МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ЧЕРКАСЬКИЙ ДЕРЖАВНИЙ ТЕХНОЛОГІЧНИЙ УНІВЕРСИТЕТ
ФАКУЛЬТЕТ ЕЛЕКТРОННИХ ТЕХНОЛОГІЙ, АВТОТРАНСПОРТУ ТА МАШИНОБУДУВАННЯ

МЕТАЕВРИСТИЧНІ МЕТОДИ ОПТИМІЗАЦІЇ В MathCAD

Навчально-методичний посібник
для здобувачів галузі знань
G «Інженерія, виробництво та будівництво»
(15 «Автоматизація та приладобудування»
17 «Електроніка, автоматизація та електронні комунікації»)
усіх форм навчання

Черкаси 2025

УДК 004.942 (075.8)
М33

Затверджено вченою радою ФЕТАМ,
протокол №1 від 18.02.2025 р.,
згідно з рішенням кафедри приладобудування,
мехатроніки та комп'ютеризованих технологій
протокол № 6 від 23.01.2025 р.

Упорядники: Гальченко В.Я., *д.т.н., професор*, Трембовецька Р.В., *д.т.н., професор*, Тичков В.В., *к.т.н., доцент*
Рецензент: Федоров Є.Є., *д.т.н., професор*.

МЕТАЕВРИСТИЧНІ МЕТОДИ ОПТИМІЗАЦІЇ В MathCAD. Навчально-методичний посібник для здобувачів галузі знань G «Інженерія, виробництво та будівництво» (15 «Автоматизація та приладобудування», 17 «Електроніка, автоматизація та електронні комунікації») усіх форм навчання. [Електронний ресурс] / [Упоряд. В.Я. Гальченко, Р.В. Трембовецька, В.В. Тичков]; М-во освіти и науки України, Черкас. держ. технол. ун-т. - Черкаси: ЧДТУ, 2025. - 65 с.

У навчальному посібнику викладено коротко основні теоретичні положення математичних методів та докладний практичний матеріал щодо метаевристичних методів розв'язку задач глобальної оптимізації у середовищі універсального математичного пакета MathCAD. Істотну увагу приділено комп'ютерній реалізації аналізованих методів, містяться комплекти завдань для самостійної роботи та велика кількість прикладів, що сприяють кращому розумінню та засвоєнню матеріалу.

Навчальне електронне видання
мережного використовування

МЕТАЕВРИСТИЧНІ МЕТОДИ ОПТИМІЗАЦІЇ В MathCAD

Навчально-методичний посібник
для здобувачів освітнього ступенів для здобувачів галузі знань
галузі знань G «Інженерія, виробництво та будівництво» (15
«Автоматизація та приладобудування», 17 «Електроніка, автоматизація та
електронні комунікації») усіх форм навчання

упорядники: Гальченко Володимир Якович
Трембовецька Руслана Володимирівна
Тичков Володимир Володимирович
В авторській редакції.

ЗМІСТ

ВСТУП

Сучасний світ стикається зі складними оптимізаційними задачами, які вимагають нестандартних підходів: від проектування ефективних логістичних мереж до навчання глибоких нейронних архітектур. Традиційні методи оптимізації, часто виявляються неефективними через обмежену масштабованість, залежність від похідних або наявність численних локальних оптимумів. Саме тут на допомогу приходять метаевристичні методи — клас алгоритмів, які поєднують стохастичні стратегії з механізмами, натхненними природними системами. Їхня універсальність дозволяє застосовувати їх у задачах, де математична модель надто складна для аналітичного аналізу, а простір розв'язів має високу розмірність або нерегулярну структуру.

Метаевристики не є алгоритмами з жорстко заданою логікою. Натомість вони визначають загальні принципи пошуку, які можна адаптувати під конкретну задачу. Їхня сила полягає в здатності імітувати природні явища, зокрема біологічні системи - поведінку мурашок у пошуку їжі (мурашині алгоритми), координацію зграї птахів (PSO), еволюційні механізми (генетичні алгоритми); фізичні процеси - охолодження металу (імітація відпалу), рух часток у силовому полі (гравітаційний пошук); соціальні моделі - колективний розум (роєвий інтелект), взаємодію між агентами (алгоритми гармонійного пошуку).

Ці алгоритми працюють за принципом балансу між дослідженням (exploration) — пошуком нових областей простору розв'язів та використанням (exploitation) — локальним вдосконаленням знайдених кандидатів. Наприклад, у генетичних алгоритмах мутація відповідає за дослідження, а схрещування — за використання найкращих особин. Метаевристики особливо корисні в галузях, де точність поступової збіжності поступається місцем швидкості отримання субоптимальних розв'язів.

Останнім часом спостерігається тенденція до інтелектуалізації метаевристик: впровадження механізмів машинного навчання для адаптації параметрів алгоритмів під динамічні умови задачі. Незважаючи на гнучкість, метаевристики мають недоліки: відсутність гарантій збіжності, алгоритми можуть "пропустити" глобальний оптимум через стохастичну природу; чутливість до параметрів;: деякі підходи вимагають значних ресурсів, що підвищує обчислювальну вартість.

Метаевристики залишаються активною областю досліджень, особливо в контексті оптимізації складних систем із динамічними обмеженнями.

У цьому посібнику детально розглянуто архітектуру ключових методів, їхню теоретичну основу та практичні кейси застосування.

В посібнику детально розглядаються окремі теоретичні положення навчальних дисциплін «Оптимізація прийняття рішень у техніці», «Математичне моделювання процесів і систем та методи їх оптимізації» та формуються у здобувачів вміння та навички їх практичного застосування шляхом індивідуального виконання завдань.

Завдання розраховані для здобувачів галузі знань G, які опановують навчальні компоненти.

В системі дистанційної освіти ЧДТУ з дисциплін «Оптимізація прийняття рішень у техніці» https://moodle.chdtu.edu.ua/course/view.php?id=53, «Математичне моделювання процесів і систем та методи їх оптимізації» http://fet.moodle.chdtu.edu.ua/course/view.php?id=18 є електронна версія цього посібника. Також наведені тестові завдання для самоконтролю до кожної теми відповідного змістового модулю. Після вивчення відповідного теоретичного лекційного матеріалу здобувач самостійно проходить тест-самоконтроль, отримує відповідну кількість балів, та аналізує свій рівень знань і може виконувати тестування необмежену кількість разів.

МЕТАЕВРИСТИЧНІ МЕТОДИ ГЛОБАЛЬНОЇ ОПТИМІЗАЦІЇ

Метаевристики є потужним і популярним останнім часом класом оптимізаційних методів, що дають змогу знаходити розв'язки для широкого кола задач. Термін метаевристика походить від композиції двох слів "мета" + "евристика". "Мета" означає "за його межами, у верхньому рівні". "Евристика" походить від дієслова heuriskein, що означає "знайти". Мета метаевристики полягає в ефективному дослідженні простору пошуку для знаходження оптимальних розв’язків або близьких до них. Перевага метаевристик полягає в їхній здатності розв'язувати складні задачі без знання особливостей простору пошуку, завдяки чому ці методи дають змогу розв'язувати важко розв'язувані задачі оптимізації. Спрощено можна розглядати метаевристики як алгоритми, що реалізують прямий випадковий пошук можливих розв'язків задачі, оптимальних або близьких до оптимальних, доки не буде виконано якоїсь умови або досягнуто заданого числа ітерацій.

Клас метаевристичних алгоритмів охоплює алгоритми оптимізації мурашиною та бджолиною колоніями, бактеріальні алгоритми, оптимізації роєм часток, еволюційні обчислення, включно з генетичними алгоритмами, метод імітації відпалу та алгоритм табу-пошуку і багато інших.

Розв'язання задач глобальної безумовної оптимізації не належить до числа тривіальних. У разі застосування для цих цілей детермінованих методів локального пошуку часто використовують стратегію мульти-старту, яка не гарантує в кінцевому підсумку знаходження глобального оптимуму. Стохастичні методи пошуку є більш перспективними для цих цілей, оскільки досліджують увесь простір пошуку значно ефективніше з подальшою локалізацією в областях, що становлять найбільший інтерес. Ройові алгоритми оптимізації, як різновид стохастичного методу пошуку, завдяки своїм біонічним особливостям добре пристосовані для розв'язання подібних задач.

1. АЛГОРИТМИ ОПТИМІЗАЦІЇ РОЄМ ЧАСТОК PSO

Біонічні передумови методу. Алгоритм оптимізації роєм часток PSO (Particle Swarm Optimization) відноситься до біонічних мультиагентних методів глобальної оптимізації, що моделюють соціальну поведінку взаємодіючих агентів. Ідея методу PSO належить J. Kennedy та R. Eberhart, які вперше його сформулювали та успішно застосували для розв'язку оптимізаційних задач та навчання нейронних мереж. Коріння роєвої оптимізації сягає ще більш ранніх робіт з комп'ютерного моделювання переміщення живих істот у пташиній зграї або косяку риб. У роботі висловлена ідея, що стала основою методу: «Принаймні теоретично, окремі члени зграї можуть отримати вигоду від досліджень та попереднього досвіду всіх інших членів зграї при пошуку їжі. Ця перевага стає вирішальною, і навіть перевершує конкуренцію, щоразу, коли харчові ресурси розташовані на шляху випадково чи невідомо». Тобто соціальний поділ інформації серед представників одного виду дає еволюційні зиски всім членам популяції. Ця гіпотеза домінування колективного інтелекту стала фундаментальною при розробці оптимізації роєм часток.

Канонічний алгоритм оптимізації роєм часток PSO. Задача безумовної глобальної оптимізації формулюється як задача мінімізації цільової функції $f(\mathbf{X})$ у просторі пошуку **D**:

$$f(\mathrm{X}) \to \min,\ \mathrm{X} \in \mathrm{D} = \{\mathrm{x} \in R^d\}, \tag{1.1}$$

де область **D** є речовий гіперкуб з розмірністю d,

X - векторний аргумент оптимізованої функції f, а її глобальний розв'язок досягається в точці $\mathbf{X}^*$.

У методі PSO рій часток є сукупність точок-розв'язків, що переміщуються у просторі у пошуках глобального оптимуму. При своєму русі частки намагаються покращити знайдений ними раніше розв'язок та обмінюються інформацією зі своїми сусідами.

Позначимо сукупність позицій часток рою через

$$\mathrm{X} = \{\mathrm{x}_1, \mathrm{x}_2, \ldots, \mathrm{x}_s\}, \tag{1.2}$$

де s – кількість часток у рої.

Під позицією i-ї частки розуміється сукупність її координат $(x_{i1}, x_{i2}, \ldots, x_{id})$ у просторі пошуку розмірністю d; $i = \overline{1, s}$. Під час проведення оптимізації зазвичай досить 10-30 часток. Рій дає можливість знайти глобальний оптимум навіть коли кількість часток у ньому менша за розмірність d простору пошуку.

На початковому етапі роботи алгоритму PSO проводиться випадкова ініціалізація рою часток. Якщо відсутня якась апріорна інформація про

оптимізовану функцію, то найпростіше початкові положення часток генерувати за формулою:

$$x_{ij} = \text{rand}(x_{j\min}, x_{j\max}), \tag{1.3}$$

де x_{ij} - *j*-я координата *i*-ї частки;

$\text{rand}(x_{j\min}, x_{j\max})$ - випадкове число з рівномірним законом розподілу на інтервалі, що визначає межі простору пошуку для *j*-го виміру.

З роєм часток також асоціюється множина векторів їх швидкостей:

$$\text{V} = \{ \text{v}_1, \text{v}_2, \ldots, \text{v}_s \}. \tag{1.4}$$

На початковому етапі всі швидкості будемо вважати рівними нулю. Проте практика показує, що кращі результати дає формула:

$$v_{ij} = [\text{rand}(x_{j\min}, x_{j\max}) - x_{ij}] / 2, \tag{1.5}$$

де v_{ij} - *j*-я компонента швидкості *i*-ї частки.

Такий метод задання початкових швидкостей гарантує, що у наступній ітерації алгоритму жодна з часток не вийде за межі простору пошуку.

На наступних кроках алгоритму компоненти швидкостей та позицій часток оновлюються за формулами:

$$v'_{ij} = wv_{ij} + c_1 r_1 (p_{ij} - x_{ij}) + c_2 r_2 (g_j - x_{ij}),$$

$$x'_{ij} = x_{ij} + v'_{ij}, \tag{1.6}$$

де $\mathbf{v}'_i$ і $\mathbf{x}'_i$ – нова швидкість та положення *i*-ї частки;

$\mathbf{p}_i$ – найкраще положення, знайдене нею раніше (*personal best*);

$\mathbf{g}$ – найкращий розв'язок, знайдений всім роєм (*global best*);

w - інерційний коефіцієнт;

c_1 і c_2 – відповідно когнітивний та соціальний коефіцієнти;

r_1, r_2 - випадкові числа, що рівномірно генеруються на інтервалі $[0,1]$, різні для кожної координати.

У формулах (1.6) передбачається, що час між оновленнями стану часток рою $\Delta t = 1$.

Якщо в процесі оптимізації частка виходить за межі простору пошуку, відбувається обнуління відповідних компонентів її швидкості, а сама частка повертається до найближчої границі.

Інерційний коефіцієнт *w* визначає вплив попередньої швидкості частки на її нове значення. Розмір когнітивного коефіцієнта c_1 характеризує ступінь індивідуальної поведінки частки та її прагнення повертатися до найкращого

знайденого нею раніше розв’язку, тоді як значення соціального коефіцієнта c_2 задає ступінь колективної поведінки та прагнення рухатися у бік найкращого розв’язку її сусідів (рис. 1.1).

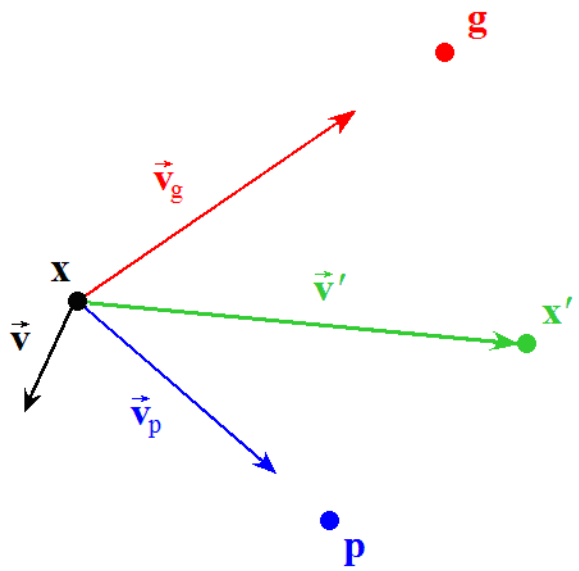


Рисунок 1.1 – Геометрична ілюстрація правила оновлення швидкостей

На рисунку через $\mathbf{v}_p = c_1 r_1 (\mathbf{p} - \mathbf{x})$ і $\mathbf{v}_g = c_2 r_2 (\mathbf{g} - \mathbf{x})$ позначені відповідно когнітивний та соціальний компоненти нової швидкості частки. Випадкові числа r_1, r_2 вносять у пошук елемент випадковості.

Величина вагових коефіцієнтів рівняння (1.6) вибирається у наступних діапазонах:

$$\begin{cases} 0 < w \le 1; \\ 1 < c_1, c_2 \le 2, \end{cases} \tag{1.7}$$

які забезпечують збіжність цього методу оптимізації.

Для підтримки балансу між локальним та глобальним пошуком, чисельні значення коефіцієнтів c_1 і c_2 зазвичай вибираються рівними. Стагнаційний аналіз, проведений у дослідженнях, показав, що чисельні значення коефіцієнтів $w = 1/(2\ln 2) \approx 0,72$ і $c = 0.5 + \ln 2 \approx 1,19$ забезпечують у більшості випадків найкращі результати та стабільність пошуку. Також можливі різні способи динамічного підстроювання параметрів рою, проте адаптація вимагає додаткових початкових ітерацій алгоритму, що може призвести до збільшення кількості обчислень цільової функції, необхідних для знаходження оптимуму.

Рій має пам'ять про найкращі розв’язки, знайдені його окремими частками і всім роєм в цілому. Під час ініціалізації початкові позиції часток вважаються найкращими. На кожній наступній ітерації алгоритму PSO після застосування

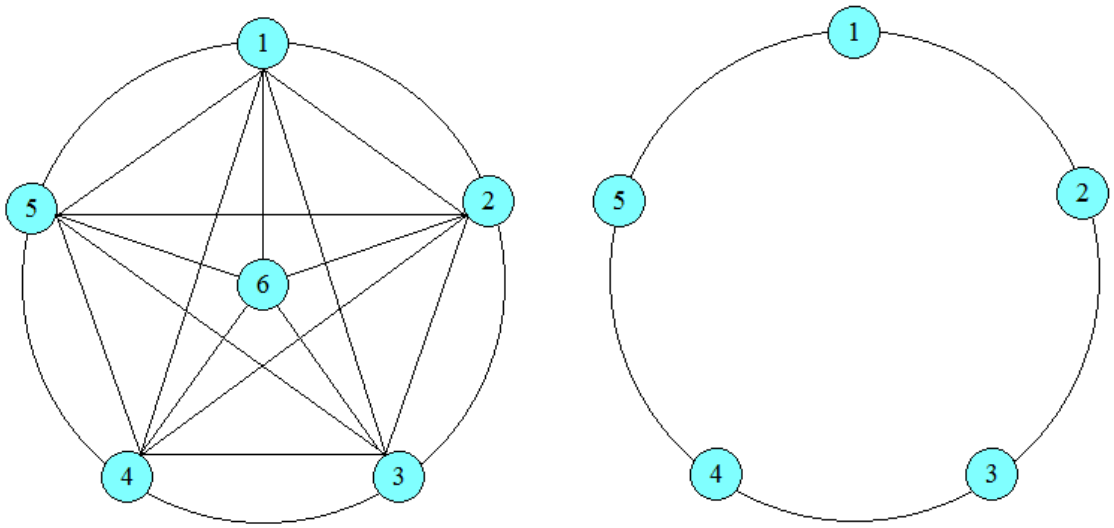
формул (1.6) індивідуальні кращі позиції кожної частки $\mathbf{p}_i$ і найкращий знайдений роєм розв'язок $\mathbf{g}$ оновлюється за правилами:

$$\begin{cases} \mathbf{p}_i = \mathbf{x}_i, & \text{якщо } f(\mathbf{x}_i) < f(\mathbf{p}_i), \\ \mathbf{g} = \mathbf{p}_i, & \text{якщо } f(\mathbf{p}_i) < f(\mathbf{g}). \end{cases} \tag{1.8}$$

Оскільки розв'язок рою $\mathbf{g}$ є найкращим розв'язком, знайденим однією з його часток, то достатньо лише пам'ятати її номер *gbest*. Якщо глобальний найкращий розв'язок рою збігається з найкращим розв'язок, знайденим часткою, то додавання соціального компонента швидкості $c_2 r_2(\mathbf{g} - \mathbf{x})$ не здійснюється. Слід звернути увагу на те, що для різних координат частки генеруються різні випадкові числа r_1, r_2.

