Одеський національний університет імені І. І. Мечникова
Факультет математики, фізики та інформаційних технологій
Кафедра оптимального керування та економічної кібернетики

Кваліфікаційна робота

на здобуття ступеня вищої освіти «бакалавр»

«Чисельні методи пошуку екстремуму за наявністю похибок»

«Numerical methods of finding the extremum based on the presence of errors»

Виконав: здобувач денної форми навчання
спеціальності 113 Прикладна математика
Освітня програма «Прикладна математика»

Балицький Ілля Олегович

Керівник Канд. фіз-мат наук, доц. Таірова М.С.
(науковий ступінь, вчене звання, прізвище та ініціали, підпис)
Рецензент Канд. фіз-мат наук, доц. Васильєв О. Б.
(науковий ступінь, вчене звання, прізвище та ініціали)

Рекомендовано до захисту:
Протокол засідання кафедри
№ ___ від _____2023 р.

Завідувач кафедри

_____ _____
(підпис) (прізвище, ініціали)

Захищено на засіданні ЕК №_____
протокол № ____ від _____ 2023 р.
Оцінка_____/___ ___/_____
(за національною шкалою, шкалою ECTS, бали)

Голова ЕК

_____ _____
(підпис) (прізвище, ініціали)

Одеса – 2023

ЗМІСТ

ВСТУП

У сучасному світі ми стикаємося з постійним зростанням складності завдань адаптації, навчання, ідентифікації, розпізнавання, планування, прогнозування та екстремального регулювання. Всі вони вимагають від нас знаходження екстремуму функції в умовах обмеженої інформації. Для розв'язання таких складних завдань існує багато алгоритмів. Найчастіше використовуються рекурентні алгоритми, включаючи градієнтні методи, методи Ньютона та псевдоградієнтні методи. Проте, у випадках, коли ми стикаємося з похибками, вибір оптимального алгоритму стає вкрай важливим. Проблема оптимальності алгоритмів залишається відкритою і потребує подальшого наукового дослідження. Головною метою кваліфікаційної роботи є аналіз та вивчення впливу наявності похибок на ефективність чисельних методів пошуку екстремуму. Зосередився на визначенні поведінки деяких рекурентних алгоритмів в умовах наявності похибок.

Мета роботи - встановити, який алгоритм буде більш оптимальним в умовах наявності похибок та яку поведінку вони застосовують задля мінімізації впливу похибок на остаточний результат. Вивчення цих аспектів є критично важливим для оптимізації рішень, що стосуються реального світу. В роботі на першому етапі були розглянуті теоретичні відомості щодо джерел та типів похибок. Далі був зроблений огляд методів мінімізації за наявністю похибок. Була наведена практична реалізація градієнтного методу за наявністю похибок і зроблений аналіз наведених методів та рекомендації щодо їх використання.

РОЗДІЛ 1. Джерела та типи похибок

1.1 Джерела похибок

У реальних задачах застосувати методи мінімізації «у чистому вигляді» не завжди є доцільним — ситуація неминуче ускладнюється наявністю різноманітних помилок і похибок.

Є доцільним розглянути задачу мінімізації за наявністю похибки. Нехай треба мінімізувати функцію $f(x)$.

Можна перерахувати деякі джерела з яких виникають похибки:

1. *Похибки обчислення:*

У найпростішому випадку, коли мінімізована функція та її градієнт задані формулами, похибки виникають внаслідок похибок обчислення, пов'язаних з округленням при виконанні арифметичних дій на ЕОМ. В результаті $f(x_k)$ та $\nabla f(x_k)$ обчислюються з деякою похибкою, тобто замість вектора $f(x_k)$ отримуємо вектор $s(k) = f(x_k) + r_k$.

2. *Результат виміру на реальних об'єктах:*

У ряді задач ми отримуємо значення $f(x_k)$ та $\nabla f(x_k)$ не за допомогою обчислень,

а в результаті вимірювань. Такою є ситуація при оптимізації на реальному об'єкті (екстремальне регулювання, планування експерименту). Тоді похибки мають випадковий характер, властивий похибкам вимірів. При цьому зазвичай буває доступна інформація про рівень та статистичну природу похибки.

3. *Невизначеність або відсутність повної інформації:*

Нерідко (особливо в задачах адаптації, навчання, розпізнавання і т. д.) проблема оптимізації ставиться таким чином: потрібно мінімізувати детерміновану функцію середнього ризику $f(x)$

$$f(x) = \mathrm{M}Q(x, \omega) = \int Q(x, \omega) d\mathrm{P}(\omega)$$

де функція $Q(x, \omega)$ відома, проте розподіл $P(\omega)$ не заданий. Дана лише вибірка $x_1, ..., x_k$ із цього розподілу. Тоді точне обчислення $f(x)$ та $\nabla f(x)$ в принципі

неможливе. Можна взяти лише наближене значення цих величин:
$Q(x, \omega k)$ і $\Delta Q(x, \omega k)$.

