Optimization by Covariance Matrix Self-Adaptation Evolution Strategy with Repelling Subpopulations. Evolutionary Computation. 2017. Vol. 25. Iss. 3. P. 439–471. URL: https://doi.org/10.1162/evco_a_00182.

142. Dang V.-H., Vien N. A., Chung T. A covariance matrix adaptation evolution strategy in reproducing kernel Hilbert space. Genetic Programming and Evolvable Machines. 2019. Vol. 20. Iss. 4. P. 479–501. URL: https://doi.org/10.1007/s10710-019-09357-1.

143. Малик І. В., Літвінчук Ю. А. Моделювання розширеного алгоритму CMA-ES на базі сумішей нормальних розподілів. *Інформаційне суспільство: технологічні, економічні та технічні аспекти становлення* (випуск 80). Матеріали Міжнародної наукової інтернет-конференції, 19 вересня 2023 р. Тернопіль, Україна, Переворськ, Польща. URL: http://www.konferenciaonline.org.ua/ua/article/id-1258/

144. D Gillespie D. T. Approximate accelerated stochastic simulation of chemically reacting systems. The Journal of Chemical Physics. 2001. Vol. 115. Iss. 4. P. 1716–1733. URL: https://doi.org/10.1063/1.1378322.
145. Kelley W. G., Peterson A. C. The Theory of Differential Equations. New York, NY: Springer New York, 2010. URL: https://doi.org/10.1007/978-1-4419-5783-2
146. Sekerci Y., Petrovskii S. Mathematical Modelling of Plankton–Oxygen Dynamics Under the Climate Change. *Bulletin of Mathematical Biology*. 2015. Vol. 77, Iss. 12. P. 2325–2353. URL: https://doi.org/10.1007/s11538-015-0126-0
147. Thiruthummal A., Kim E. Monte Carlo Simulation of Stochastic Differential Equation to Study Information Geometry. Entropy. 2022. Vol. 24, Iss. 8. P. 1113. URL: https://doi.org/10.3390/e24081113.
148. Abdulle A., Blumenthal A. Stabilized multilevel Monte Carlo method for stiff stochastic differential equations. Journal of Computational Physics. 2013. Vol. 251. P. 445–460. URL: https://doi.org/10.1016/j.jcp.2013.05.039.
149. Brand J., Yang M., Pahl E. Stochastic differential equation approach to understanding the population control bias in full configuration interaction quantum Monte Carlo. Physical Review. 2022. Vol. 105. P. 235144. URL: https://doi.org/10.1103/PhysRevB.105.235144.
150. Doumbia M., Oudjane N., Warin X. Unbiased Monte Carlo estimate of stochastic differential equations expectations. ESAIM: Probability and Statistics. 2017. Vol. 21. P. 56–87. URL: https://doi.org/10.1051/ps/2017001.
151. Spigler R. Monte Carlo-type simulation for solving stochastic ordinary differential equations. Mathematics and Computers in Simulation. 1987. Vol. 29, Iss. 3-4. P. 243–251. URL: https://doi.org/10.1016/0378-4754(87)90134-0.
152. Appleby J. A. D., Mao X. Stochastic stabilisation of functional differential equations. Systems & Control Letters. 2005. Vol. 54, Iss. 11. P. 1069–1081. URL: https://doi.org/10.1016/j.sysconle.2005.03.003.