Вибір топології зв'язків рою часток. Ключовою особливістю методу PSO є наявність зв'язків між частками, які визначають, наскільки ефективним є процес передачі інформації між окремими агентами рою. Виділяється два основних підходи їх організації: *gbest* та *lbest*.

Канонічна версія PSO використовує підхід *gbest*, у цьому випадку кожна частка рою пов'язана з усіма іншими частками (топологія «зірка» – рис. 1.2, а).



а) б)

Рисунок 1.2 -Варіанти топології рою часток: а) «зірка»; б) «кільце»

У результаті кожна частка прагне переміщатися у бік найкращого розв'язку $\mathbf{g}$ знайденого всім роєм. У локальному методі *lbest* частка обмінюється інформацією лише з кількома сусідами, орієнтуючись на найкращий знайдений ними розв'язок $\mathbf{g}_i$. При цьому сусідство не обов'язково означає, що частки знаходяться поруч, а лише визначає, які особини рою інформують цю *i*-у частку.

При використанні підходу *gbest* рій, як правило, швидше сходиться до розв'язку, проте висока швидкість збіжності призводить до менш ретельного

дослідження простору розв’язків. У методі *lbest* швидкість збіжності до розв’язку нижча, проте, ймовірність попадання в локальний оптимум менша.

Найпростішим випадком, що реалізує підхід *lbest*, є топологія «кільце» (рис. 1.2 б). У цьому випадку кожна *i*-я частка рою має лише двох інформаторів: (*i*-1)-у та (*i*+1)-у частки; $i = \overline{2, s-1}$. При цьому 1-у частку інформує 2-а та *s*-а частка, а *s*-у – частку з номерами 1 та $(s-1)$.

Оскільки найкращої топології зв'язків між частками рою, що підходить для будь-якої задачі оптимізації, не існує, то часто використовується випадкова топологія, яка може змінюватися від ітерації до ітерації. Для визначення сусідства застосовується матриця інцидентності **L** розміру $s \times s$, елементи якої $L_{ij} = 1$ у разі, коли частка *j* інформує частку *i* та $L_{ij} = 0$ в іншому випадку. Слід зауважити, що ця матриця в загальному випадку несиметрична, тобто з того, що *i*-а частка інформує *j*-у не випливає наявність зворотного обміну інформацією про знайдений найкращий розв’язок.

Для мультимодальних задач та задач високої розмірності бажано, щоб середня кількість сусідів у частки була не дуже велика. З іншого боку, збільшення кількості інформаторів зазвичай збільшує швидкість збіжності алгоритму. А іноді корисно, щоб частка якийсь час зовсім не мала сусідів і проводила лише локальний пошук навколо свого найкращого розв’язку **p** знайденого на минулих ітераціях. Тому, виходячи з цих міркувань, для отримання кращих результатів розумніше використовувати змінну кількість інформаторів.

На рис. 1.3 показаний приклад випадкової топології сусідства в рої, що складається з 10 часток, котрий розбитий на три підрої: $\{1, 2, 5, 8\}$, $\{3, 4, 6, 9, 10\}$ і $\{7\}$. Останній підрій складається всього з однієї частки, яка здійснює лише локальний пошук розв’язку в її околі.

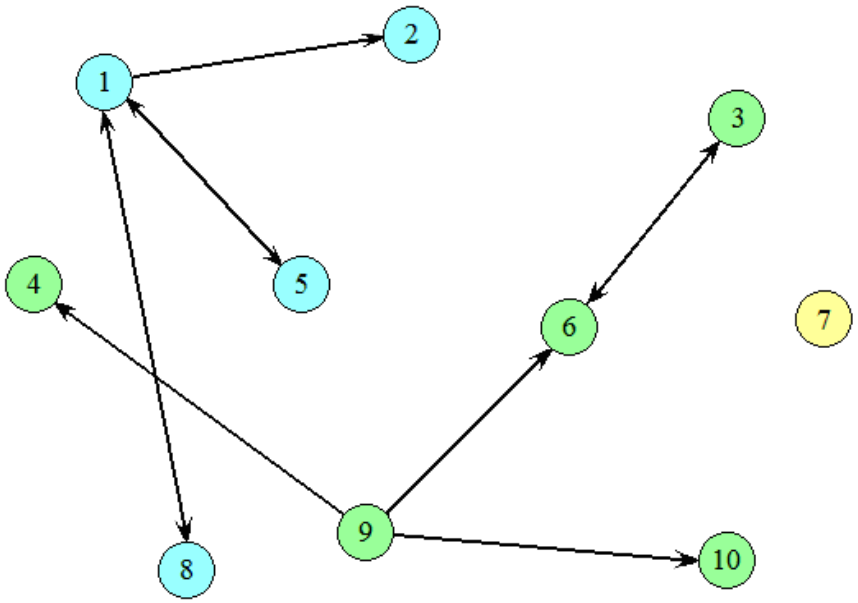


Рисунок 1.3 – Приклад випадкової організації зв'язків між частками

У розглянутому прикладі кожна частка в середньому має по одному інформатору. У топології присутні як односторонні, так і двосторонні зв'язки. Частка 9, хоча інформує інші три частки, сама інформаторів немає. Відповідна цьому випадку матриця зв'язків представлена в наступному вигляді:

$$L = \begin{pmatrix} 1 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \end{pmatrix} \quad (1.9)$$

Топологія зв'язків не залишається однією і тією ж протягом всієї роботи алгоритму, а періодично змінюється. У цьому алгоритмі оптимізації зв'язки між частками оновлюються, якщо після завершення поточної ітерації не відбулося поліпшення глобального розв’язку рою **g**. Ця ситуація може свідчити про те, що випадково згенерована топологія виявилася невдалою і, отже, її потрібно змінити, або ж наявні підрої вже знайшли свої локальні оптимуми, і для забезпечення глобального пошуку необхідно перебудувати зв'язки.

При випадковій організації зв'язків кожна частка повинна інформувати не більше K інших часток. З цією метою в матриці зв'язків **L** у кожному стовпці j вибирається випадково K інформованих часток i, причому не виключається ймовірність повторного вибору. Усі частки рою інформують самі себе, тому для елементів головної діагоналі матриці інцидентності $L_{ii} = 1$. За такого способу ініціалізації зв'язків кожна j-а частка може одночасно інформувати від 1 до $K+1$ часток, а довільна i-а частка може мати від 1 до s інформаторів, але з нерівномірно розподіленою ймовірністю. Ймовірність $p(n)$ того, що частка має рівно n інформаторів, включаючи себе, визначається за такою формулою:

$$p(n) = C_{s-1}^{n-1} \left(\frac{K}{s} \right)^{n-1} \left(1 - \frac{K}{s} \right)^{s-n}, \quad (1.10)$$

де C_{s-1}^{n-1} - число поєднань з $(s-1)$ елементів по $(n-1)$ елементу.

Графік розподілу ймовірності встановлення різної кількості зв'язків між частками рою для кількох значень K при розмірі рою $s = 20$ показано на рис. 1.4.

З графіка видно, що з найбільшою ймовірністю у кожної частки буде близько K інформаторів, проте їх кількість може бути і меншою, що корисно для локального пошуку, і більше, що сприятливо впливає на глобальний пошук і швидкість збіжності.

При використанні випадкової динамічно змінної топології деякі ділянки коду алгоритму ускладнюються.

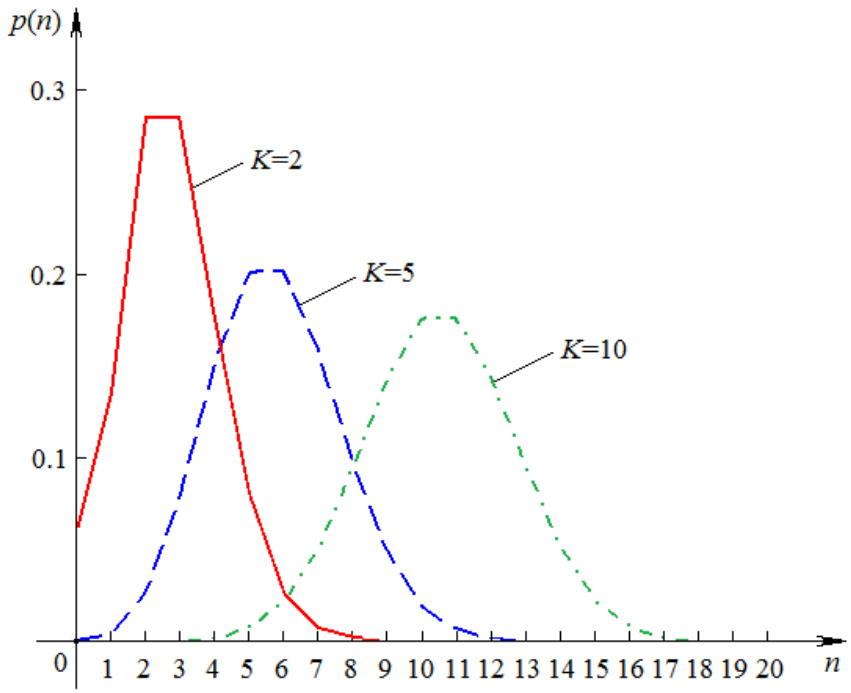


Рисунок 1.4 - Розподіл ймовірності встановлення n зв'язків

При випадковій генерації одного зв'язку для i-ї частки у рої розміром s ймовірність того, що деяка j-а виявиться її інформатором дорівнює $1/s$, ймовірність того, що вона не буде обрана, становить $1-1/s$. Тоді ймовірність, що при встановленні K випадкових зв'язків (можливо з повторами) ця частка не потрапить до списку її сусідів складає $(1-1/s)^K$. В результаті ймовірність того, що j-а частка буде інформатором i-ї має значення $p = 1-(1-1/s)^K$. Імовірність того, що j-а частка не стане інформатором i-ї після закінчення t ітерацій алгоритму PSO дорівнює $p = (1-1/s)^{Kt}$. Остаточно, ймовірність того, що після t ітерацій інформація про знайдений j-ю часткою розв'язок буде передана i-й, виявиться рівною:

$$p = 1-(1-1/s)^{Kt}. \qquad (1.11)$$

Ця ймовірність збільшується дуже швидко зі зростанням t (рисунок 1.5), тому при використанні випадкової топології параметр K не повинен бути занадто великим, щоб уникнути надмірно швидкого поширення інформації в рої, а отже, і можливості передчасної збіжності до локального розв'язку. При використанні випадкової топології для більшості випадків хороші результати вдається досягти при $K = 3$ (s = 20).

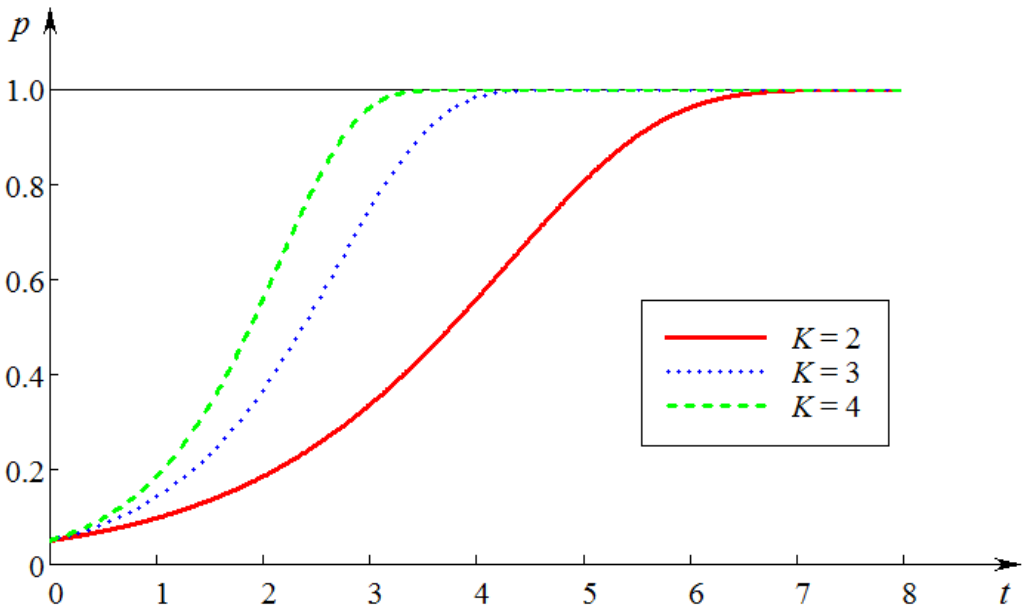


Рисунок 1.5 - Ймовірність повної поінформованості рою через t-ітерацій (s = 20)

З формули (1.11) випливає, що для того, щоб довільні дві частки рою обмінялися інформацією з ймовірністю p, повинна пройти кількість ітерацій, що обчислюються згідно з виразом.

$$t = \frac{\ln\left[\dfrac{\ln(1-p)}{\ln(1-1/s)}\right]}{\ln K}. \qquad (1.12)$$

Приклад 1. Задана багатоекстремальна цільова функція

$$x_1 = \frac{x_1}{7{,}5} - 2; \quad x_2 = \frac{x_2}{5} - 3; \; f_1 = 3\cdot(x_1 - 1)\cdot e^{-x_1^2-(x_2+1)^2};$$

$$f_2 = 10\cdot\left[0.2\cdot x_1 - x_1^3 - x_2^5\right]\cdot e^{-\left[x_1^2+x_2^2\right]}; \; f_3 = \frac{e^{-\left[(x_1+1)^2+x_2^2\right]}}{3}; \; f(x_1, x_2) = f_1 + f_2 + f_3$$

Знайти мінімум цільової функції методом рою часток PSO, використовуючи *gbest* підхід.

Порядок виконання завдання:

1. Задати значення інерційного, когнітивного, соціального коефіцієнтів.
2. Задати багатоекстремальну цільову функцію векторного аргументу.
3. Перетворити вид функції від векторного аргументу до незалежних аргументів.
4. Графічно дослідити поведінку багатоекстремальної функції за допомогою вбудованої функції MathCAD ***CreateMesh (Створити сітку)***.
5. Записати функцію пошуку екстремуму PSO_G(s, xmin, xmax, Iter_MAX, Fitness). Задати вихідні дані: s – число часток у рої, максимальне число ітерацій, нижні та верхні границі зміни аргументів функції.
6. Отримати значення шуканих змінних, використовуючи функцію PSO_G(s, xmin, xmax, Iter_MAX, Fitness).
7. Перевірити розв'язок задачі з використанням функції ***Minimize(),*** задаючи початкові значення змінних в околі глобального оптимуму. Отримати значення цільової функції у точці екстремуму.

Приклад виконання завдання у MathCAD

Приклад 1. Дана багатоекстремальна цільова функція

$$f_1 = 3\cdot(x_1 - 1)\cdot e^{-x_1^2-(x_2+1)^2};$$

$$f_2 = 10\cdot\left[0.2\cdot x_1 - x_1^3 - x_2^5\right]\cdot e^{-\left[x_1^2+x_2^2\right]};$$

$$f_3 = \frac{e^{-\left[(x_1+1)^2+x_2^2\right]}}{3}$$

Знайти мінімум цільової функції методом рою часток PSO, використовуючи підхід gbest.

АЛГОРИТМ ОПТИМИЗАЦІЇ РОЄМ ЧАСТОК PSO - GBEST

ORIGIN := 1

$w := \frac{1}{2\cdot\ln(2)}$ - інерційний коефіцієнт

c1 := 0.5 + ln(2) - когнітивний коефіцієнт

c2 := 0.5 + ln(2) - соціальний коефіцієнт

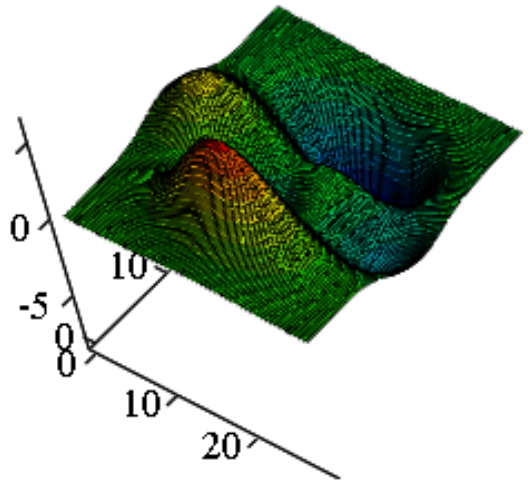
$$\text{Fitness}(x) := \left| \begin{array}{l} x_1 \leftarrow \left(\frac{x_1}{7.5}\right) - 2 \\ x_2 \leftarrow \left(\frac{x_2}{5}\right) - 3 \\ \text{Part1} \leftarrow 3\cdot(x_1 - 1)\cdot e^{-(x_1)^2-(x_2+1)^2} \\ \text{Part2} \leftarrow 10\cdot\left[0.2\cdot x_1 - (x_1)^3 - (x_2)^5\right]\cdot e^{-\left[(x_1)^2+(x_2)^2\right]} \\ \text{Part3} \leftarrow \frac{e^{-\left[(x_1+1)^2+(x_2)^2\right]}}{3} \\ \text{Part1} + \text{Part2} + \text{Part3} \end{array} \right.$$

- багатоекстремальна цільова функція векторного аргументу

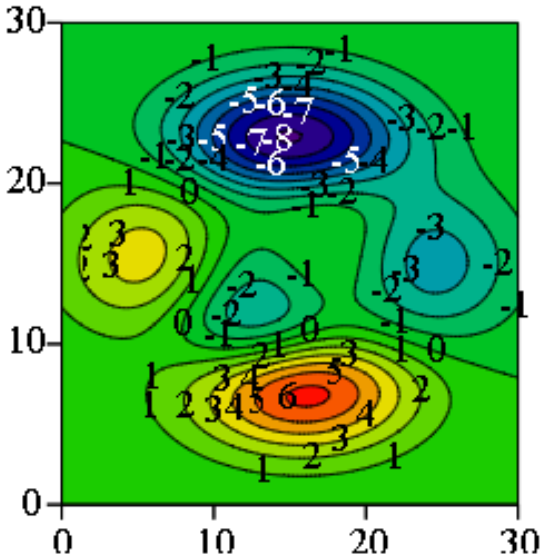
$f(x,y) := \text{Fitness}\left(\begin{pmatrix} x \\ y \end{pmatrix}\right)$ - перетворення виду функції від векторного аргументу до незалежних аргументів

$f(1,1) = 0.011252$ - перевірка коректного опису функції обчислення в контрольній точці