У цьому випадку значення функції та градієнта містять випадкову похибку. Якщо брати як наближення для $f(x_k)$ і $\nabla f(x_k)$ величини $Q(x_k, \omega_k)$ і $\nabla Q(x_k, \omega_k)$, то похибки будуть незалежні в різних точках.

4. *Трудомісткість обчислення:*

Аналогічна ситуація виникає в методі Монте-Карло, коли завдання полягає в мінімізації $f(x)$ і розподіл $P(\omega)$ відоме, проте обчислення інтеграла надто трудомістке. Тоді можна точні значення $f(x)$ та $\nabla f(x)$ замінити вибірковими значеннями, як і вище.

5. *Непереборність похибки:*

У ряді завдань похибки виникають через те, що значення функції та градієнта обчислюються за спрощеними або наближеними формулах. Нерідко точне обчислення вимагає громіздкого розрахунку функцій впливу, розв'язання складних допоміжних завдань, урахування взаємодії всіх параметрів і т. д. Всі ці обчислення недоцільно (іноді й неможливо) проводити повністю. Їх спрощення призводять до похибок у визначенні функції та градієнта. Це так звані непереборні похибки.

6. *Похибки методу:*

У багатьох методах похибки виникають через необхідність вирішення допоміжних завдань, яке не може бути здійснено точно. Наприклад, у методі Ньютона на кожному кроці треба вирішувати систему лінійних рівнянь, що неминуче пов'язано з похибками; у методі сполучених градієнтів потрібно проводити одновимірну мінімізацію, що також може бути зроблено лише приблизно і т. д. У такому випадку говорять про похибки методу.

1.2 Типи похибок

Як ми бачили вище, похибки при обчисленні функції та градієнта можуть

мати різне походження та різну природу. Дещо спрощуючи реальну ситуацію, можна виділити такі основні типи похибок. Всюди нижче йдеться про обчислення градієнта, коли замість точного значення f (x^k) нам доступний вектор

s(k) =∇f (x^k) + r^k. Загалом відрізняють такі типи похибок:

1. *Абсолютні детерміновані похибки*

Вони задовольняють умову || r^k || ≤ ε

Тож градієнт вираховується з заданою абсолютною похибкою. Передбачається що похибка потрібна лиш задовольняти цю умову, тобто зокрема вектор r_k може не бути випадковим або він може залежати від інших похибок і т.д.

2. *Відносні детерміновані похибки*

Вони задовольняють умову || r^k || ≤ ε||∇f (x^k)||, тобто градієнт вираховується з відносною похибкою. Крім цієї умови, передбачається що о похибці нічого не відомо, отже такі похибки можуть виникати, наприклад, при застосуванні формул, які дають фіксовану відносну похибку.

3. *Абсолютні випадкові похибки*

Припустимо що похибки r^k випадкові і незалежні від x, центровані та мають обмежену дисперсію: $Mr^k = 0, \quad M\left\|r^k\right\|^2 \leqslant \sigma^2.$

Похибки такого типу характерні для завдань, у яких градієнт отримуємо в результаті виміру на реальних об'єктах або у завдань з функцією типу середнього ризику

4. *Відносні випадкові похибки*

Мають ті ж умови та властивості третього типу, але відрізняються тим, що їх дисперсія зменшується з приближенням до точки мінімума: $\mathbf{M}r^k = 0, \quad \mathbf{M}\left\|r^k\right\|^2 \leqslant \tau\left\|\nabla f\left(x^k\right)\right\|^2$ ·Бувають також інші типи похибок як систематичні або відносні обмежені похибки($Mr^k = 0$, $||r^k|| \leq \varepsilon$) але усі ці похибки врешті є лише комбінацією цих основних типів.

РОЗДІЛ 2. Огляд методів мінімізації за наявністю похибок

2.1 Метод Ньютона

Розглянемо кілька характерних прикладів, цікавлячись лише аналізом якості процесу: Нехай у результаті всіх обчислень виходить вектор, що відрізняється від істинного:

$s^k = [\nabla^2 f(x^k)]^{-1} \nabla f(x^k) + r^k$, де r^k похибка, та робиться шаг $x^{k+1} = x^k - s^k$.

Нехай буде систематична похибка: $\| r^k \| \leq \varepsilon$.

Труднощі виникнуть бо метод Ньютона сходиться локально в деякій області U та якщо ε є більшою за U то сходимості не буде - при любому x_0 скільки завгодно близьким до x_* процес виходить за U.

Випадкові або відносні похибки не роблять метод непридатним - проте вони значно впливають на швидкість його сходження.