153. Identifiability analysis for stochastic differential equation models in systems biology / A. P. Browning et al. Journal of The Royal Society Interface. 2020. Vol. 17, Iss. 173. P. 20200652. URL: https://doi.org/10.1098/rsif.2020.0652.
154. Kelly D., Melbourne I. Smooth approximation of stochastic differential equations. The Annals of Probability. 2016. Vol. 44, Iss. 1. P. 479–520. URL: https://doi.org/10.1214/14-aop979.
155. Shen J., Lu K. Wong–Zakai approximations and center manifolds of stochastic differential equations. Journal of Differential Equations. 2017. Vol. 263, Iss. 8. P. 4929–4977. URL: https://doi.org/10.1016/j.jde.2017.06.005.
156. Tsarkov Y. F., Yasinsky V. K., Malyk I. V. Stability in impulsive systems with Markov perturbations in averaging scheme. 2. Averaging principle for impulsive Markov systems and stability analysis based on averaged equations. Cybernetics and Systems Analysis. 2011. Vol. 47, Iss. 1. P. 44–54. URL: https://doi.org/10.1007/s10559-011-9288-4.
157. Priyadarshini I., Cotton C. A novel LSTM–CNN–grid search-based deep neural network for sentiment analysis. The Journal of Supercomputing. 2021. URL: https://doi.org/10.1007/s11227-021-03838-w.
158. Neural Architecture Search Using Deep Neural Networks and Monte Carlo Tree Search / L. Wang et al. Proceedings of the AAAI Conference on Artificial Intelligence. 2020. Vol. 34, Iss. 06. P. 9983–9991. URL: https://doi.org/10.1609/aaai.v34i06.6554.
159. An improved grid search algorithm to optimize SVR for prediction / Y. Sun et al. Soft Computing. 2021. Vol. 25, Iss. 7. P. 5633–5644. URL: https://doi.org/10.1007/s00500-020-05560-w.
160. Xue Y., Wang Y., Liang J. A self-adaptive gradient descent search algorithm for fully-connected neural networks. Neurocomputing. 2022. Vol. 478. P. 70–80. URL: https://doi.org/10.1016/j.neucom.2022.01.001.
161. Hansen N. and Ostermeier A., Ostermeier A. Completely derandomized self-adaptation in evolution strategies.Evolutionary computation, Vol. 9 (2). 2001. P. 159–195.

162. Loshchilov I., Schoenauer M., Sebag M. Self-adaptive surrogate-assisted covariance matrix adaptation evolution strategy. the fourteenth international conference, м. Philadelphia, Pennsylvania, USA. 2012 p. New York, New York, USA, 2012. URL: https://doi.org/10.1145/2330163.2330210.

163. Watanabe S., Le Roux J. Black box optimization for automatic speech recognition. ICASSP 2014 - 2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), м. Florence, Italy. 2014 p. 2014. URL: https://doi.org/10.1109/icassp.2014.6854202.

164. McLachlan G., Peel D. Finite Mixture Models. Hoboken, NJ, USA : John Wiley & Sons, Inc., 2000. URL: https://doi.org/10.1002/0471721182 .

165. Lee G., Scott C. EM algorithms for multivariate Gaussian mixture models with truncated and censored data. Computational Statistics & Data Analysis. 2012. Vol. 56, 9. P. 2816–2829. URL: https://doi.org/10.1016/j.csda.2012.03.003.

166. Buchen P. W., Kelly M. The Maximum Entropy Distribution of an Asset Inferred from Option Prices. The Journal of Financial and Quantitative Analysis. 1996. Vol. 31, 1. P. 143. URL: https://doi.org/10.2307/2331391.

167. Dempster A. P., Laird N. M., Rubin D. B. Maximum Likelihood from Incomplete Data Via theEMAlgorithm. Journal of the Royal Statistical Society: Series B (Methodological). 1977. Vol. 39, 1. P. 1–22. URL: https://doi.org/10.1111/j.2517-6161.1977.tb01600.x.

168. Karlis D. EM Algorithm for Mixed Poisson and Other Discrete Distributions. ASTIN Bulletin. 2005. Vol. 35, 01. P. 3–24. URL: https://doi.org/10.2143/ast.35.1.583163.

169. Karlis D., Xekalaki E. Minimum Hellinger distance estimation for Poisson mixtures. Computational Statistics & Data Analysis. 1998. Vol. 29, 1. P. 81–103. URL: https://doi.org/10.1016/s0167-9473(98)00047-4.

170. Anaya-Izquierdo K., Marriott P. Local mixture models of exponential families. Bernoulli. 2007. Vol. 13, 3. P. 623–640. URL: https://doi.org/10.3150/07-bej6170.