$Z := \text{CreateMesh}(f, 0, 30, 0, 30, 100, 100)$



Z



Z

$\text{PSO\_G}(s, xmin, xmax, Iter\_MAX, \text{Fitness}) :=$

```
d ← length(xmin)
for i ∈ 1..s
  for j ∈ 1..d
    x_i,j ← rnd(xmin_j − xmax_j) + xmax_j
    v_i,j ← ⌊(rnd(xmin_j − xmax_j) + xmax_j) − x_i,j⌋ / 2
    p_i,j ← x_i,j
for i ∈ 1..s
  fit_fun_i ← Fitness⌈(x^T)^⟨i⟩⌉
  fp_i ← fit_fun_i
gbest ← 1
for i ∈ 2..s
  gbest ← i if fp_i < fp_gbest
k ← 0
while 1
  for i ∈ 1..s
    for j ∈ 1..d
      r1 ← rnd(1)
      r2 ← rnd(1)
      v_i,j ← w·v_i,j + c1·r1·(p_i,j − x_i,j)
      v_i,j ← v_i,j + c2·r2·(p_gbest,j − x_i,j) if i ≠ gbest
      x_i,j ← x_i,j + v_i,j
      if x_i,j < xmin_j
        x_i,j ← xmin_j
        v_i,j ← 0
      if x_i,j > xmax_j
        x_i,j ← xmax_j
        v_i,j ← 0
  for i ∈ 1..s
```

```
    fit_fun_i ← Fitness[(x^T)^⟨i⟩]
    if fit_fun_i < fp_i
        fp_i ← fit_fun_i
        for j ∈ 1 .. d
            p_i,j ← x_i,j
  for i ∈ 1 .. s
    gbest ← i if fp_i < fp_gbest
  k ← k + 1
  return (p^T)^⟨gbest⟩ if k > Iter_MAX
```

$S := 10$ - число часток в рою, s = 10 - 30

$M\_it := 10^3$ - максимальне число ітерацій

$xmin := \begin{pmatrix} 0 \\ 0 \end{pmatrix}$ - нижні границі зміни аргументів функції

$xmax := \begin{pmatrix} 30 \\ 30 \end{pmatrix}$ - верхні границі зміни аргументів цільової функції

$VV := PSO\_G(S, xmin, xmax, M\_it, Fitness)$ - виклик функції оптимізації

$VV = \begin{pmatrix} 14.931915 \\ 22.906883 \end{pmatrix}$ - знайдені значення шуканих змінних

$Fitness(VV) = -8.106178$ - значення цільової функції в точці екстремуму

ПЕРЕВІРКА РОЗВ'ЯЗКУ ЗАДАЧІ

$x := 8 \qquad y := 18$ - вихідні значення змінних, вибрані з графіку ліній рівня цільової функції в околі глобального оптимуму

$VV\_prov := Minimize(f, x, y)$

$VV\_prov = \begin{pmatrix} 14.931915 \\ 22.906886 \end{pmatrix}$ - знайдений розв'язок

$Fitness(VV\_prov) = -8.106178$ - значення цільової функції в точці екстремуму

Приклад 2. Задана багатоекстремальна цільова функція

$$x_1 = \frac{x_1}{7,5} - 2; \quad x_2 = \frac{x_2}{5} - 3; \; f_1 = 3 \cdot (x_1 - 1) \cdot e^{-x_1^2 - (x_2+1)^2};$$

$$f_2 = 10 \cdot \left[0.2 \cdot x_1 - x_1^3 - x_2^5\right] \cdot e^{-\left[x_1^2 + x_2^2\right]}; \; f_3 = \frac{e^{-\left[(x_1+1)^2 + x_2^2\right]}}{3}; \; f(x_1, x_2) = f_1 + f_2 + f_3$$

Знайти мінімум цільової функції методом рою часток PSO, використовуючи *lbest* підхід.

Приклад виконання завдання у MathCAD

Приклад 2. Дана багатоекстремальня цільова функція

$$f_1 = 3 \cdot (x_1 - 1) \cdot e^{-x_1^2 - (x_2+1)^2};$$

$$f_2 = 10 \cdot \left[0.2 \cdot x_1 - x_1^3 - x_2^5\right] \cdot e^{-\left[x_1^2 + x_2^2\right]};$$

$$f_3 = \frac{e^{-\left[(x_1+1)^2 + x_2^2\right]}}{3}$$

Знайти мінімум цільової функції методом рою часток PSO, використовуючи підхід lbest.

АЛГОРИТМ ОПТИМІЗАЦІЇ РОЄМ ЧАСТОК PSO - LBEST

ORIGIN := 1

w := $\frac{1}{2 \cdot \ln(2)}$ - інерційний коефіцієнт

c1 := 0.5 + ln(2) - когнітивний коефіцієнт

c2 := 0.5 + ln(2) - соціальний коефіцієнт

Fitness(x) :=

$x_1 \leftarrow \left(\frac{x_1}{7.5}\right) - 2$

$x_2 \leftarrow \left(\frac{x_2}{5}\right) - 3$

$Part1 \leftarrow 3 \cdot (x_1 - 1) \cdot e^{-(x_1)^2 - (x_2+1)^2}$

$Part2 \leftarrow 10 \cdot \left[0.2 \cdot x_1 - (x_1)^3 - (x_2)^5\right] \cdot e^{-\left[(x_1)^2 + (x_2)^2\right]}$

$Part3 \leftarrow \frac{e^{-\left[(x_1+1)^2 + (x_2)^2\right]}}{3}$

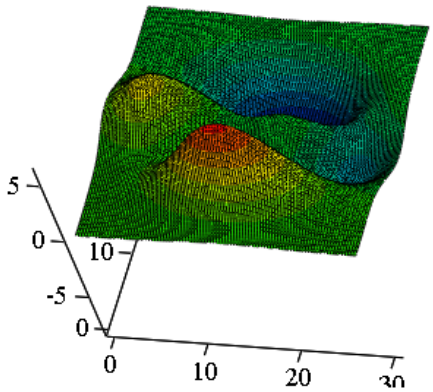
Part1 + Part2 + Part3

- багатоекстремальна цільова функція векторного аргументу

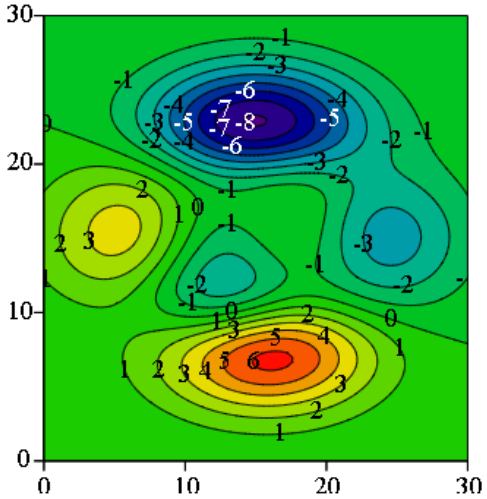
$f(x,y) := \text{Fitness}\left(\begin{pmatrix} x \\ y \end{pmatrix}\right)$ - перетворення виду функції від векторного аргументу до незалежних аргументів

$f(1,1) = 0.011252$ - перевірка коректного опису функції обчисленням в контрольній точці

$Z := \text{CreateMesh}(f,0,30,0,30,100,100)$



Z



Z

$Z := \text{CreateMesh}(f,0,30,0,30,100,100)$

$\text{PSO\_L}(s,xmin,xmax,\varepsilon,\text{Fitness}) :=$

$d \leftarrow \text{length}(xmin)$

for $i \in 1 .. s$

  for $j \in 1 .. d$

    $x_{i,j} \leftarrow \text{rnd}(xmin_j - xmax_j) + xmax_j$

    $v_{i,j} \leftarrow \dfrac{\left[\left(\text{rnd}(xmin_j - xmax_j) + xmax_j\right) - x_{i,j}\right]}{2}$

    $p_{i,j} \leftarrow x_{i,j}$

for $i \in 1 .. s$

  $fit\_fun_i \leftarrow \text{Fitness}\left[\left(x^T\right)^{\langle i \rangle}\right]$

  $fp_i \leftarrow fit\_fun_i$

$lbest \leftarrow 1$

for $i \in 2 .. s$

  $lbest \leftarrow i$ if $fp_i < fp_{lbest}$

$m \leftarrow 2$

$k \leftarrow 0$

$fp\_krit_1 \leftarrow 1000000$

$fp\_krit_m \leftarrow fp_{lbest}$

while $\left(\left|fp\_krit_m - fp\_krit_{m-1}\right| > \varepsilon\right)$

  for $i \in 1 .. s$

    $lbest \leftarrow i$

    $fmin \leftarrow fp_i$

    $neighbor1 \leftarrow i - 1$

    $neighbor2 \leftarrow i + 1$

    $neighbor1 \leftarrow s$ if $i = 1$

    $neighbor2 \leftarrow 1$ if $i = s$

    if $fp_{neighbor1} < fmin$

      $fmin \leftarrow fp_{neighbor1}$

      $lbest \leftarrow neighbor1$

    $lbest \leftarrow neighbor1$ if $fp_{neighbor2} < fmin$

    for $j \in 1 .. d$

      $r1 \leftarrow \text{rnd}(1)$

      $r2 \leftarrow \text{rnd}(1)$

      $v_{i,j} \leftarrow w \cdot v_{i,j} + c1 \cdot r1 \cdot \left(p_{i,j} - x_{i,j}\right)$

      $v_{i,j} \leftarrow v_{i,j} + c2 \cdot r2 \cdot \left(p_{lbest,j} - x_{i,j}\right)$ if $i \neq lbest$

```
        x_{i,j} ← x_{i,j} + v_{i,j}
        if x_{i,j} < xmin_j
            x_{i,j} ← xmin_j
            v_{i,j} ← 0
        if x_{i,j} > xmax_j
            x_{i,j} ← xmax_j
            v_{i,j} ← 0
    for i ∈ 1 .. s
        fit_fun_i ← Fitness[(x^T)^⟨i⟩]
        if fit_fun_i < fp_i
            fp_i ← fit_fun_i
            for j ∈ 1 .. d
                p_{i,j} ← x_{i,j}
    for i ∈ 1 .. s
        lbest ← i if fp_i < fp_lbest
    k ← k + 1
    m ← m + 1
    fp_krit_m ← fp_lbest
    Sol_Point ← (p^T)^⟨lbest⟩
    for i ∈ 1 .. d
        Sol_X_i ← Sol_Point_i
    Sol_X_{d+1} ← fp_lbest
Sol_X
```

$S := 40$ - число часток в рою, s = 10 - 30

$\varepsilon := 10^{-5}$ - задана точність

$xmin := \begin{pmatrix} 0 \\ 0 \end{pmatrix}$ - нижні границі зміни аргументів функції

$xmax := \begin{pmatrix} 30 \\ 30 \end{pmatrix}$ - верхні границі зміни аргументів цільової функції

$VV := PSO\_L(S, xmin, xmax, \varepsilon, Fitness)$ - виклик функції оптимізації

$VV = \begin{pmatrix} 15.913026 \\ 22.951848 \\ -7.968956 \end{pmatrix}$ - знайдені значення шуканих змінних, останній елемент вектора розв'язку - значення цільової функції в точці екстремуму

ПЕРЕВІРКА РОЗВ'ЯЗКУ

x := 14 y := 19 - початкові значення змінних, вибрані з графіку ліній рівня цільової функції в околі глобального оптимуму

VV_prov := Minimize(f, x, y)

$$VV\_prov = \begin{pmatrix} 14.931915 \\ 22.906886 \end{pmatrix}$$ - знайдений розв'язок

Fitness(VV_prov) = −8.106178 - значення цільової функції в точці екстремуму

Приклад 3. Задана багатоекстремальна цільова функція

$$x_1 = \frac{x_1}{7,5} - 2; \quad x_2 = \frac{x_2}{5} - 3; \; f_1 = 3 \cdot (x_1 - 1) \cdot e^{-x_1^2 - (x_2+1)^2};$$

$$f_2 = 10 \cdot \left[0.2 \cdot x_1 - x_1^3 - x_2^5 \right] \cdot e^{-\left[x_1^2 + x_2^2\right]}; \quad f_3 = \frac{e^{-\left[(x_1+1)^2 + x_2^2\right]}}{3}; \; f(x_1, x_2) = f_1 + f_2 + f_3$$

Знайти мінімум цільової функції методом рою часток PSO, використовуючи стратегію випадкової топології зв'язків у рої.

Приклад виконання завдання у MathCAD

Приклад 3. Дана багатоекстремальна цільова функція

$$f_1 = 3 \cdot (x_1 - 1) \cdot e^{-x_1^2 - (x_2+1)^2};$$

$$f_2 = 10 \cdot \left[0.2 \cdot x_1 - x_1^3 - x_2^5 \right] \cdot e^{-\left[x_1^2 + x_2^2\right]};$$

$$f_3 = \frac{e^{-\left[(x_1+1)^2 + x_2^2\right]}}{3}$$

Знайти мінімум цільової функції методом рою часток PSO, використовуючи стратегію випадкової топології зв'язків у рої.

АЛГОРИТМ ОПТИМІЗАЦІЇ РОЄМ ЧАСТОК PSO - RND

ORIGIN := 1

$$w := \frac{1}{2 \cdot \ln(2)}$$ - інерційний коефіцієнт

c1 := 0.5 + ln(2) - когнітивний коефіцієнт

c2 := 0.5 + ln(2) - соціальний коефіцієнт

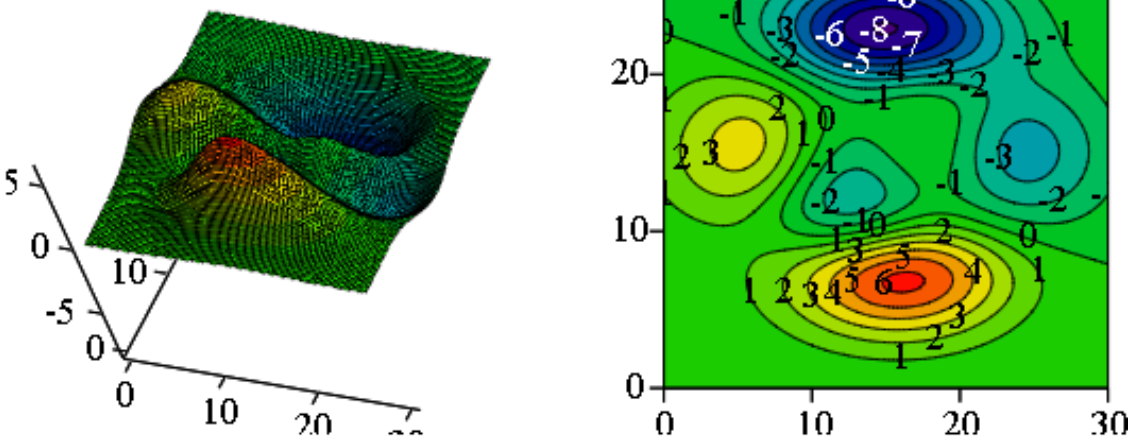
$$\text{Fitness}(x) := \left| \begin{array}{l} x_1 \leftarrow \left(\frac{x_1}{7.5}\right) - 2 \\ x_2 \leftarrow \left(\frac{x_2}{5}\right) - 3 \\ \text{Part1} \leftarrow 3\cdot(x_1 - 1)\cdot e^{-(x_1)^2-(x_2+1)^2} \\ \text{Part2} \leftarrow 10\cdot\left[0.2\cdot x_1 - (x_1)^3 - (x_2)^5\right]\cdot e^{-\left[(x_1)^2+(x_2)^2\right]} \\ \text{Part3} \leftarrow \frac{e^{-\left[(x_1+1)^2+(x_2)^2\right]}}{3} \\ \text{Part1} + \text{Part2} + \text{Part3} \end{array} \right.$$

- багатоекстремальна цільова функція векторного аргументу

$f(x,y) := \text{Fitness}\left(\begin{pmatrix} x \\ y \end{pmatrix}\right)$ - перетворення виду функції від векторного аргументу до незалежних аргументів

$f(1,1) = 0.011252$ - перевірка коректного опису функції обчисленням в контрольній точці

$Z := \text{CreateMesh}(f, 0, 30, 0, 30, 100, 100)$



Z Z

$Z := \text{CreateMesh}(f, 0, 30, 0, 30, 100, 100)$

$\text{PSO\_RND}(s, K, xmin, xmax, Iter\_MAX, Fitness) :=$

$d \leftarrow \text{length}(xmin)$

for $i \in 1 .. s$

  for $j \in 1 .. d$

    $x_{i,j} \leftarrow \text{rnd}(xmin_j - xmax_j) + xmax_j$

    $v_{i,j} \leftarrow \dfrac{\left[\left(\text{rnd}(xmin_j - xmax_j) + xmax_j\right) - x_{i,j}\right]}{2}$

    $p_{i,j} \leftarrow x_{i,j}$

for $i \in 1 .. s$

  $fit\_fun_i \leftarrow \text{Fitness}\left[\left(x^T\right)^{\langle i \rangle}\right]$

  $fp_i \leftarrow fit\_fun_i$

$gbest \leftarrow 1$

for $i \in 2 .. s$

  $gbest \leftarrow i$ if $fp_i < fp_{gbest}$

$kk \leftarrow 0$

$improvement \leftarrow 0$

$relink \leftarrow 1$

---

while 1

  $previous \leftarrow fp_{gbest}$

  if $relink = 1$

    for $i \in 1 .. s$

      for $j \in 1 .. s$

        $L_{i,j} \leftarrow 1$ if $i = j$

        $L_{i,j} \leftarrow 0$ otherwise

    for $j \in 1 .. s$

      for $k \in 1 .. K$

        $i \leftarrow \text{round}(\text{rnd}(s - 1)) + 1$

        $L_{i,j} \leftarrow 1$

    $improvement \leftarrow 0$

  for $i \in 1 .. s$

```
        lbest ← i
        fmin ← fp_i
        for neighbor ∈ 1 .. s
          if fp_neighbor < fmin        if L_i,neighbor = 1
            fmin ← fp_neighbor
            lbest ← neighbor
        for j ∈ 1 .. d
          r1 ← rnd(1)
          r2 ← rnd(1)
          v_i,j ← w·v_i,j + c1·r1·(p_i,j − x_i,j)
          v_i,j ← v_i,j + c2·r2·(p_lbest,j − x_i,j)  if i ≠ lbest
          x_i,j ← x_i,j + v_i,j
          if x_i,j < xmin_j
            x_i,j ← xmin_j
            v_i,j ← 0
          if x_i,j > xmax_j
            x_i,j ← xmax_j
            v_i,j ← 0
      for i ∈ 1 .. s
        fit_fun_i ← Fitness[(x^T)^⟨i⟩]
        if fit_fun_i < fp_i
          fp_i ← fit_fun_i
          for j ∈ 1 .. d
            p_i,j ← x_i,j
      for i ∈ 1 .. s
        gbest ← i  if fp_i < fp_gbest
      if fp_gbest < previous
        improvement ← improvement + 1
        relink ← 0
```

relink ← 1 otherwise

relink ← 1 if improvement = 5

kk ← kk + 1

return $(p^T)^{\langle gbest \rangle}$ if kk > Iter_MAX

S := 20 - число часток в рої, s = 10 - 30

$M\_it := 10^3$ - максимальне число ітерацій

$xmin := \begin{pmatrix} 0 \\ 0 \end{pmatrix}$ - нижні границі зміни аргументів функції

$xmax := \begin{pmatrix} 30 \\ 30 \end{pmatrix}$ - верхні границі зміни аргументів цільової функції

VV := PSO_RND(S, 3, xmin, xmax, M_it, Fitness)

- виклик функції оптимізації

$VV = \begin{pmatrix} 14.931912 \\ 22.906883 \end{pmatrix}$ - знайдені значення шуканих змінних

Fitness(VV) = −8.106178 - значення цільової функції в точці екстремуму

ПЕРЕВІРКА РОЗВ'ЯЗКУ ЗАДАЧІ

x := 8 y := 18 - початкові значення змінних, вибрані з графіку ліній рівня цільової функції в околі глобального оптимуму

VV_prov := Minimize(f, x, y)

$VV\_prov = \begin{pmatrix} 14.931915 \\ 22.906886 \end{pmatrix}$ - знайдений розв'язок

Fitness(VV_prov) = −8.106178 - значення цільової функції в точці екстремуму

ЗАВДАННЯ ДО САМОСТІЙНОЇ РОБОТИ

Завдання 1. Знайти глобальний екстремум заданої функції за допомогою одного з алгоритмом PSO.

Вибір алгоритму здійснюватиме залежно від кратності номера варіанта: кратність 2 – PSO-LBEST; кратність 3 – PSO-RND; для інших випадків – PSO-GBEST.

№ п/п	*Функція*	*Область пошуку*
1	$f(X)=\left(x_1^2+x_2^2\right)^{0,25}\cdot\sin^2\left(50\cdot\left(x_1^2+x_2^2\right)^{0,1}\right)+1\to\min$ $X^*=(0;0),\ \ f^*=0$	$[-\alpha;\alpha]^2$ $\alpha=100$
2	$f(X)=\left[1+(x_1+x_2+1)^2\cdot\begin{pmatrix}19-14\cdot x_1+3\cdot x_1^2-\\-14\cdot x_2+6\cdot x_1\cdot x_2+3\cdot x_2^2\end{pmatrix}\right]\times$ $\times\left[30+(2\cdot x_1+3\cdot x_2)^2\cdot\begin{pmatrix}18-32\cdot x_1+12\cdot x_1^2+\\+48\cdot x_2-36\cdot x_1\cdot x_2+27\cdot x_2^2\end{pmatrix}\right]\to\min$ $X^*=(0;-1),\ \ f^*=3$	$[-\alpha;\alpha]^2$ $\alpha=2$
3	$f(X)=0{,}5+\dfrac{\sin^2\sqrt{x_1^2+x_2^2}-0.5}{\left(1.0+0.001\cdot\left(x_1^2+x_2^2\right)\right)^2}\to\min$ $X^*=(0;0),\quad f^*=0$	$[-\alpha;\alpha]^2$ $\alpha=0{,}5$
4	$f(X)=(x_1-x_2)^2+\left(\dfrac{(x_1+x_2-10)}{3}\right)^2\to\min$ $X^*=(5;5),\quad f^*=0$	$[-\alpha;\alpha]^2$ $\alpha=10$

№ п/п	Функція	Область пошуку
5	$f(X) = -21.5 - x_1 \cdot \sin(4 \cdot \pi \cdot x_1) - x_2 \cdot \sin(20 \cdot \pi \cdot x_2) \to \min$ $X^* = (11.62523; 5.72082)$ $f^* = -38.85$	$[\alpha;\beta]^2$ $\alpha_1 = -3$ $\beta_1 = 12.1$ $\alpha_2 = 4.1$ $\beta_2 = 5.8$
6	$f(X) = \left(x_1^3 - 3 \cdot x_1 \cdot x_2^2 - 1\right)^2 + \left(3 \cdot x_1^2 \cdot x_2 - x_3^3\right)^2 \to \min$ $X^* = \left\{(-1;0), \left(-\frac{1}{2}; \sqrt{\frac{1}{2}}\right), \left(-\frac{1}{2}; \sqrt{\frac{3}{2}}\right)\right\}$ $f^* = 0$	$[-\alpha;\alpha]^2$ $\alpha = 2$
7	$f(X) = \left(x_1^2 - 4 \cdot x_2\right)^2 + \left(x_2^2 - 2 \cdot x_1 + 4 \cdot x_2\right)^2 \to \min$ $X^* = \{(0;0), (1.695415; 0.7186082)\}, \quad f^* = 0$	$[\alpha;\beta]^2$ $\alpha = -1$ $\beta = 4$
8	$f(X) = \sum_{i=1}^{\lvert X \rvert} [x_i]$ $X^* = \{[-5.12;-5); [-5.12;-5); \ldots; [-5.12;-5)\}$ $f^* = 0$	$[-\alpha;\alpha]^2$ $\alpha = 5.12$
9	$f(X) = \left(x_1^2 + x_2 - 11\right)^2 + \left(x_1 + x_2^2 - 7\right)^2 \to \min$ $X^* = \begin{Bmatrix} (3;2), (-2.805118; 3.131312), (3.584428; -1.848126), \\ (-3.779310; -3.283186) \end{Bmatrix}$ $f^* = 0$	$[-\alpha;\alpha]^2$ $\alpha = 6$

№ п/п	Функція	Область пошуку
10	$f(X)=\left(x_1^2-x_2^2\right)^2+\left(x_2^2-x_3^2\right)^2+\left(x_3^2-x_4^2\right)^2+$ $+\left(x_4^2-x_1^2\right)^2\to\min$ $X^*=(0;0;0;0),\ \ f^*=0$	$[\alpha;\beta]^4$ $\alpha=-1$ $\beta=10$
11	$f(X)=\sin^2\left(\pi\cdot\left(1+\frac{x_1-1}{4}\right)\right)+$ $+\sum_{i=2}^{9}\left(\frac{x_{i-1}-1}{4}\right)^2\cdot\left[1+10\cdot\sin^2\left(\pi\cdot\left(1+\frac{x_i-1}{4}\right)\right)\right]+$ $+\left(\frac{x_{10}-1}{4}\right)^2\to\min$ $X^*=(0;0;\ldots;0),\ \ f^*=0$	$[-\alpha;\alpha]^{\lvert X\rvert}$ $\alpha=10$
12	$f(X)=\sum_{i=1}^{5}(x_i-1)^2-\sum_{i=2}^{5}x_i\cdot x_{i-1}\to\min$ $X^*=(5;8;9;8;5),\ \ f^*=-30$	$[-\alpha;\alpha]^5$ $\alpha=25$
13	$f(X)=\sum_{i=1}^{\lvert X\rvert}\left(z_i^2-10\cdot\cos(2\cdot\pi\cdot z_i)+10\right)+f^0\to\min$ $X^*=\left(x_1^0;\ldots;x_{\lvert X\rvert}^0\right),\ \ f^*=f^0=-330$	$[-\alpha;\alpha]^{\lvert X\rvert}$ $\alpha=5$
14	$f(X)=\sum_{i=1}^{\lvert X\rvert}\frac{\sin(\pi\cdot k\cdot x_i)}{\pi\cdot k\cdot x_i}\to\min$ $X^*=(0;0;\ldots;0),\ \ f^*=0$	$[-\alpha;\alpha]^{\lvert X\rvert}$ $\alpha=0.5$

№ п/п	Функція	Область пошуку
15	$f(X)=\sum_{i=1}^{\lvert X\rvert} z_i^2 + f^0 \to \min$ $X^*=\left(x_1^0;\ldots;x_{\lvert X\rvert}^0\right),\ f^*=f^0=-450$	$[-\alpha;\alpha]^{\lvert X\rvert}$ $\alpha=100$
16	$f(X)=\left(x_1+10\cdot x_2\right)^2+5\cdot\left(x_3-x_4\right)^2+$ $+\left(x_2-2\cdot x_3\right)^4+10\cdot\left(x_1-x_4\right)^4\to\min$ $X^*=(3.0;-1.0;0;1.0)$	
17	$f(X)=100\cdot\left(x_2-x_1^2\right)^2+\left(1-x_1\right)^2+90\cdot\left(x_4-x_3^2\right)^2+$ $+\left(1-x_3\right)^2+10.1\cdot\left[\left(x_2-1\right)^2+\left(x_4-1\right)^2\right]+$ $19.8\cdot\left(x_2-1\right)\cdot\left(x_4-1\right)\to\min$	
18	$f(X)=\left[15-x_1\cdot\left(1-x_2\right)\right]^2+\left[2.25-x_1\cdot\left(1-x_2^2\right)\right]^2+$ $+\left[2.625-x_1\cdot\left(1-x_2^3\right)\right]^2\to\min$	
19	$f(X)=-\left(0.5+0.5\cdot x_1\right)^4\cdot x_2^4\times$ $\times\exp\left[2-\left(0.5+0.5\cdot x_1\right)^4-x_2^4\right]\to\min$	
20	$f(X)=-\left(0.3+0.4\cdot x_1+0.3\cdot x_2\right)^4-\left(0.8-0.6\cdot x_1+0.8\cdot x_2\right)\times$ $\times\exp\left[2-\left(0.3+0.4\cdot x_1+0.3\cdot x_2\right)^4-\left(0.8-0.6\cdot x_1+0.8\cdot x_2\right)^4\right]$	

2. ГЕНЕТИЧНИЙ АЛГОРИТМ

Алгоритм пошуку глобального розв'язку задачі оптимізації можна подати у вигляді ітеративного процесу, який породжує послідовність точок відповідно до запропонованого набору правил, включаючи критерій закінчення рахунку. Пошук глобального розв'язку задач оптимізації відбувається перебором локальних розв’язків. Загалом не можна гарантувати точний розв’язок задачі глобальної оптимізації для багатоекстремальної функції за кінцеву кількість кроків. Для доказу того, що знайдений розв’язок є глобальним оптимумом, необхідно виконати повний перебір всіх можливих значень вектору параметрів. В більшості випадків це неможливо, тому при глобальній оптимізації мова йде зазвичай не про пошук оптимального, а про пошук близького до нього, тобто субоптимального розв’язку. Відомими є детерміновані та стохастичні методи глобальної оптимізації. Перевага стохастичних методів викликана їх універсальністю, що пояснюється оцінкою значень цільової функції у випадкових точках допустимої множини з подальшим аналізом результатів у пробних точках простору пошуку. Генетичний алгоритм (ГА) відноситься до класу стохастичних алгоритмів глобальної оптимізації. У ГА відсутні обмеження на властивості цільової функції у вигляді неперервності, диференційованості тощо.

Біонічні передумови алгоритму. В основі алгоритму лежить механізм природного відбору. Природний відбір – це процес виживання та відтворення організмів, найбільш пристосованих до умов середовища, та загибелі під час еволюції непристосованих. Для обґрунтування внутрішніх механізмів еволюції може бути використана генетика. Як одиниця спадкової інформації використовується ген. Гени утворюють хромосоми. У кожній клітині організму є набір хромосом, що несуть інформацію про всю особину. Хромосоми еволюціонують, ознаки особини поточного покоління формуються внаслідок схрещування батьків із попереднього покоління. Слабкі особини, які не мають якостей, що сприяють виживанню, з великою ймовірністю не проживуть довго і не зможуть створити багаточисельне потомство. Природний відбір, схрещування та мутація забезпечують розвиток популяції. Кожне нове покоління у середньому більш пристосовано, ніж попереднє, тобто воно краще задовольняє вимогам довкілля. Процес удосконалення популяції називається еволюцією.

Введемо до розгляду основні терміни, запозичені з біології та генетики та використовувані при описі еволюційних алгоритмів:

популяція - сукупність особин певного виду. Кожна популяція еволюціонує самостійно, популяції змінюють один одного;

покоління - чергова популяція;

особина - окремий елемент популяції, кожна особина популяції є носієм унікального набору ознак;

фенотип - сукупність ознак організму;

хромосома – носій ознак організму;

ген - частина хромосоми, що відповідає за конкретну ознаку;

геном - сукупність генів організму;

репродукція - процес виникнення нових хромосом, що включає схрещування та мутації;

схрещування - створення двох нових хромосом-нащадків з двох хромосом-пращурів;

мутація - довільна зміна окремих генів організму.

Особливо слід виділити таке поняття як ***фітнес-функція*** або функція відносної придатності, що характеризує процес виживання та поширення найбільш пристосованих організмів. У ГА фітнес-функція характеризує якість розв'язку.

Алгоритм роботи ГА. Схема роботи будь-якого генетичного алгоритму схематично виглядає так:

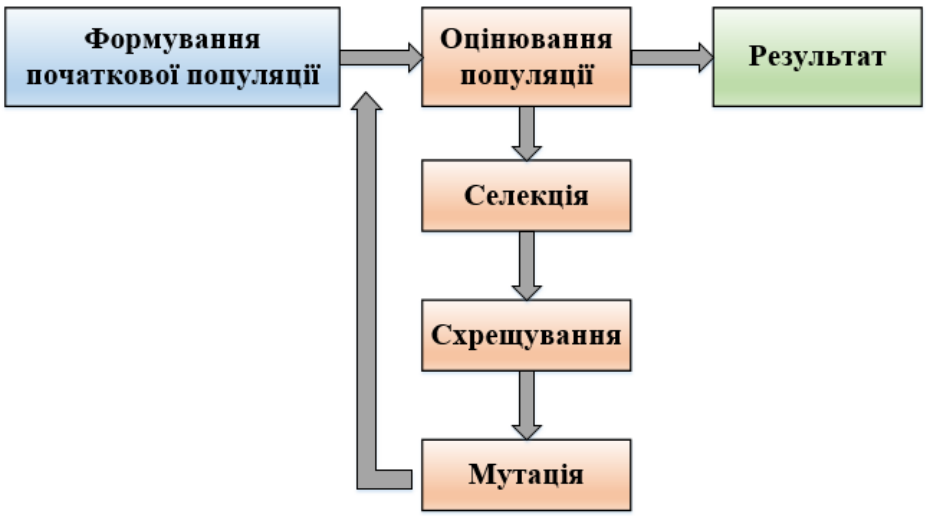


Рис.2.6 - Схема роботи ГА

ГА працює з популяцією особин, у хромосомі (генотип) кожної з яких закодовано можливий розв'язок задачі (фенотип). На початку роботи алгоритму популяція формується випадково. Для того щоб оцінити якість закодованих розв'язків використовують фітнес-функцію, яка необхідна для обчислення пристосованості кожної особини. За результатами оцінювання особин найбільш пристосовані із них вибираються за допомогою виконання операції "селекція" для "схрещування" (рекомбінації). В результаті схрещування обраних особин за допомогою застосування генетичного оператора "кросинговера" ("кросовера") створюється потомство, генетична інформація якого формується в результаті обміну хромосомною інформацією між батьківськими особинами. Створені нащадки формують нову популяцію, причому частина нащадків мутує з використанням генетичного оператора "мутація", що виражається у випадковій зміні їх генотипів. Сукупність етапів, що включає послідовність "Оцінювання популяції" - "Селекція" - "Схрещування" - "Мутація", називається поколінням. Еволюція популяції складається із послідовності зміни таких поколінь.

Селекція особин популяції. Селекція (відбір) необхідна, щоб вибрати більш пристосованих особин для схрещування. Проміжна популяція - це набір особин, які отримали право розмножуватися. У класичному ГА ймовірність кожної особини потрапити до проміжної популяції пропорційна її пристосованості, тобто працює пропорційний відбір. Існує множина варіантів селекції, найбільш відомими з яких є рулеткова селекція та турнірний відбір. У варіанті рулеткової селекції ймовірність *i*-ї особи взяти участь у схрещуванні пропорційна значенню її пристосованості. Процес відбору особин для схрещування асоціюється з грою в "рулетку". Рулетковий круг ділиться на сектори, причому площа *i*-го сектора пропорційна значенню ймовірності відбору. У разі використання турнірного відбору для схрещування відбираються *n* особин. Для цього з популяції випадково вибираються *t* особин, і найпристосованіша із них допускається до схрещування. Вважають, що формується турнір із *t* особин, *t* – розмір турніру. Ця операція повторюється *n* разів. Чим більше значення *t*, тим більша значущість селекції. Варіант турнірного відбору, коли $t = 2$ називають бінарним турніром. Типові значення розміру турніру $t = 2, 3, 4, 5$.

Схрещування особин. Відібрані в результаті селекції особини, які називають батьківськими, схрещуються і дають потомство. Хромосоми нащадків формуються із застосуванням оператора кросинговера у процесі обміну генетичною інформацією між батьківськими особинами. Створені в такий спосіб нащадки становлять популяцію наступного покоління. Виконання операції кросинговера залежить від способу кодування, що застосовується в ГА. Вибір способу кодування є одним із найважливіших етапів під час використання еволюційних алгоритмів. Зокрема, повинна існувати можливість закодувати з допустимою похибкою в хромосомі будь-яку точку із розглядуваної області простору пошуку. Як правило, у хромосомі кодуються чисельні параметри розв'язку. Для цього можливе використання цілочисельного та дійснозначного кодування.

У класичному ГА при **цілочисельному** кодуванні хромосома це бітовий рядок, в якому закодовані параметри розв'язку поставленої задачі. На рис.2.7 показаний приклад кодування чотирьох 10-розрядних параметрів у 40-розрядній хромосомі.

Рис.2.7 – Приклад цілочисельного кодування

Хромосома на рис.2.7 складається з чотирьох 10-розрядних генів. Незважаючи на те, що кожен параметр закодований у хромосомі цілим числом у

вигляді двійкової послідовності, йому можуть бути поставлені у відповідність і дійснозначні числа.

При **дійснозначному** кодуванні в гені кодується не ціле число, а дійснозначне. Це дозволяє позбавитися від операцій кодування-декодування, що використовуються в цілочисельному кодуванні, а також збільшити точність знайденого розв'язку. Приклад дійснозначного кодування представлений на рис.2.8.

Рис.2.8 – Приклад дійснозначного кодування

Розглянемо оператори кросинговера для цілочисельного та дійснозначного кодування. Для цілочисельного кодування часто використовуються 1-но точковий, 2-во точковий і однорідний оператори кросинговера.

При 1-но точковому кросинговері вибирається довільна точка розриву і для створення нащадків проводиться обмін частинами батьківських хромосом. Ілюстративний приклад роботи 1-но точкового кросинговера представлений на рис.2.9 а.

Рис.2.9 – Приклади роботи: а) 1-но точкового оператора кросинговера; б) 2-во точкового оператора кросинговеру

Для оператора 2-во точкового кросинговера вибираються дві випадкові точки розриву, після чого для створення нащадків батьківські хромосоми обмінюються ділянками, що лежать між точками розриву (рис.2.9 б).