171. A variational Expectation–Maximization algorithm for temporal data clustering / H. El Assaad et al. Computational Statistics & Data Analysis. 2016. Vol. 103. P. 206–228. URL: https://doi.org/10.1016/j.csda.2016.05.007.
172. Computational Complexity Evaluation of Neural Network Applications in Signal Processing. Freire, Pedro & Srivallapanondh, Sasipim & Napoli, Antonio & Prilepsky, Jaroslaw & Turitsyn, Sergei. 2022. 10.48550/arXiv.2206.12191.
173. An Efficient Hybrid of an Ant Lion Optimizer and Genetic Algorithm for a Model Parameter Identification Problem / O. Roeva et al. Mathematics. 2023. Vol. 11, Iss. 6. P. 1292. URL: https://doi.org/10.3390/math11061292.
174. Honey Bees Inspired Optimization Method: The Bees Algorithm / B. Yuce et al. Insects. 2013. Vol. 4, Iss. 4. P. 646–662. URL: https://doi.org/10.3390/insects4040646.
175. Genetic Algorithm Based on Natural Selection Theory for Optimization Problems / M. A. Albadr et al. Symmetry. 2020. Vol. 12, Iss. 11. P. 1758. URL: https://doi.org/10.3390/sym12111758.
176. Xuefeng W., Chen M. Application of Mathematical Model Based on Optimization Theory and Particle Swarm Algorithm in Radar Station Layout Optimization. Journal of Physics: Conference Series. 2021. Vol. 1848, Iss. 1. P. 012087. URL: https://doi.org/10.1088/1742-6596/1848/1/012087.
177. CMA-ES: evolution strategies and covariance matrix adaptation/ N. Hansen, A. Auger. Proceedings of the 13th Annual Conference Companion on Genetic and Evolutionary Computation. GECCO' 11: Conference Series. 2011. P. 991–1010. URL: https://doi.org/10.1145/2001858.2002123
178. Dorsey R. E., Mayer W. J. Genetic Algorithms for Estimation Problems With Multiple Optima, Nondifferentiability, and Other Irregular Features. Journal of Business & Economic Statistics. 1995. Vol. 13, Iss. 1. P. 53–66. URL: https://doi.org/10.1080/07350015.1995.10524579.
179. Малик І. В., Літвінчук Ю. А. Побудова еволюційної стратегії на основі сумішей. *Інформаційне суспільство: технологічні, економічні та технічні аспекти*

становлення (випуск 70): матеріали Міжнародної наукової інтернет-конференції, 22-23 вересня 2022 р. Тернопіль, Україна, Переворськ, Польща. URL: http://www.konferenciaonline.org.ua/ua/article/id-656/

180. Alhijawi B., Awajan A. Genetic algorithms: theory, genetic operators, solutions, and applications. Evolutionary Intelligence. 2023. URL: https://doi.org/10.1007/s12065-023-00822-6.
181. Літвінчук Ю.А., Малик І.В. Розширений алгоритм стратегії еволюції адаптації коваріаційної матриці. Буковинський математичний журнал. 2022. 10 (2). С. 137–143. URL: https://doi.org/10.31861/bmj2022.02.09.
182. Sundberg R. Statistical Modelling by Exponential Families. Cambridge University Press, 2019. URL: https://doi.org/10.1017/9781108604574
183. Variations on the Clustering Algorithm BIRCH / B. Lorbeer et al. Big Data Research. 2018. Vol. 11. P. 44–53. URL: https://doi.org/10.1016/j.bdr.2017.09.002.
184. Lang A., Schubert E. BETULA: Numerically Stable CF-Trees for BIRCH Clustering. Similarity Search and Applications. Cham, 2020. P. 281–296. URL: https://doi.org/10.1007/978-3-030-60936-8_22.
185. Grouping Multidimensional Data: Recent Advances in Clustering / ред.: J. K. (Editor), C. N. (Editor), M. T. (Editor). Springer, 2006. P. 268.
186. Guha S., Rastogi R., Shim K. Cure: an efficient clustering algorithm for large databases. Information Systems. 2001. Vol. 26, 1. P. 35–58. URL: https://doi.org/10.1016/s0306-4379(01)00008-4.
187. Yun-Tao Qian, Qing-Song Shi, Qi Wang. CURE-NS: a hierarchical clustering algorithm with new shrinking scheme. 2002 International Conference on Machine Learning and Cybernetics, Beijing, China. URL: https://doi.org/10.1109/icmlc.2002.1174512.
188. Літвінчук Ю. А. Про одне узагальнення еволюційних алгоритмів. International Scientific Technical Journal «Problems of Control and Informatics». 2023. 68(6), с. 64–75. URL: DOI: https://doi.org/10.34229/1028-0979-2023-6-4.