З використанням однорідного оператора кросинговера розряди батьківських хромосом успадковуються незалежно один від одного. Для цього визначають

ймовірність p, що i-й розряд хромосоми 1-го з батьків потрапить до першого нащадку, а 2-го з батьків – до другого нащадку. Імовірність протилежної події дорівнює $(1 - p)$. Кожен розряд батьківських хромосом «розігрується» відповідно до значення p між хромосомами нащадків. У більшості випадків ймовірність обох подій однакова, тобто $p = 0.5$.

Для дійснозначного кодування розглянемо 2-во точковий оператор кросинговера. Двоточковий кросинговер для дійснозначного кодування подібний до 2-во точкового кросинговеру для цілочисельного кодування. Відмінність у тому, що точка розриву може бути обрана «всередині» гена, а повинна потрапити між генами (рис.2.10).

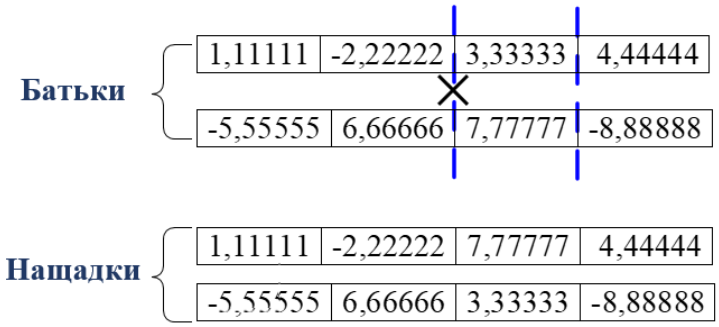


Рис.2.10 – Приклад роботи 2-во точкового кросинговера для дійснозначного кодування

Оператори кросинговера характеризуються здатністю до руйнування батьківських хромосом, але в той же час кросинговер може створити абсолютно нові хромосоми, що не зустрічалися раніше в процесі еволюційного пошуку.

Формування нового покоління. Результатом схрещування є нащадки, що формують популяцію наступного покоління. Оновлена популяція необов'язково повинна включати лише особин-нащадків. Позначимо частку оновлюваних особин як T, причому $0 < T < 1$. Тоді в нове покоління потрапить $T \times n$ нащадків, n – розмір популяції, а $(1 - T) \times n$ особин у новій популяції є найбільш пристосованими батьківськими особинами, які називають елітними. Параметр T називається розривом поколінь. Використання елітних особин дозволяє збільшити швидкість збіжності ГА.

Мутація особин. Оператор мутації використовується для внесення випадкових змін до хромосоми особин. Це дозволяє виходити з локальних екстремумів та детальніше дослідити простір пошуку. Аналогічно оператору кросинговера робота оператора мутації залежить від ймовірності застосування оператора мутації.

Операція мутації полягає у випадковому виборі однієї хромосоми популяції та випадковій зміні одного з її генів. У випадку, якщо гени хромосом приймають значення на множині дійсних чисел, операцію мутації для двох змінних описує

рис.2.11. Максимальна величина кроку мутації залежить від розміру області визначення відповідного параметру.

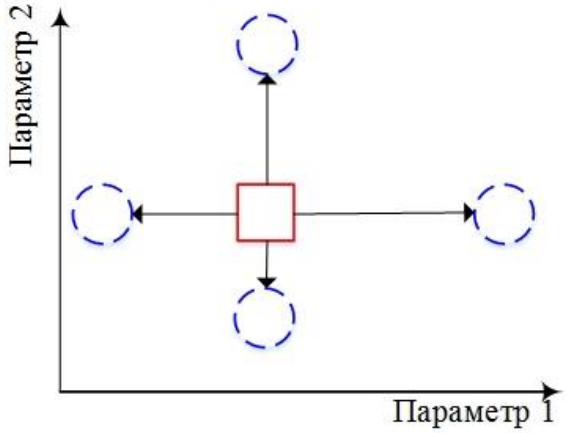


Рис.2.11 - Виконання мутації: ◌ - можливі нащадки; □ - батько

Оператор мутації для дійснозначного кодування змінює вміст кожного гена із заданою ймовірністю. При цьому величина зміни вибирається випадково в деякому діапазоні [-ξ; +ξ], наприклад, [-0,15; 0,15], і може мати як рівномірний, так і будь-який інший розподіл, наприклад, нормальний з m = 0, σ = 0,5.

При цілочисельному кодуванні хромосом оператор мутації змінює деяку позицію хромосоми випадково, тобто відбувається бітова мутація. У результаті мутація змінює окремі розряди у хромосомі. Для цього кожен розряд інвертується з деякою апріорі заданою ймовірністю. Імовірність виконання мутації досить мала, наприклад одна мутація гена на сотню хромосом. Застосування оператора мутації розігрується стільки разів, скільки розрядів міститься у хромосомі. Невелике значення ймовірності мутації дозволяє не сильно руйнувати знайдені хороші хромосоми.

Критерії зупинки. Процес еволюції, загалом, може тривати нескінченно. Як критерій зупинки може бути задана кількість поколінь або збіжність популяції. Збіжністю називається такий стан популяції, коли всі особини популяції перебувають у області деякого екстремуму і майже однакові, тобто кросовер практично ніяк не змінює популяції, а особи, що мутують схильні вимирати, як найменш пристосовані. Отже, збіжність популяції означає, що досягнутий розв’язок близький до оптимального.

Оцінювання популяції. Оцінювання популяції необхідно для того, щоб виявити у ній більш пристосовані і менш пристосовані особини. Для підрахунку пристосованості кожної особини використовується фітнес-функція. У випадку використання цілочисельного кодування для обчислення значення фітнес-функції часто буває необхідно перетворити закодовані в хромосомі цілочисельні значення до дійснозначних чисел. Як правило, використання еволюційного алгоритму має на увазі розв’язок задачі максимізації (мінімізації) цільової функції, коли необхідно знайти такі значення параметрів функції, при яких значення фітнес-функції максимальне (мінімальне). Відповідно з цим, якщо

розв’язується задача мінімізації, то вважають, що *i*-та особина краща (більш пристосована) *j*-ої особини, якщо значення фітнес-функції для неї менше. У разі задачі максимізації, навпаки, *i*-та особина вважається більш пристосованою, ніж *j*-та особина, якщо значення фітнес-функції для неї більше.

Налаштування параметрів ГА. Ефективність роботи ГА істотно залежить від того, яким чином налаштовані його параметри. Основними параметрами ГА є кількість поколінь, розмір популяції, тип оператора кросинговера, ймовірність кросинговера, тип оператора мутації, ймовірність мутації, величина розриву поколінь. Ці параметри впливають на швидкість еволюційного пошуку оптимуму, оскільки визначають здібності ГА до ефективного пошуку розв’язку та характеризують можливості алгоритму уникати локальних екстремумів. Крім того, їх правильний вибір є суттєвим для поступового поліпшення наявних результатів від покоління до покоління на основі вже знайдених "проміжних" розв’язів. Отже, стає можливим передчасна збіжність ГА, тобто виродження популяції, коли розв’язок ще не знайдено, але у популяції майже всі особини стають однаковими і протягом кількох десятків поколінь не спостерігається покращення функції пристосованості. Тому досягнення балансу між дослідженням простору пошуку та ефективним використанням знайдених розв’язів є основною метою налаштування параметрів ГА. На рис.2.12 схематично зображено вплив зміни деяких параметрів ГА на характеристики еволюційного пошуку.

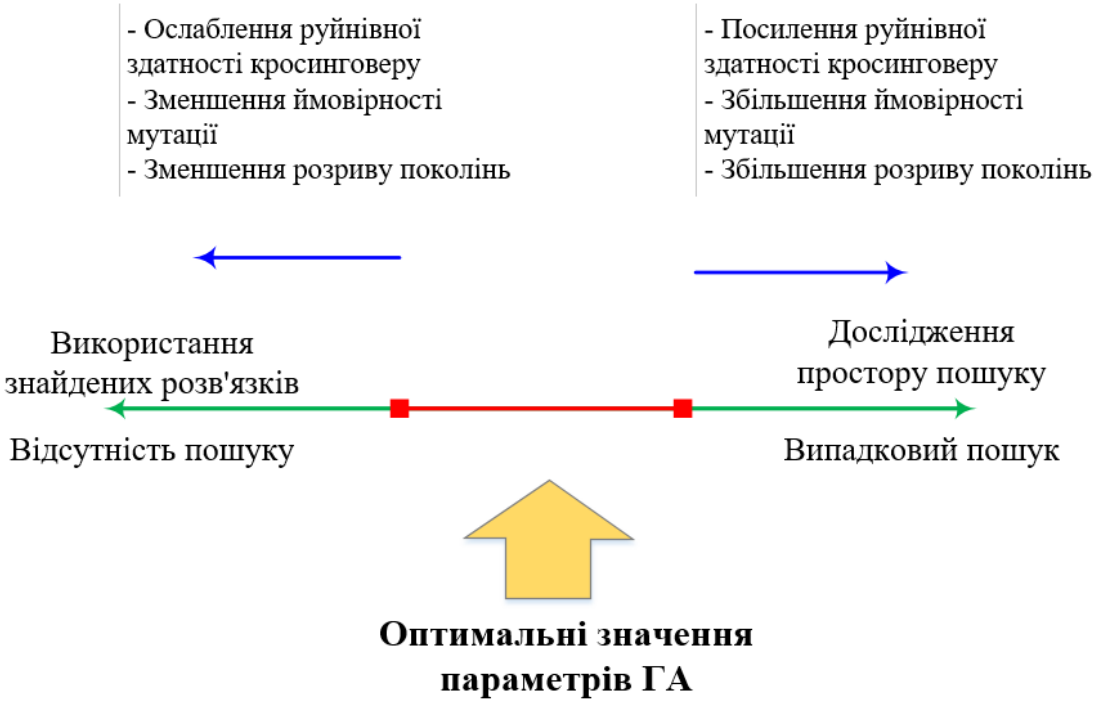


Рис.2.12 - Вплив параметрів ГА на характеристики еволюційного пошуку

Приклад 4. Задана цільова функція

$$f(x,y) = 837.9657 - x \cdot \sin\left(\sqrt{|x|}\right) - y \cdot \sin\left(\sqrt{|y|}\right)$$

Знайти екстремум цільової функції, використовуючи дійснозначний - кодований генетичний алгоритм.

Порядок виконання завдання:

1. Задати цільову функцію.
2. Графічно дослідити поведінку багатоекстремальної функції з використанням вбудованої функції MathCAD ***CreateMesh (Створити сітку)***.
3. Перетворити цільову функцію на вигляд з одним векторним аргументом.
4. Задати параметри генетичного алгоритму: число змінних цільової функцій, розмір популяції особин, ймовірність виконання кросовера, ймовірність мутації, максимальне число поколінь, нижні та верхні границі зміни аргументів функції.
5. Задати зовнішні допоміжні функції.
6. Задати функції турнірного відбору особин ***Tournament()***, неоднорідної мутації ***Mutate()***, повного лінійного кросовера ***Crossover()***, генетичної еволюції особин ***GeneticAlg()***.
7. Отримати значення цільової функції у точці екстремуму.

Приклад виконання завдання у MathCAD

Приклад 4. Дана цільова функція

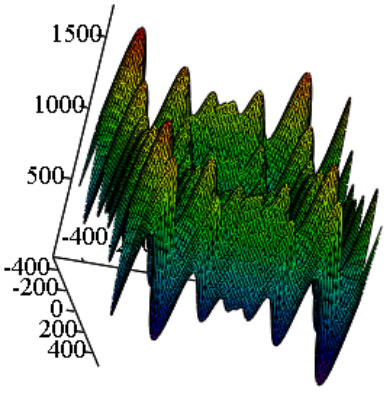
$$f(x,y) = 837.9657 - x \cdot \sin\left(\sqrt{|x|}\right) - y \cdot \sin\left(\sqrt{|y|}\right)$$

Знайти екстремум цільової функції, використовуючи дійснозначний кодований генетичний алгоритм.

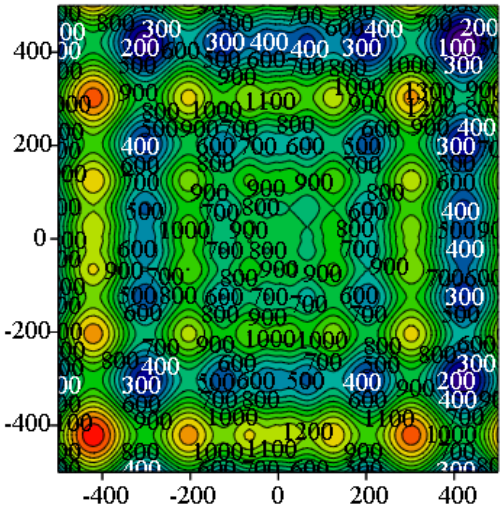
ДІЙСНОЗНАЧНИЙ КОДОВАНИЙ ГЕНЕТИЧНИЙ АЛГОРИТМ

$f(X,Y) := 837.9657 - X \cdot \sin\left(\sqrt{|X|}\right) - Y \cdot \sin\left(\sqrt{|Y|}\right)$ - цільова функція

$Z := \text{CreateMesh}(f, -500, 500, -500, 500, 100, 100)$



Z



Z

- графічний аналіз багатоекстремальної цільової функції

$$FUN(x) := 837.9657 - x_0 \cdot \sin\left(\sqrt{|x_0|}\right) - x_1 \cdot \sin\left(\sqrt{|x_1|}\right)$$

- перетворення цільової функції до виду з одним векторним аргументом

ПАРАМЕТРИ ГЕНЕТИЧНОГО АЛГОРИТМУ

$NV := 2$ - число змінних цільової функції

$NP := 120$ - розмір популяції особин

$PC := 0.85$ - ймовірність виконання кросовера

$PM := 0.05$ - ймовірність мутації

$BM := 5$ - параметр операції мутації

$GMAX := 300$ - максимальне число поколінь

$$minL := \begin{pmatrix} -500 \\ -500 \end{pmatrix}$$ - нижні границі зміни аргументів функції

$$maxL := \begin{pmatrix} 500 \\ 500 \end{pmatrix}$$ - верхні границі зміни аргументів функції

$$rval(a, b) := rnd(1) \cdot (b - a) + a$$

$$D(g, y) := y \cdot rnd(1)\left(1 - \frac{g}{GMAX}\right)^{BM}$$ - зовнішні допоміжні функції

$$Check(s, v) := if\left(v > maxL_s \vee v < minL_s, rval\left(minL_s, maxL_s\right), v\right)$$

ФУНКЦІЯ ТУРНІРНОГО ВІТБОРУ ОСОБИН

```
Tournament(M) := | for i ∈ 0 .. NP − 1
                 |   n ← floor(rnd(NP))
                 |   m ← floor(rnd(NP))
                 |   for j ∈ 0 .. NV
                 |     H_{i,j} ← M_{n,j}
                 |     H_{i,j} ← M_{m,j}  if M_{m,NV} < M_{n,NV}
                 | H
```

ФУНКЦІЯ НЕОДНОРІДНОЇ МУТАЦІЇ

```
Mutate(t, M) := | for i ∈ 0 .. NP − 1
                |   if rnd(1) < PM
                |     s ← floor(rnd(NV))
                |     M_{i,s} ← M_{i,s} + D(t, maxL_s − M_{i,s})
                |     M_{i,s} ← M_{i,s} − D(t, M_{i,s} − minL_s)  if floor(rnd(2)) =
```

ФУНКЦІЯ ПОВНОГО ЛІНІЙНОГО КРОСОВЕРУ

```
Crossover(M, F) := | i ← 0
                   | while i < NP − 1
                   |     if rnd(1) < PC
                   |         for j ∈ 0 .. NV − 1
                   |             B0,j ← 0.5·Mi,j + 0.5·Mi+1,j
                   |             B1,j ← Check(j, 1.5·Mi,j − 0.5·Mi+1,j)
                   |             B2,j ← Check(j, −0.5·Mi,j + 1.5·Mi+1,j)
                   |         for k ∈ 0 .. 2
                   |           Bk,NV ← F[(B^T)^⟨k⟩]
                   |         Q ← csort(B, NV)
                   |         for j ∈ 0 .. NV
                   |             Mi,j ← Q0,j
                   |             Mi+1,j ← Q1,j
                   |     i ← i + 2
                   | M
```

ФУНКЦІЯ ГЕНЕТИЧНОЇ ЕВОЛЮЦІЇ ОСОБИН

```
GeneticAlg(F) := | for i ∈ 0 .. NP − 1
                 |   for j ∈ 0 .. NV − 1
                 |     Mi,j ← rval(minLj, maxLj)
                 | Mi,NV ← F[(M^T)^⟨i⟩]
                 | for g ∈ 0 .. GMAX
                 |     M ← Tournament(M)
                 |     M ← Crossover(M, F)
                 |     M ← Mutate(g, M)
                 |     for i ∈ 0 .. NP − 1
                 |       Mi,NV ← F[(M^T)^⟨i⟩]
                 | (csort(M, NV)^T)^⟨0⟩
```

$$\text{GeneticAlg(FUN)} = \begin{pmatrix} 420.969 \\ 420.969 \\ -7.454 \times 10^{-5} \end{pmatrix}$$

РЕЗУЛЬТАТ ВИКОНАННЯ ФУНКЦІЇ ГА:
вектор-стовпчик, NV рядків якого представляють знайдені значення змінних глобального мінімуму цільової функції, а останній рядок - значення цільової функції в точці екстремуму

ЗАВДАННЯ ДО САМОСТІЙНОЇ РОБОТИ

Завдання 2. Мінімізувати функцію на заданому інтервалі. Перевірити отриманий розв'язок візуально, побудувавши графік функції. Оцінити кількість локальних мінімумів функції. Оцінити середню кількість кроків ГА, необхідних для досягнення розв'язку, виконуючи запуск з різними початковими умовами.

№ п/п	Функція	Діапазон
1	$z(x,y)=x\cdot\sin(4\cdot x)+1.1\cdot y\cdot\sin(2\cdot y)$	$0\le x,y\le 10$
2	$z(x,y)=0.5+\dfrac{\sin^2\left(\sqrt{x^2+y^2}\right)-0.5}{1+0.1\cdot\left(x^2+y^2\right)}$	$-5\le x,y\le 5$
3	$z(x,y)=-e^{-0.2\cdot\sqrt{x^2+y^2}+3\cdot(\cos 2x+\sin 2y)}$	$-5\le x,y\le 5$
4	$z(x,y)=-x\cdot\sin\left(\sqrt{\lvert x-(y+9)\rvert}\right)-$ $-(y-9)\cdot\sin\sqrt{\lvert y+0.5\cdot x+9\rvert}$	$-20\le x,y\le 20$
5	$z(x,y)=x^2+y^2$	$-4\le x,y\le 4$
6	$z(x,y)=\dfrac{x^2+y^2}{\sin(x)+\cos(x)}$	$-4\le x,y\le 4$
7	$z(x,y)=e^{-x^2-y^2}$	$-2\le x,y\le 2$
8	$z(x,y)=\sin(x)\cdot e^{-3\cdot y}$	$x\in[0,2\pi],\ y\in[0,1]$
9	$z(x,y)=\sin^2(x)\cdot\ln(y)$	$x\in[0,\pi],\ y\in[-1,1]$

№ п/п	Функція	Діапазон
10	$z(x,y)=\sin^2(x-y)\cdot e^{-\lvert y\rvert}$	$x\in[0,\pi],\ y\in[-1,1]$
11	$z(x,y)=\dfrac{\sin(x\cdot y)}{x}$	$x\in[0,1.5],\ y\in[-\pi,\pi]$
12	$z(x,y)=\dfrac{x^2\cdot y^2+2\cdot x\cdot y-3}{x^2+y^2+1}$	$x\in[-2,2],\ y\in[-1,1]$
13	$z(x,y)=\left(\sin(x^2)+\cos(y^2)\right)^{x\cdot y}$	$x\in[-1,1],\ y\in[-1,1]$
14	$z(x,y)=(1+x\cdot y)\cdot(3-x)\cdot(4-y)$	$x\in[0,3],\ y\in[0,4]$
15	$z(x,y)=(y^2-3)\cdot\sin\dfrac{x}{\lvert y\rvert+1}$	$x\in[-2\pi,2\pi],\ y\in[-3,3]$
16	$z(x,y)=e^{-\lvert x\rvert}\cdot(x^5+y^4)\cdot\sin(x\cdot y)$	$x\in[-2,2],\ y\in[-3,3]$
17	$z(x,y)=\arctan(x+y)\times$ $\times(\arccos(x)+\arcsin(y))$	$x\in[-1,1],\ y\in[-1,1]$
18	$z(x,y)=\sin^2(x)\cdot e^{\lvert y\rvert}$	$x\in[0,\pi],\ y\in[-1,1]$

3. АЛГОРИТМ ІМІТАЦІЇ ВІДПАЛУ

Алгоритм імітації відпалу є метаевристичний алгоритм знаходження глобального мінімуму цільової функції.

> ***Метод відпалу*** (метод випалу, метод симуляції відпалу, метод модельного загартування, simulated annealing) – це техніка оптимізації, яка використовує впорядкований випадковий пошук на основі аналогії з процесом утворення в речовині кристалічної структури з мінімальною енергією при охолодженні.

Фізичні передумови. Термін "відпал" запозичений з фізики конденсованого стану і відображає поведінку металевого тіла при затвердінні в процесі охолодження температури до нуля. У процесі повільного керованого охолодження, що називається відпалом, кристалізація тіла супроводжується зменшенням його енергії. Енергія тіла може бути інтерпретована як один із параметрів в алгоритмі оптимізації.

Опис методу алгоритму. Ідея перенесення механізму відпалу металів на розв’язок оптимізаційної задачі полягає в тому, що процес оптимізації пов'язують із деякою температурою. На кожному кроці пошуку випадково здійснюється мала зміна стану об'єкта і обчислюється зміна ΔE енергії системи. Нова конфігурація системи приймається з ймовірністю 1, якщо $\Delta E \le 0$, і з ймовірністю, що дорівнює $\exp\left(-\Delta E / k \cdot t\right)$, якщо $\Delta E > 0$. Ця процедура переноситься на розв’язок оптимізаційних задач. При цьому стани фізичної системи замінюються зміною критерію якості, а значення $k \cdot t$ замінюється узагальненим поняттям «температура» T, яка може розглядатися як управляючий параметр оптимізаційної процедури.

На початковому етапі температуру приймають високою, а потім її крок за кроком знижують. При кожній температурі виконують серію пробних переборів розв’язів і після кожної перестановки підраховується значення цільової функції. Найкращі розв’язки приймаються з ймовірністю 1, а «погані», для яких значення цільової функції погіршується, приймаються з певною ймовірністю. Такий ймовірнісний механізм дає можливість, приймаючи в якості вихідних деякі «погані» розв’язки, проскакувати через локальні оптимуми і знаходити глобальні.

У рамках алгоритму задаються параметри, назви яких відображають історію виникнення методу. Це $T_н, T_к$ - початкова та кінцева температура, Δt - інтервал зміни температури. Температура T змінюється від $T_н$ до $T_к$ з інтервалом Δt. Початкові значення $T_н$ - висока, $T_к$ - низька, зазвичай $T_к = 0$. При кожному значенні T виконується задана множина ітерацій. На кожній ітерації виконуються дії, подані на рис.3.1.

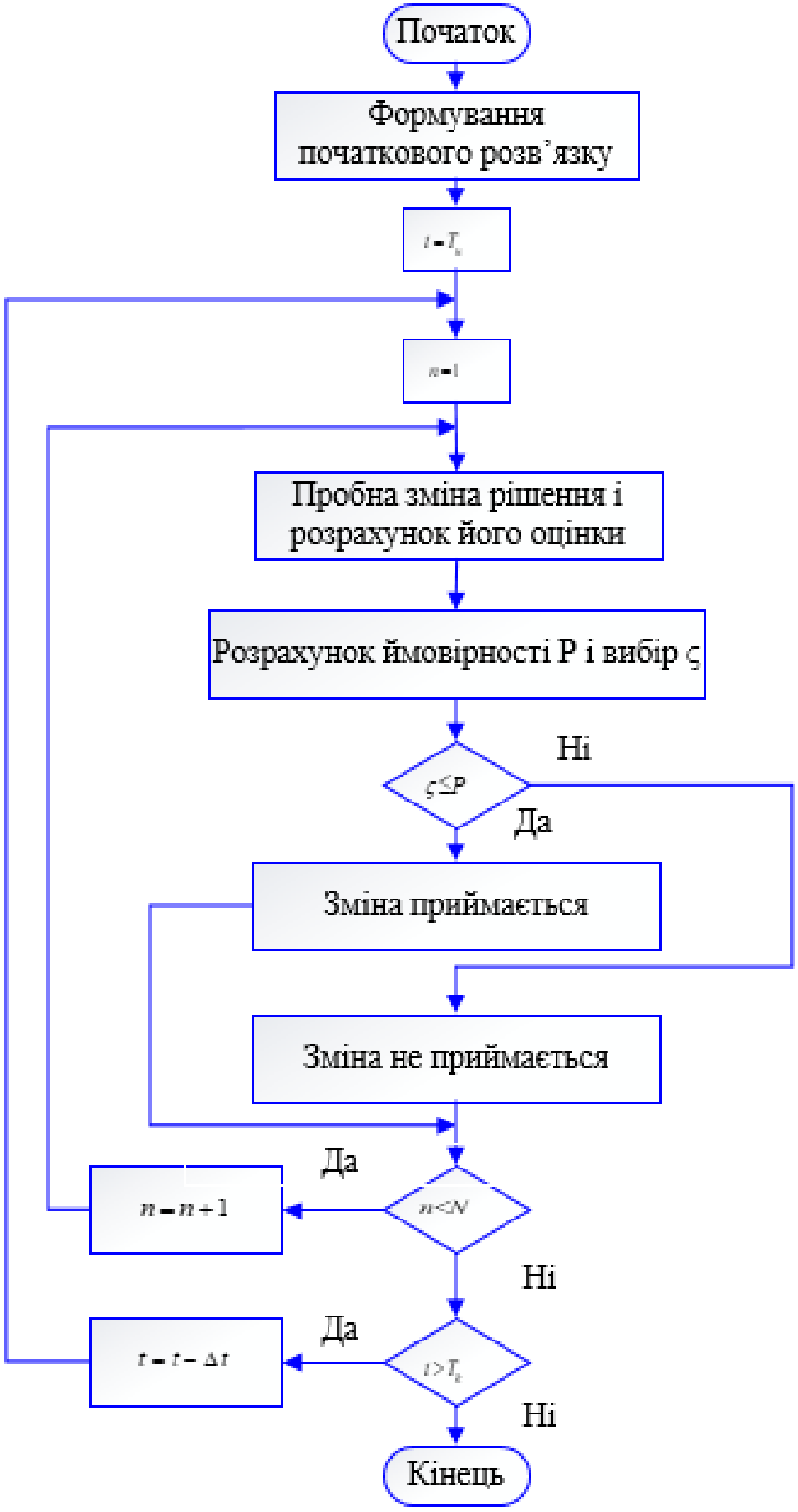


Рис.3.1 - Структура методу моделювання відпалу

За допомогою оператора D здійснюється пробна зміна стану (розв'язку).

Якщо пробна зміна призвела до покращення цільової функції F, то ця зміна фіксується. Якщо пробна зміна призвела до погіршення F на величину ΔF, то розраховується ймовірність збереження зміни за формулою:

$$P = \exp\left(-\frac{\Delta F}{k \cdot T}\right),$$

де k - константа.

Вибирається випадкове число ς із рівномірного розподілу від нуля до одиниці. Якщо $\varsigma \le P$, то зміна зберігається, якщо $\varsigma > P$, то здійснюється повернення до попереднього стану.

Принципи, покладені в основу роботи алгоритму, можна пояснити наступною фізичною аналогією. На рис.3.2 зображено кульку в коробці, внутрішня поверхня якої відповідає ландшафту цільової функції.

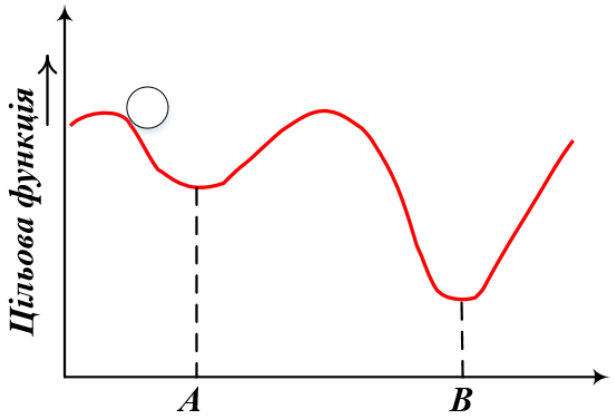


Рис.3.2 – Ілюстрація принципів роботи алгоритмів імітації відпалу

При сильному струшуванні коробки в горизонтальному напрямку кулька може переміститися з будь-якої точки в будь-яку іншу. При поступовому зменшенні сили струшування буде досягнуто умови, коли сила струшування достатня для переміщення кульки з точки A в точку B, але недостатня для того, щоб кулька могла переміститися з B у A. При подальшому зменшенні сили струшування до нуля кулька зупиниться в точці B - точці глобального мінімуму. В алгоритмах імітації відпалу аналогом сили струшування є ймовірність переходу в стан з більш високим значенням цільової функції. На початку роботи алгоритму ця ймовірність повинна бути достатньо великою, щоб була можливість переходу від обраного початкового розв'язку до будь-якого іншого розв'язку. У процесі роботи алгоритму ймовірність переходу зменшується відповідно до обраного закону.

Недоліком методу моделювання відпалу є те, що він не зберігає інформацію про різні дії, виконані на попередніх ітераціях. Особливістю методу і те, що якість одержуваного розв'язку багато в чому залежить від початкового. Чим

кращий початковий розв’язок, тим вищий шанс отримання нового розв’язку з найкращою якістю.

Приклад 5. Знайти глобальний оптимум функції $f(X)=\sin(2\cdot x)+\sin(2\cdot y)+\exp(\alpha\cdot|x|)+\exp(\beta\cdot|y|)$, використовуючи алгоритм імітації відпалу.

АЛГОРИТМ СТОХАСТИЧНОЇ ОПТИМІЗАЦІЇ ІМІТАЦІЄЮ ВІДПАЛУ

Процедура **Prob( )** визначення ймовірності прийнятності розв'язку

$$\text{Prob}(\text{New},\text{Old},\text{Temp}) := \begin{cases} \text{return } 1 \text{ if New} < \text{Old} \\ \text{return } \exp\left[\dfrac{(\text{Old}-\text{New})}{\text{Temp}}\right] \text{ otherwise} \end{cases}$$

$$\text{RandomInts}(N,n) := \text{round}(\text{runif}(N,0,n+1)-0.5)$$

- повертає N випадкових цілих чисел в межах від 0 до n

Функция **Candidate( )**, визначає новий варіант розв'язку на кожній ітерації:
Values1 - вектор із N рядків, що містить поточні значення аргументів;
Values2 - вектор з N рядків, що містить мінімальні значення аргументів;
Values3 - вектор з N рядків, що містить максимальні значення аргументів;
N - кількість аргументів, що змінюються випадково.

$$\text{Candidate}(\text{Values},\text{MinValues},\text{MaxValues},N) :=$$
$$\text{Indicies} \leftarrow \text{RandomInts}(N,\text{last}(\text{Values}))$$
$$\text{for } i \in \text{Indicies}$$
$$\text{Values}_i \leftarrow \text{MinValues}_i + (\text{MaxValues}_i - \text{MinValues}_i)\cdot\text{rnd}(1)$$
$$\text{Values}$$

Функція Anneal( ) штучної імітації відпалу:
Obj_Function - назва цільової функції, що підлягає мінімізації;
MinValues - вектор мінімально допустимих значень для аргументів цільової функції;
MaxValues - вектор максимально допустимих значень для аргументів цільової функції;
Temp - початкова температура;
MinTemp - мінімальна кінцева температура;
CoolRate - швидкість охолодження;
N - число аргументів цільової функції;
Ntries - кількість нових кандидатів розв'язку, що створюються при кожній температурі.
Результат роботи функції: матриця з двох стовпчиків, перший з яких містить поточний розв'язок (температура досягла мінімального кінцевого значення), а другий - найкращий розв'язок.

```
LastSol ← | Rand ← runif(rows(MinValues), 0, 1)
          | t1 ← Rand·[(MaxValues − MinValues)]
          | MinValues + t1
LastCost ← Obj_Function(LastSol)
BestSol ← LastSol
BestCost ← LastCost
while Temp > MinTemp
    for i ∈ 1 .. Ntries
        CurrentSol ← Candidate(LastSol, MinValues, MaxValues, N)
        CurrentCost ← Obj_Function(CurrentSol)
        P ← Prob(CurrentCost, LastCost, Temp)
        if P ≥ rnd(1)
            LastSol ← CurrentSol
            LastCost ← CurrentCost
        if LastCost < BestCost
            BestSol ← LastSol
            BestCost ← LastCost
    Temp ← Temp·(1 − CoolRate)
augment(LastSol, BestSol)
```

$Alfa := 0.03 \qquad Beta := 0.03$

$f(Alfa, Beta, x, y) := \sin(2\cdot x) + \sin(2\cdot y) + \exp(Alfa\cdot |x|) + \exp(Beta\cdot |y|)$

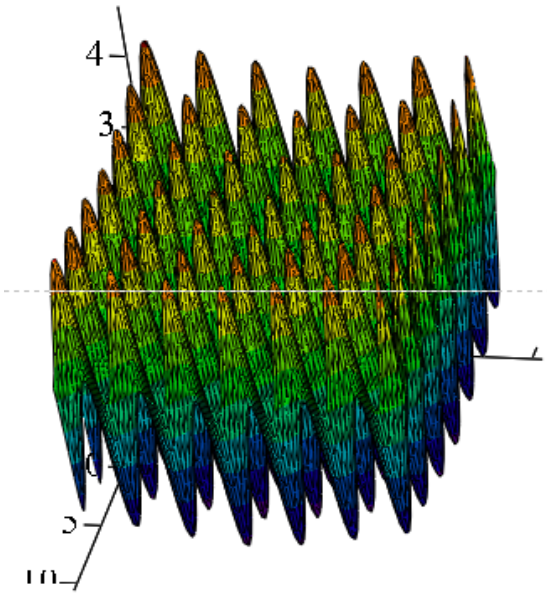
- цільова функція

$$f2(x, y) := \begin{pmatrix} x \\ y \\ f(Alfa, Beta, x, y) \end{pmatrix}$$

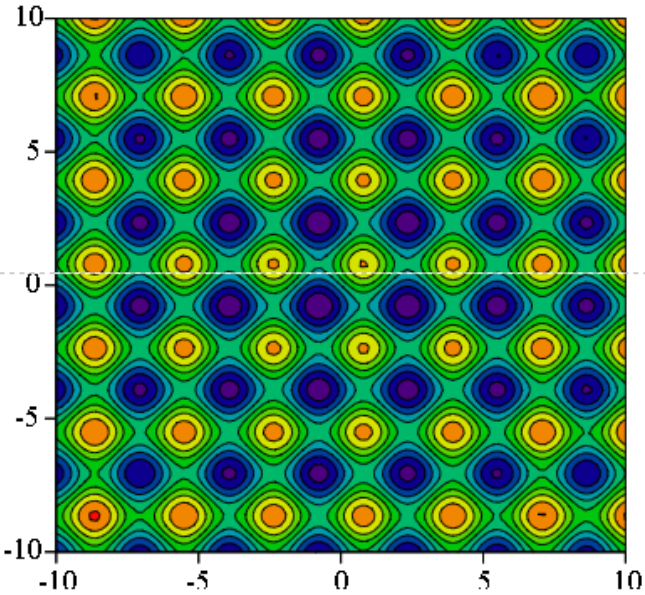
- так як функція **f( )** містить чотири аргументи, то для можливості використання функції **CreateMesh( )**, що допускає тільки два аргументи, виконаємо перетворення функції **f( )** в **f2( )**

$F := CreateMesh(f2, -10, 10, -10, 10, 100, 100)$

- побудова таблиці значень функції



F



F

$f3(p) := f(Alfa, Beta, p_0, p_1)$ - перетворення цільової функції до виду з одним векторним аргументом

$MinVals := \begin{pmatrix} -10 \\ -10 \end{pmatrix}$ $MaxVals := \begin{pmatrix} 10 \\ 10 \end{pmatrix}$

$Temp := 25$ $MinTemp := 0.0001$ - ініціалізація фактичних параметрів функції імітації відпалу

$CoolRate := 0.01$ $N := 2$

$Ntries := 25$

$Solution := Anneal(f3, MinVals, MaxVals, Temp, MinTemp, CoolRate, N, Ntries)$

$Solution = \begin{pmatrix} -0.77965 & -0.77965 \\ -0.78093 & -0.78093 \end{pmatrix}$ - отриманий розв'язок: глобальний мінімум

$Solution^{\langle 0 \rangle} = \begin{pmatrix} -0.77965 \\ -0.78093 \end{pmatrix}$ - отриманий поточний розв'язок

$Solution^{\langle 1 \rangle} = \begin{pmatrix} -0.77965 \\ -0.78093 \end{pmatrix}$ - отриманий найкращий розв'язок

$f3(Solution^{\langle 0 \rangle}) = 0.04748$ - значення цільової функції в точці поточного розв'язку

$f3(Solution^{\langle 1 \rangle}) = 0.04748$ - значення цільової функції в точці найкращого розв'язку

ЗАВДАННЯ ДО САМОСТІЙНОЇ РОБОТИ

Завдання 3. Знайти глобальний оптимум функції за допомогою алгоритму імітації відпалу.