189. Малик І.В., Літвінчук Ю.А. Про один підхід побудови самоадаптивних алгоритмів на основі сумішей розподілу. Буковинський математичний журнал. 2023. 11 (2). С. 183-189. URL: https://doi.org/10.31861/bmj2023.02.18.
190. Barbu V., Lefter C. Chapter 1 Optimal control of ordinary differential equations. Handbook of Differential Equations: Ordinary Differential Equations. 2006. P. 1–75. URL: https://doi.org/10.1016/s1874-5725(05)80003-3.
191. Warga J. Optimal Control of Ordinary Differential Equations. Optimal Control of Differential and Functional Equations. 1972. P. 346–406. URL: https://doi.org/10.1016/b978-0-12-735150-6.50011-7.
192. Särkkä S., Solin A. Applied Stochastic Differential Equations. Cambridge University Press, 2019.
193. N. I. Portenko, A. V. Skorokhod, On the existence of ε-optimal Markov strategies for controlled diffusion processesthe book: Questions of Statistics and Control of Random Processes, Institute of Mathematics of the Ukranian SSR Academy of Sciences, Kiev, 1973, 204–207.
194. PORTENKO N., SALEHI H., SKOROKHOD A. On optimal filtering of multitarget tracking systems based on point processes observations. Random Operators and Stochastic Equations. 1997. Vol. 5, 1. P. 1–34. URL: https://doi.org/10.1515/rose.1997.5.1.1.
195. PORTENKO N., SALEHI H., SKOROKHOD A. Optimal filtering in target tracking in presence of uniformly distributed errors and false targets. Random Operators and Stochastic Equations. 1998. Vol. 6, 3. URL: https://doi.org/10.1515/rose.1998.6.3.213.
196. Bather J. A., Fleming W. H., Rishel R. W. Deterministic and Stochastic Optimal Control. Journal of the Royal Statistical Society. Series A (General). 1976. Vol. 139, 4. P. 546. URL: https://doi.org/10.2307/2344363.
197. Levajković T., Mena H., Tuffaha A. The Stochastic LQR Optimal Control with Fractional Brownian Motion. Generalized Functions and Fourier Analysis. Cham, 2017. P. 115–151. URL: https://doi.org/10.1007/978-3-319-51911-1_8

198. Kasinathan R., Kasinathan R., Sandrasekaran V. Solvability and optimal controls of neutral stochastic integro-differential equations driven by fractional Brownian motion. Journal of Control and Decision. 2023. P. 1–8. URL: https://doi.org/10.1080/23307706.2023.2197911.
199. Mishura Y., Zili M. Stochastic Analysis of Mixed Fractional Gaussian Processes. Elsevier, 2018. 210 p.
200. Mishura Y. Stochastic Calculus for Fractional Brownian Motion and Related Processes (Lecture Notes in Mathematics Book 1929). Springer, 2008. 416 p.
201. Paul L. Random functions: general theory with special reference to Laplacian random functions. Berkeley : University of California Press, 1953. 390 p.
202. Stochastic Calculus for Fractional Brownian Motion and Applications / F. Biagini et al. London : Springer London, 2008. URL: https://doi.org/10.1007/978-1-84628-797-8.
203. Kurtz T. G., Stockbridge R. H. Existence of Markov Controls and Characterization of Optimal Markov Controls. SIAM Journal on Control and Optimization. 1998. Vol. 36, 2. P. 609–653. URL: https://doi.org/10.1137/s0363012995295516.
204. Janssen, Jacques & Manca, Raimondo.Applied Semi-Markov Processes. New York : Springer-Verlag, 2006. URL: https://doi.org/10.1007/0-387-29548-8..
205. Zong G., Ren H. Guaranteed cost finite-time control for semi-Markov jump systems with event-triggered scheme and quantization input. International Journal of Robust and Nonlinear Control. 2019. Vol. 29, 15. P. 5251–5273. URL: https://doi.org/10.1002/rnc.4672.
206. Boel R., Kohlmann M. Semimartingale Models of Stochastic Optimal Control, with Applications to Double Martingales. SIAM Journal on Control and Optimization. 1980. Vol. 18, 5. P. 511–533. URL: https://doi.org/10.1137/0318038.
207. Т.О. Лукашів, В.К. Ясинський Стійкість стохастичних систем випадкової структури з марковськими перемиканнями і збуреннями. Кибернетика и системный анализ. 2020. Vol. 56, 2. P. 157–165.

208. Т.О. Лукашів, І.В. Юрченко, В.К. Ясинський. Про необхідні та достатні умови стійкості в середньому квадратичному лінійних стохастичних диференціально-різницевих рівнянь у частинних похідних під дією зовнішних збурень типу випадкових величин. Кибернетика и системный анализ. 2020. Vol. 56, 2. P. 157–165.
209. Lukashiv T., Malyk I. Stability of controlled stochastic dynamic systems of random structure with markov switches and poisson perturbations. Bukovinian Mathematical Journal. 2022. Vol. 10, 1. P. 85–99. URL: https://doi.org/10.31861/bmj2022.01.08 .
210. Lukashiv T., Litvinchuk Y., Malyk I., Golebiewska A., Nazarov P. Stabilization of Stochastic Dynamical Systems of a Random Structure with Markov Switches and Poisson Perturbations. Mathematics. 2023. Vol. 11. Iss. 3. P. 1-22.