№ п/п	*Функція*
1	$f(X)=\frac{1}{10}\cdot\left[12+x_1^2+\frac{1+x_2^2}{x_1^2}+\frac{x_1^2\cdot x_2^2+100}{(x_1\cdot x_2)^4}\right]$
2	$f(X)=\left[9-(x_1-3)^2\right]\cdot\frac{x_2^3}{27\cdot\sqrt{3}}$
3	$f(X)=C_0+C_1\cdot x_1+C_2\cdot x_2\cdot x_1+C_3\cdot x_3\cdot x_1+$ $+C_4\cdot x_4\cdot x_1+C_5\cdot x_5\cdot x_1$
4	$f(X)=\sum_{i=1}^{10}\left(e^{-\left(\frac{x_1\cdot i}{10}\right)}-e^{-\left(\frac{x_2\cdot i}{10}\right)}-e^{-\left(\frac{i}{10}\right)}+e^{-i}\right)$
5	$f(X)=\sum_{i=1}^{n} i\cdot x_i^2$
6	$f(X)=\sum_{i=1}^{n} i\cdot x_i^2+x_1^4$
7	$f(X)=\sum_{i=1}^{n} i\cdot x_i^2+n\cdot x_1^4$
8	$f(X)=\sum_{i=1}^{n} i\cdot\left(x_i^2+x_i^4\right)$
9	$f(X)=(1-x_1)^2+100\cdot\sum_{i=2}^{n}\left(x_i-x_{i-1}^2\right)$

№ п/п	Функція
10	$f(X)=(1-x_1)^2+$ $+100\cdot\sum_{i=2}^{n}\left(x_i-\sin\left(\frac{\pi\cdot x_1}{2}\right)\cdot\cos^{i-2}\left(\frac{\pi\cdot x_1}{2}\right)\right)^2+$ $+100\cdot\left(x_n-\cos^{n-2}\left(\frac{\pi\cdot x_1}{2}\right)\right)$
11	$f(X)=\left\|x_2-a_1\cdot\left\|x_1\right\|\right\|+a_2\cdot\left\|1-x_1\right\|,\ x\in R^2,\ a_1>0,\quad a_2>0$
12	$f(X)=\left(\cos\left(2\cdot x_1^{\,2}\right)-1.1\right)^2+\left(\sin\left(0.5\cdot x_1\right)-1.2\right)^2-$ $-\left(\cos\left(2\cdot x_2^2\right)-1.1\right)^2+\left(\sin\left(0.5\cdot x_2\right)-1.2\right)^2$
13	$f(X)=3905.93-100\cdot\left(x_1^2-x_2^2\right)^2-(1-x_1)^2\to\max$
14	$f(X)=\sum_{i=1}^{n}i\cdot\left\|x_i\right\|$
15	$f(X)=\sum_{i=1}^{n}\left(E_i-\sum_{j=1}^{n}\left(A_{ij}\cdot\sin\left(x_j\right)+B_{ij}\cdot\cos\left(x_j\right)\right)\right)^2,$ де E_i, A_{ij}, B_{ij} - псевдовипадкові коефіцієнти
16	$f(X)=-x_1^2\cdot\exp\left[1-x_1^2-20.25\cdot\left(x_1-x_2\right)^2\right]$
17	$f(X)=-\left(0.3\cdot x_1^2+0.7\cdot x_2^2\right)\times$ $\times\exp\left[1-0.6\cdot\left(x_1-x_2\right)^2-\left(0.3\cdot x_1^2+0.7\cdot x_2^2\right)^3\right]$

№ п/п	Функція
18	$$f(X)=-\sum_{j=1}^{4}\left(c_j\cdot\exp\left(-\sum_{i=1}^{3}a_{i,j}\cdot\left(x_i-p_{i,j}\right)^2\right)\right)\rightarrow\min$$

j	$a_{1,j}$	$a_{2,j}$	$a_{3,j}$	$p_{1,j}$	$p_{2,j}$	$p_{3,j}$	c_j
1	3.0	10	30	0.36890	0.11700	0.26730	1.0
2	0.1	10	35	0.46990	0.43870	0.74700	1.2
3	3.0	10	30	0.10910	0.87320	0.55470	3.0
4	0.1	10	35	0.03815	0.57430	0.88280	3.2

4. ГІБРИДНИЙ АЛГОРИТМ ОПТИМІЗАЦІЇ НА ОСНОВІ ГЕНЕТИЧНОГО АЛГОРИТМУ З ЛОКАЛЬНИМ ПОШУКОМ МЕТОДОМ НЕЛДЕРА-МІДА

Дослідження останніх років показали, що ГА є найкращими із існуючих методів для розв'язку багатоекстремальних задач оптимізації. Такі задачі успішно розв'язуються за допомогою ГА, але значні витрати часу на їх розв'язок стримують їх застосування у складних задачах, що відрізняються великими обсягами обчислень цільових функцій, заданих аналітично або алгоритмічно.

Важливою особливістю ГА є те, що генетичні оператори кросовера, мутації та інверсії в процесі генерування нащадків не використовують інформації про локальний рельєф поверхні цільової функції і ця властивість робить їх застосування досить перспективним. Генерування нащадків у ГА відбувається випадково, при цьому відсутні гаранті'ї, що знайдені розв'язки будуть кращими за батьківські. Тому в процесі еволюції зустрічаються і невдалі нащадки, які в результаті збільшують кількість звернень до цільової функції, і тим самим збільшують час пошуку глобального екстремуму. Розглянутий гібрид ГА здатний з необхідною точністю здійснити пошук глобального екстремуму з найменшою кількістю звернень до цільової функції.

Цей гібрид ГА зберігає в собі генетичні якості стохастичної селекції популяції пошукових точок, а для виключення невдалих нащадків при їх генеруванні в алгоритмі використовується процедура регулярного пошуку локальних екстремумів із застосуванням алгоритму деформованого багатогранника. Крім того, в процесі регулярного пошуку гібрид ГА здатний знайти глобальний екстремум поза заданим діапазоном варіювання змінних.

Опис гібридного алгоритму. На рис. 4.1, 4.2 наведено дві частини блок-схем роботи гібриду ГА. У першій її частині реалізовані процедури статистичного задання початкової популяції, сортування та відкидання (елімінування) неперспективних особин, ймовірнісної селекції групи особин для початку процесів регулярного пошуку локальних екстремумів, а також процедур завершення роботи алгоритму. У другій частині реалізовано операції регулярного пошуку локальних екстремумів.

Аргументами функції, що реалізує гібрид ГА, є змінні ε і μ, область пошуку D, задана діапазонами варіювання змінних задачі оптимізації. У багатомірному досліджуваному просторі пошуку E^N генерується початкова популяція особин розміром $\mu(\mu > N)$, де N - розмірність задачі. У свою чергу, ε є достатньо малим числом, яке характеризує точність відшукання глобального екстремуму. При зміні поколінь в алгоритмі закладено рекомендовану багатьма дослідниками 10-ти відсоткову заміну неперспективних особин. При еволюції в кожному поколінні видалялися по три найгірші особини, а на заміну їм знаходилися точки в локальних екстремумах.

У першому блоці алгоритму вводиться розмірність задачі оптимізації N, число особин у популяції μ, точність розв'язку задачі ε, а також дві матриці

початкових граничних максимальних і мінімальних значень координат векторів оптимізуючих популяції.

У першому пункті блоку 2 створюється матриця випадкових чисел для координат μ початкових векторів оптимізації Xn:

$$xn_{i,j} = rnd\left(0, xx_{i,j}^{\max} - xn_{i,j}^{\min}\right) + xn_{i,j}^{\min};\ i \in 1\ldots N,\ j = 1\ldots\mu.$$

У другому пункті блоку 2 для матриці координат векторів початкової популяції обчислюються значення цільової функції $F^{\langle i\rangle} = f\left(Xn^{\langle i\rangle}\right), i \in 1\ldots\mu$.

Цільова функція задається в аналітичному вигляді або її значення обчислюються чисельним методом алгоритмічно.

У третьому пункті блоку 2 наводиться об'єднання матриці Xn і вектору F і створюється розширена матриця стану популяції X:

$$X = \begin{vmatrix} x_{1,1} & x_{1,2} & \ldots & x_{1,\mu-1} & x_{1,\mu} \\ x_{2,1} & x_{2,2} & \ldots & x_{2,\mu-1} & x_{2,\mu} \\ \ldots & \ldots & \ldots & \ldots & \ldots \\ x_{N,1} & x_{N,2} & \ldots & x_{N,\mu-1} & x_{N,\mu} \\ x_{N+1,1} & x_{N+1,2} & \ldots & x_{N+1,\mu-1} & x_{N+1,\mu} \end{vmatrix},$$

де елементами останнього рядка N+1 є значення вектору F.

Матриця X у процесі генетичного відбору оновлюється від покоління до покоління та є показником ступеня наближення розв'язку до найкращого чи глобального екстремуму.

У блоці 3 проводиться сортування стовпців матриці X за зростанням елементів N+1 рядку. Після сортування останні стовпці матриці X будуть містити неперспективні особини, які при зміні покоління популяції повинні бути елімінованні, тобто замінені на нові, перспективніші.

У блоці 4 популяція оцінюється на виродженість матриці стану X. Якщо умови блоку 4 виконуються, задача вважається розв'язаною і алгоритм виходить на закінчення розрахунку і в блоці 5 виводиться перший стовпець матриці X.

В іншому випадку в блоці 5 встановлюється початкове значення лічильника поколінь $r = 0$ та алгоритм переходить до блоку 7.

У першому пункті блоку 7 створюється вектор N+1 цілих випадкових неповторюючихся чисел P. Випадкові числа вибираються з інтервалу від 1 до $0{,}9 \cdot \mu$ (за кількістю перспективних особин, що залишилися в популяції):

$$P_j = \left[rnd\left(0, 0.9 \cdot \mu\right)\right],\ j \in 1\ldots N+1,$$

де $\left[\ \right]$ - виділення цілої частини числа.

У другому пункті блоку 7 із матриці стану X вибираються стовпці батьківської групи N+1 особин для r-го регулярного пошуку нащадка. Принцип формування матриці батьківської групи An показаний нижче:

$$An = \begin{vmatrix} x_{1,P_1} & x_{1,P_2} & \dots & x_{1,P_N} & x_{1,P_{N+1}} \\ x_{2,P_1} & x_{2,P_2} & \dots & x_{2,P_N} & x_{2,P_{N+1}} \\ \dots & \dots & \dots & \dots & \dots \\ x_{N,P_1} & x_{N,P_2} & \dots & x_{N,P_N} & x_{N,P_{N+1}} \end{vmatrix}$$

У третьому пункті блоку 7 обчислюються N+1 елементів вектору значень цільової функції Fr для стовпців матриці An, а в четвертому створюється матриця стану батьківської групи особин r-го регулярного пошуку локального екстремуму. Елементами N+1 рядка матриці є вектор значень цільової функції Fr:

$$A = \begin{vmatrix} a_{1,1} & a_{1,2} & \dots & a_{1,N} & a_{1,N+1} \\ a_{2,1} & a_{2,2} & \dots & a_{2,N} & a_{2,N+1} \\ \dots & \dots & \dots & \dots & \dots \\ a_{N,1} & a_{N,2} & \dots & a_{N,N} & a_{N,N+1} \\ a_{N+1,1} & a_{N+1,2} & \dots & a_{N+1,N} & a_{N+1,N+1} \end{vmatrix}$$

У блоці 8 проводиться сортування стовпців матриці за зростанням N+1 рядку, а в блоці 9 оцінюється ступінь стиснення стовпців матриці A. Оцінка проводиться по елементам N+1 рядку. Якщо умова блоку 9 виконується, то вважається, що локальний екстремум знайдено і в блоці 10 перший стовпець матриці записується A в $\left[0.9 \cdot m\right] + r$ - ий стовпець матриці X і встановлюється нове значення лічильника поколінь $r = r + 1$. Потім у блоці 11 перевіряється кількість знайдених локальних екстремумів r.

Якщо виконується умова:

$$r < \left[0.1 \cdot \mu\right],$$

тоді алгоритм повертається до блоку 3. Інакше – до блоку 7.

Якщо умова не виконується, то алгоритм переходить до другої частини блок-схеми до блоків регулярного пошуку локального екстремуму.

У блоці 12 обчислюються координати центру тяжіння матриці A. Центр тяжіння обчислюється без N+1 – го найгіршого стовпця. Там же, у другому пункті виконується операція «відображення» від найгіршого стовпця і обчислюється значення цільової функції для відображеного N+2 – стовпця.

Якщо значення цільової функції у відображеній точці $a_{N+1,N+3}$ менше чи рівне, ніж найкраще $a_{N+1,1}$, то в блоці 14 виконується операція «розтягування» матриці у перспективному напрямку. Для нового $a_{N+1,N+4}$ стовпця обчислюється цільова функція і порівнюється з найкращим значенням $a_{N+1,1}$. Якщо $a_{N+1,N+4} < a_{N+1,1}$, то стовпець матриці $A^{\langle N+4\rangle}$ записується на місце найгіршого стовпця $A^{\langle N+1\rangle}$ і алгоритм повертається до блоку 8 для нового сортування стовпців матриці A. В іншому випадку в стовпець $A^{\langle N+1\rangle}$ записуються елементи стовпця $A^{\langle N+3\rangle}$ та алгоритм також переходить до блоку 8.

Якщо у блоці 13 $a_{N+1,N+3} > a_{N+1,1}$, то в блоці 18 елемент $a_{N+1,N+3}$ порівнюється з елементом $a_{N+1,N}$, другий після найгіршого стовпця. Якщо $a_{N+1,N+3} < a_{N+1,N}$, то в стовпець $A^{\langle N+1\rangle}$ записуються елементи стовпця $A^{\langle N+3\rangle}$ і алгоритм повертається до блоку 8. В іншому випадку цільова функція в $N+3$ стовпці порівнюється з найгіршим $N+1$ значенням.

Якщо $a_{N+1,N+3} < a_{N+1,N+1}$, то в стовпець $A^{\langle N+1\rangle}$ записуються елементи стовпця $A^{\langle N+3\rangle}$ та алгоритм переходить до блоку 20. В іншому випадку алгоритм переходить до блоку 20 без передачі елементів стовпця.

У блоці 20 виконується операція «стиснення» матриці A в просторі між центром тяжіння і найгіршим стовпцем і, відповідно, обчислення цільової функції в новому $N+5$ стовпці. Якщо $a_{N+1,N+5} < a_{N+1,N+1}$, то в блоці 22 у стовпець $A^{\langle N+1\rangle}$ записуються елементи стовпця $A^{\langle N+5\rangle}$ і алгоритм повертається до блоку 8. В іншому випадку в блоці 23 відбувається «редукція» матриці A. Редукція – це зменшення удвічі відстані всіх стовпців від найкращого стовпця A^1.

Для нових значень елементів перших N рядків всіх стовпців матриці A обчислюються та записуються в $N+1$ рядок значення цільової функції. Потім алгоритм повертається до 8 блоку.

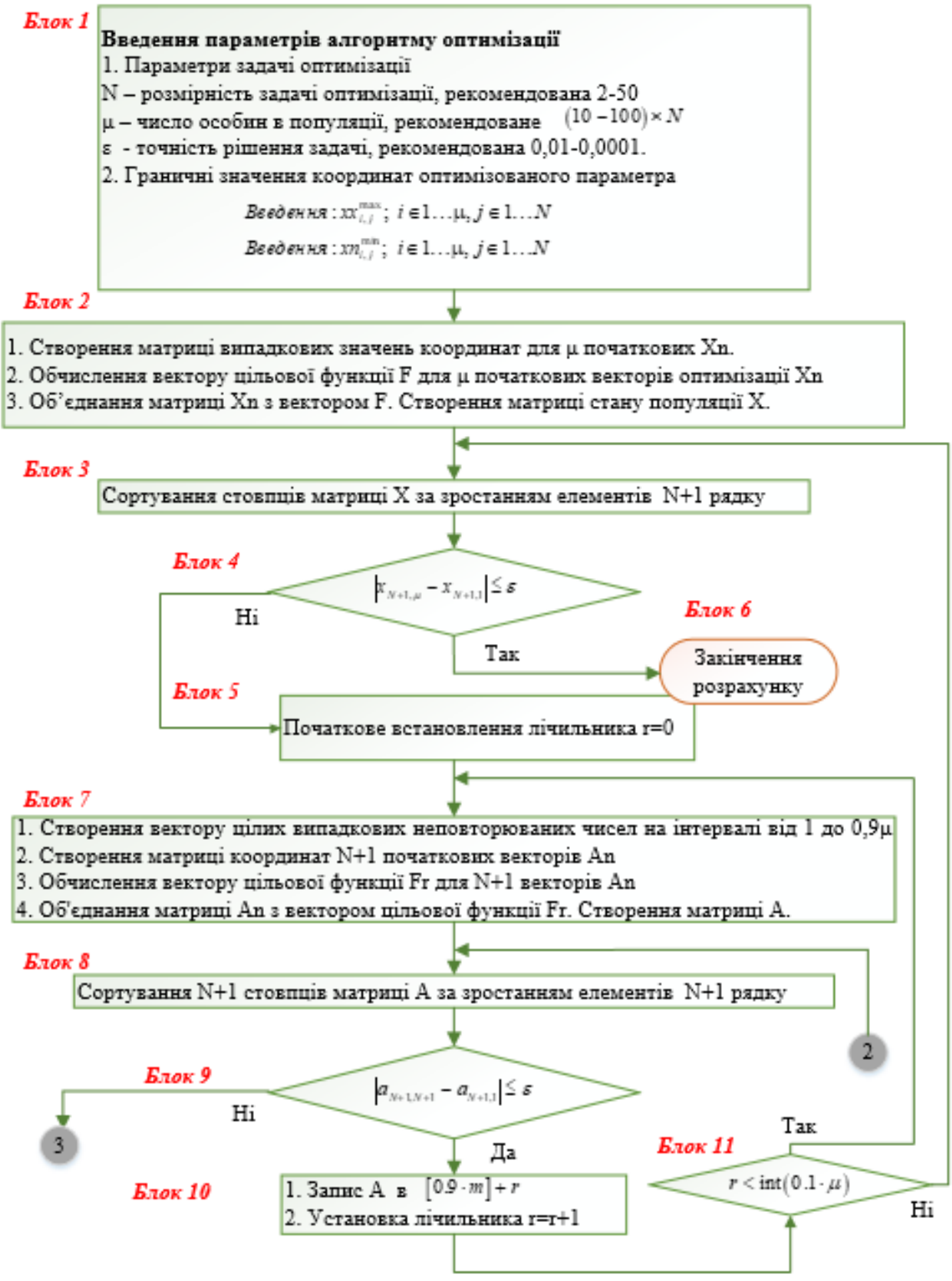


Рис.4.15 -Блок-схема гібрида ГА (частина 1)