ДОДАТОК

СПИСОК ПУБЛІКАЦІЙ ЗА ТЕМОЮ ДИСЕРТАЦІЇ

Наукові праці у періодичних наукових виданнях, проіндексованих у наукометричній базі даних Scopus:

1. Lukashiv T., Litvinchuk Y., Malyk I., Golebiewska A., Nazarov P. Stabilization of Stochastic Dynamical Systems of a Random Structure with Markov Switches and Poisson Perturbations. Mathematics. 2023. Vol. 11. Iss. 3. P. 1-22. (Scopus) (Q2 – URL:https://www.scimagojr.com/journalsearch.php?q=21100830702&tip=sid&clean=0).

Наукові праці у виданнях, включених до переліку наукових фахових видань України:

2. Літвінчук Ю.А., Малик І.В. Розширений алгоритм стратегії еволюції адаптації коваріаційної матриці. *Буковинський математичний журнал.* 2022. 10 (2). С. 137–143. URL: https://doi.org/10.31861/bmj2022.02.09.
3. Літвінчук Ю. А. Про одне узагальнення еволюційних алгоритмів. International Scientific Technical Journal «Problems of Control and Informatics». 2023. 68(6), с. 64–75. URL: DOI: https://doi.org/10.34229/1028-0979-2023-6-4.
4. Малик І.В., Літвінчук Ю.А. Про один підхід побудови самоадаптивних алгоритмів на основі сумішей розподілу. Буковинський математичний журнал. 2023. 11 (2). С. 183-189. URL: https://doi.org/10.31861/bmj2023.02.18.

Наукові праці, які засвідчують апробацію матеріалів дисертації:

5. Малик І.В., Літвінчук Ю.А. Адаптивний метод навчання обмеженої машини Больцмана з алгоритмом генерації та знищення нейронів. *Інформаційні технології: наука, техніка, технологія, освіта, здоров'я*: тези доп. XXVIII Міжнар. наук.-практ. конф. MicroCAD-2020. Ч. IV. Харків. 2020. С. 174. URL:https://science.kpi.kharkov.ua/wpcontent/uploads/2020/10/Tezi_chastina_4_2020.pdf

6. Малик І. В., Літвінчук Ю. А. Побудова еволюційної стратегії на основі сумішей. *Інформаційне суспільство: технологічні, економічні та технічні аспекти становлення (випуск 70*): матеріали Міжнародної наукової інтернет-конференції, 22-23 вересня 2022 р. Тернопіль, Україна, Переворськ, Польща. URL: http://www.konferenciaonline.org.ua/ua/article/id-656/
7. Літвінчук Ю. А. Порівняльний аналіз оптимізації гіперпараметрів нейронних мереж. *Прикладна математика та інформаційні технології.* Міжнар. наук. конф. присвяченої 60-річчю кафедри прикладної математики та інформаційних технологій (м. Чернівці, 22-24 вересня 2022 р.): Чернівецький нац. ун-т, 2022. С. 181-183. URL: https://archer.chnu.edu.ua/jspui/bitstream/123456789/5858/1/AMIT2022-Materials.pdf
8. Малик І. В., Літвінчук Ю. А. Алгоритм CMA-ES для оптимізації гіперпараметрів нейронної мережі. *Молодіжна наука заради миру та розвитку.* Міжнар. наук.-практ. конф. присвячена Всесвітньому дню науки 9–11 листопада 2022 р. Чернівці, Україна. 2022. С. 647-649. URL: https://archer.chnu.edu.ua/xmlui/handle/123456789/6979
9. Малик І. В., Літвінчук Ю. А. Моделювання розширеного алгоритму CMA-ES на базі сумішей нормальних розподілів. *Інформаційне суспільство: технологічні, економічні та технічні аспекти становлення* (випуск 80). Матеріали Міжнародної наукової інтернет-конференції, 19 вересня 2023 р. Тернопіль, Україна, Переворськ, Польща. URL: http://www.konferenciaonline.org.ua/ua/article/id-1258/