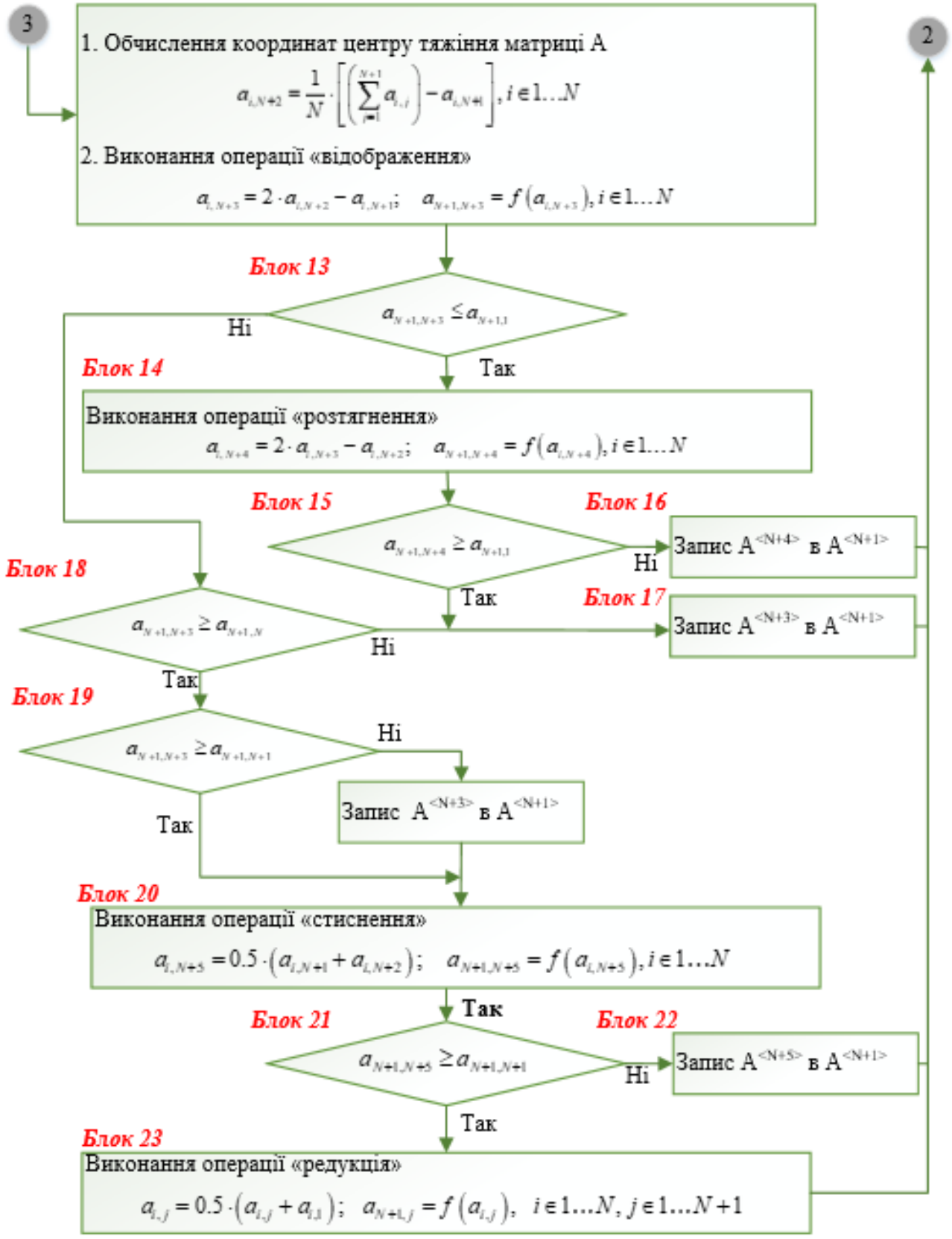


Рис.4.16 -Блок-схема гібрида ГА (частина 2)

Приклад 6. Задана цільова функція

$$f\left(x_1, x_2\right) = \left(\sin\left(\pi \cdot x_1\right) + \sin\left(\pi \cdot x_2\right)\right) \cdot 0.2 +$$
$$+0.01 \cdot \left[0.4 \cdot \left(x_1 - 5.5\right)^2 + 0.5 \cdot \left(x_2 - 5.5\right)^2\right] + 0.4$$

Знайти екстремум цільової функції, використовуючи генетичний алгоритм із локальним пошуком методом Нелдера-Міда.

Порядок виконання завдання:

1. Задати функцію MGA(μ, xx, xn, ε, Sqr), що визначає екстремум цільової функції.
2. Задати цільову функцію.
3. Графічно дослідити поведінку багатоекстремальної функції з використанням вбудованої функції MathCAD ***CreateMesh (Створити сітку)***.
4. При необхідності перетворити цільову функцію до вигляду з одним векторним аргументом.
5. Задати параметри генетичного алгоритму: розмір популяції особин, задану точність, нижні та верхні границі зміни аргументів функції.
6. Визначити координати точки екстремуму та отримати значення цільової функції.
7. Перевірити знайдений розв'язок за допомогою функції ***Minimize(),*** задаючи початкові значення, близькі до знайдених на попередньому етапі.

Приклад виконання завдання у MathCAD

Приклад 6. Дана цільова функція

$$f\left(x_1, x_2\right) = \left(\sin\left(\pi \cdot x_1\right) + \sin\left(\pi \cdot x_2\right)\right) \cdot 0.2 +$$
$$+0.01 \cdot \left[0.4 \cdot \left(x_1 - 5.5\right)^2 + 0.5 \cdot \left(x_2 - 5.5\right)^2\right] + 0.4$$

Знайти екстремум цільової функції, використовуючи алгоритм з локальним пошуком методом Нелдера-Міда.

ГІБРИД ГЕНЕТИЧНОГО АЛГОРИТМУ З ЛОКАЛЬНИМ ПОШУКОМ МЕТОДОМ НЕЛДЕРА - МІДА

Функція MGA(μ, xx, xn, ε, Sqr);

μ - розмір популяції особин, μ > N, де N - розмірність оптимізаційної цільової функції;

xx,xn - вектори нижніх та верхніх границь зміни аргументів цільової функції;

ε - задана точність розв'язку задачі;

Sqr - назва функції, яка обчислює значення цільової функції.

Результат роботи функції - вектор, перші N елементів якого це значення шуканих змінних, а останній - значення цільової функції в точці экстремуму.

```
MGA(μ, xx, xn, ε, Sqr) :=
    N ← length(xx)
    m ← floor( [[(μ + N) + 1]] / 0.9 )
    for j ∈ 0 .. m
        for i ∈ 0 .. N − 1
            x_{i,j} ← rnd(xx_i − xn_i) + xn_i
        x_{N,j} ← Sqr(x^⟨j⟩)
    z ← 0
    x ← rsort(x, N)
    while |x_{N,m} − x_{N,0}| > ε
        z ← z + 1
        r ← 0
        while r ≤ floor(0.1·m)
            r ← r + 1
            h ← 0
            while h = 0
                Mt ← 0
                for j ∈ 0 .. N
                    P_j ← floor(rnd(0.9·m))
                for j ∈ 1 .. N
                    for jj ∈ 0 .. j − 1
                        Mt ← 1 if P_jj = P_j
                h ← 1 if Mt = 0
            for j ∈ 0 .. N
                as^⟨j⟩ ← x^⟨(P_j)⟩
            k ← 0
            as ← rsort(as, N)
            while |as_{N,N} − as_{N,0}| > ε
                a ← as
                for i ∈ 0 .. N − 1
```

$$a_{i,N+1} \leftarrow \frac{1}{N}\cdot\left[\left(\sum_{j=0}^{N} a_{i,j}\right) - a_{i,N}\right]$$

$$a_{i,N+2} \leftarrow a_{i,N+1} + 1.0\cdot\left(a_{i,N+1} - a_{i,N}\right)$$

$$a_{N,N+2} \leftarrow \mathrm{Sqr}\left(a^{\langle N+2\rangle}\right)$$

if $a_{N,N+2} \le a_{N,0}$

for $i \in 0..N-1$

$$a_{i,N+3} \leftarrow a_{i,N+1} + 2\left(a_{i,N+2} - a_{i,N+1}\right)$$

$$a_{N,N+3} \leftarrow \mathrm{Sqr}\left(a^{\langle N+3\rangle}\right)$$

$$a^{\langle N\rangle} \leftarrow a^{\langle N+3\rangle} \quad \text{if } a_{N,N+3} \le a_{N,0}$$

$$a^{\langle N\rangle} \leftarrow a^{\langle N+2\rangle} \quad \text{otherwise}$$

otherwise

$$a^{\langle N\rangle} \leftarrow a^{\langle N+2\rangle} \quad \text{if } a_{N,N+2} < a_{N,N-1}$$

$$a^{\langle N\rangle} \leftarrow a^{\langle N+2\rangle} \quad \text{if } a_{N,N+2} < a_{N,N}$$

otherwise

for $i \in 0..N-1$

$$a_{i,N+4} \leftarrow a_{i,N+1} + 0.5\left(a_{i,N} - a_{i,N+1}\right)$$

$$a_{N,N+4} \leftarrow \mathrm{Sqr}\left(a^{\langle N+4\rangle}\right)$$

for $j \in 0..N$ if $a_{N,N+4} \ge a_{N,N}$

for $i \in 0..N-1$

$$a_{i,j} \leftarrow a_{i,0} + 0.5\left(a_{i,j} - a_{i,0}\right)$$

$$a_{N,j} \leftarrow \mathrm{Sqr}\left(a^{\langle j\rangle}\right)$$

$$a^{\langle N\rangle} \leftarrow a^{\langle N+4\rangle} \quad \text{otherwise}$$

$as \leftarrow 0$

for $j \in 0..N$

$$as^{\langle j\rangle} \leftarrow a^{\langle j\rangle}$$

$as \leftarrow \mathrm{rsort}(as, N)$

$k \leftarrow k + 1$

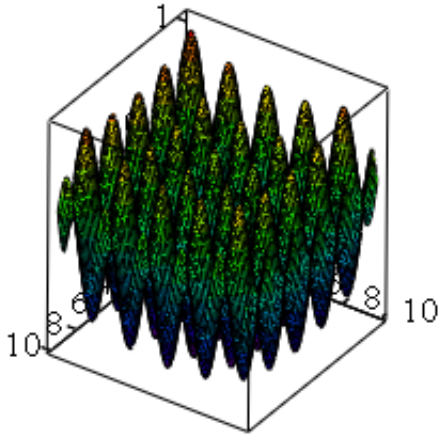
if $k > 5000\cdot N$

```
                ads ← as^⟨0⟩
                ads_{N+1} ← "Alarm Program stop in DM"
                ads_{N+2} ← k
                return ads
            x^⟨floor(0.9m)+r⟩ ← a^⟨0⟩
            x ← rsort(x, N)
            return x  if  z > 2m
        x^⟨0⟩
```

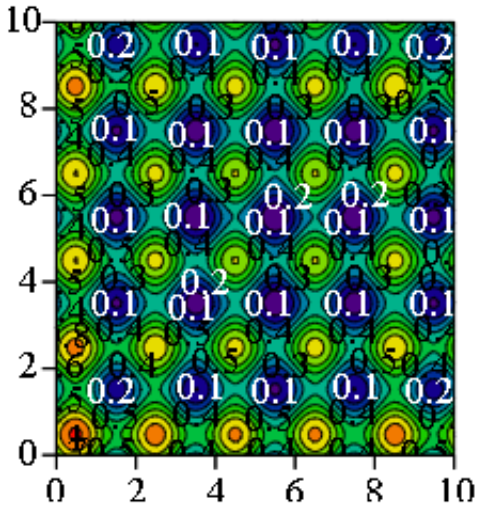
$$f\_Sqr(x1, x2) := (\sin(\pi \cdot x1) + \sin(\pi \cdot x2)) \cdot 0.2 \ldots + 0.01 \cdot \left[0.4 \cdot (x1 - 5.5)^2 + 0.5 \cdot (x2 - 5.5)^2\right] + 0.4$$

- цільова функція

$FF := CreateMesh(f\_Sqr, 0, 10, 0, 10, 100, 100)$ - побудова таблиці значень функції



FF



FF

$Sqr(x) := f\_Sqr(x_0, x_1)$ - перетворення цільової функції до виду з одним векторним аргументом

$\mu := 30$ - розмір популяції особин

$\varepsilon := .00001$ - задана точність

$xx := \begin{pmatrix} 0 \\ 0 \end{pmatrix}$ - нижні границі зміни аргументів функції

$xn := \begin{pmatrix} 10 \\ 10 \end{pmatrix}$ - верхні границі зміни аргументів функції

$V := MGA(\mu, xx, xn, \varepsilon, Sqr)$ - виклик функції

$$V = \begin{pmatrix} 5.501 \\ 5.499 \\ 1.289 \times 10^{-6} \end{pmatrix}$$ - отриманий розв'язок

ПЕРЕВІРКА РОЗВ'ЯЗКУ ЗАДАЧІ

$x := 5 \quad y := 6$ - початкові значення змінних

$$\text{Minimize}(f\_Sqr, x, y) = \begin{pmatrix} 5.5 \\ 5.5 \end{pmatrix}$$ - отриманий розв'язок

$f\_Sqr(5.5, 5.5) = 0$ - значення цільової функції в точці екстремуму

ЗАВДАННЯ ДО САМОСТІЙНОЇ РОБОТИ

Завдання 4. Знайти глобальний екстремум функції із застосуванням гібридного алгоритму, використовуючи варіанти завдань 1.

ТЕРМІНИ ТА ВИЗНАЧЕННЯ

Популяційні метаевристичні методи оптимізації — це алгоритми, що оперують множиною кандидатів на розв'язок (популяцією), одночасно досліджуючи різні області простору пошуку.

Популяційний метод - алгоритм, який на кожній ітерації формує та покращує низку розв'язів (популяцію), замість роботи з єдиним кандидатом. Приклади: генетичні алгоритми, оптимізація роєм часток, бджолині алгоритми.

Диверсифікація - стратегія дослідження широких областей простору пошуку для уникнення локальних екстремумів. Реалізується через механізми рекомбінації розв'язів або нелокальний пошук.

Рекомбінація - операція об'єднання частин батьківських розв'язів для створення нових кандидатів (наприклад, схрещування в генетичних алгоритмах).

Методи сканування простору - підклас популяційних методів, що формують напрями пошуку на основі аналізу поточної популяції.

Роєві алгоритми - імітують колективну поведінку біологічних систем (мурашині алгоритми, оптимізація роєм часток). Особливість: самоорганізація через локальні взаємодії агентів.

Еволюційні алгоритми - використовують біологічно натхненні оператори (селекція, мутація, схрещування).

Гібридні методи - поєднують популяційні підходи з локальним пошуком для підвищення точності.

Елітизм - збереження найкращих розв'язів між ітераціями для запобігання втраті якісних кандидатів.

Нелокальний пошук - створення нових розв'язів за межами поточних "околів" для подолання локальних оптимумів.

Стохастичні оператори - ймовірнісні правила для вибору, мутації або рекомбінації розв'язів .

Кооперативні метаевристики - використовують часткові моделі, сформовані різними алгоритмами, для пошуку в просторі розв'язів.

Природно-натхненні алгоритми - включають методи, що імітують поведінку бджіл, світлячків, бактерій або косяків риб

Генетичний алгоритм (ГА) — це стохастичний метод оптимізації, заснований на принципах природного відбору та генетичної еволюції. Використовується для розв'язання складних задач з високою розмірністю простору пошуку.

Алгоритм оптимізації роєм часток (PSO) — це метаевристичний алгоритм, натхненний колективною поведінкою соціальних організмів, таких як зграї птахів або косяки риб. Він використовується для пошуку оптимальних розв'язів у багатовимірних просторах шляхом ітеративного покращення кандидатів-розв'язів (часток).

Алгоритм імітації відпалу — метаевристичний метод глобальної оптимізації, що імітує фізичний процес відпалу матеріалів для пошуку оптимальних розв'язів у задачах з численними локальними екстремумами.

СПИСОК ЛІТЕРАТУРИ

1. Oliveira, C. A., & Pardalos, M. P. (2025). Handbook of Artificial Intelligence and Data Sciences for Routing Problems.
2. Hamadneh, T., Batiha, B., Gharib, G. M., Montazeri, Z., Dehghani, M., Aribowo, W., ... & Ibraheem, I. K. (2025). Revolution Optimization Algorithm: A New Human-based Metaheuristic Algorithm for Solving Optimization Problems. *International Journal of Intelligent Engineering & Systems*, *18*(2).
3. Cuevas, E., Chavarin-Fajardo, A., Ascencio-Piña, C., & Garcia-De-Lira, S. Optimization Strategies: A Decade of Metaheuristic Algorithm Development.
4. Cotta, C. (2025). Harnessing memetic algorithms: a practical guide. TOP, 1-30.
5. Serbet, F., & Kaya, T. (2025). New comparative approach to multi-level thresholding: chaotically initialized adaptive meta-heuristic optimization methods. Neural Computing and Applications, 1-26.
6. Mousavirad, S. J., Schaefer, G., Rezaee, K., Oliva, D., Zabihzadeh, D., Chakrabortty, R. K., ... & Pedram, M. (2025). A novel metaheuristic population algorithm for optimising the connection weights of neural networks. Evolving Systems, 16(1), 1-23.
7. Dong, Y., Zhang, S., Zhang, H., Zhou, X., & Jiang, J. (2025). Chaotic evolution optimization: A novel metaheuristic algorithm inspired by chaotic dynamics. Chaos, Solitons & Fractals, 192, 116049.
8. Liu, X., Wang, T., Zeng, Z., Tian, Y., & Tong, J. (2025). Three stage based reinforcement learning for combining multiple metaheuristic algorithms. Swarm and Evolutionary Computation, 95, 101935.
9. Kumari, M. (2025). A review on metaheuristic algorithms: Recent and future trends. Metaheuristics-Based Materials Optimization, 103-128.
10. Гулаєва, Н. М., & Шило, В. П. (2021). Генетичні алгоритми як обчислювальні методи скінченновимірної оптимізації. Кібернетика та комп'ютерні технології.
11. Герега, Б. Д. (2021). Еволюційні алгоритми глобальної пошукової оптимізації.
12. Авдусь, А. В. (2020). Дослідження та використання алгоритмів еволюційного моделювання в інтелектуальних системах прийняття розв'язів.
13. Гуляницький, Л. Ф., & Мулеса, О. Ю. (2015). До класифікації метаевристик.