Нехай потрібно мінімізувати квадратичну функцію

$f(x) = (Ax, x)/2 - (b, x), \ A > 0$

та матриці A та A^{-1} обчислюються точно, а градієнт має випадкову похибку:

$s^k = \nabla f(x^k) + r^k = Ax^k - b + r^k, \quad Mr^k = 0, \ M||r^k||^2 = \sigma^2$

Розглянемо узагальнений метод Ньютона:

$x^{k+1} = x^k - \gamma_k A^{-1} s^k$, цей метод не буде збігатись швидше ніж за O(1/k), проте таку швидкість може забезпечити більш простий метод - метод градієнту. Таким чином метод Ньютона втрачає його основну перевагу - швидкість збіжності. Аналогічно з відносною похибкою - для градієнту з відносною похибкою метод Ньютона буде збігатись зі швидкістю геометричної прогресії. Метод Ньютона сохраняє свої переваги лише з високою точністю обчислень.

2.2 Метод важкої кульки

Розглянемо метод при абсолютних випадкових похибках: нехай

$f(x) = (Ax, x)/x - (b, x), \quad lI \le A \le LI, \quad l>0,$

$s^k = \nabla f(x^k) + r^k = Ax^k - b + r^k, \; Mr^k = 0, \; Mr^k (r^k)^T = \sigma^2 I,$

причому похибки r^k взаємно незалежні. Отже метод важкої кульки $x^{k+1} = x^k - as^k + \beta(x^k - x^{k-1})$ в цій ситуації не збігається до $x^* = A^{-1}b$, а призводить лише до області навколо x^*. Однак метод з змінними коефіцієнтами: $x^{k+1} = x^k - a_k y^k, \; y^{k+1} = y^k - \beta_k (y^k - s^k)$ буде повільніше ніж градієнтний метод

$x^{k+1} = x^k - \gamma_k s^k$

2.3 Квазіньютонівські методи

Квазіньютонівські методи дуже чутливі до похибок обчислення градієнта. Дійсно, у них відновлюється матриця $A = \nabla^2 f(x)$ за вимірюваннями градієнта:

$Ap^i \approx y^i, \qquad p^i = x^{i+1} - x^i, \quad y^i = \nabla f(x^{i+1}) - \nabla f(x^i),$ i = 0,...., k-1.

Якщо шаг малий (x^{i+1} близько до x^i), а обчислення $\nabla f(x^i)$ містить похибки, то матриця буде відновлюватись погано. Для завдань с випадковим похибками це можна рішати шляхом збільшення числа вимірів - треба відновлювати не по n значеннях $\nabla f(x)$, а по N>n замірам. Для невипадкових похибок ця модифікація не призводить до збільшення точності.

2.4 Градієнтний метод за наявністю похибок

Постановка задачі

Розглянемо градієнтний метод мінімізації функції $f(x)$ на R^n в ситуації коли градієнт обчислюється с похибкою:

$x^{k+1} = x^k - \gamma_k s^k, \quad s^k = \nabla f(x^k) + r^k.$

Функція $f(x)$ буде сильно випуклою (з константою l) та з градієнтом, що задовольняє умову Ліпшица (з константою L).

Нам потрібно дослідити поведінку звичайного градієнтного методу з $\gamma_k \equiv \gamma$ за наявності похибок, а також вплив вибору довжини шагу в умовах похибок.

Абсолютні детерміновані похибки

Теорема (1). Нехай $\| r^k \| \leq \varepsilon$, $\gamma_k \equiv \gamma$. Тоді знайдеться таке $\bar{\gamma} > 0$, що при $0 < \gamma < \bar{\gamma}$, в методі $x^{k+1} = x^k - \gamma_k s^k, \quad s^k = \nabla f(x^k) + r^k$ буде $\left\| x^k - x^* \right\| \leq \rho + q^k \left\| x^0 - x^* \right\|$, де $0 \leq q < 1$, $\rho = \rho(\varepsilon)$, при $\varepsilon \to 0$, x^* – точка мінімуму $f(x)$.

Доказ:

Введемо функцію Ляпунова: $V(x) = \frac{1}{2}\left(\left\| x - x^* \right\| - \frac{1}{l}\varepsilon \right)_+^2$.

Отримаємо :

$$\left(\nabla V\left(x^k\right), s^k\right) = \left(\left\|x^k - x^*\right\| - \frac{1}{l}\varepsilon\right)_+ \frac{\left(\nabla f\left(x^k\right) + r^k, x^k - x^*\right)}{\left\|x^k - x^*\right\|} \geqslant$$

$$\geqslant \left(\left\|x^k - x^*\right\| - \frac{1}{l}\varepsilon\right)_+ \left(l\left\|x^k - x^*\right\| - \varepsilon\right) = 2lV\left(x^k\right),$$

$$\left\|s^k\right\|^2 = \left\|\nabla f\left(x^k\right) + r^k\right\|^2 \leqslant \left(L\left\|x^k - x^*\right\| + \varepsilon\right)^2 \leqslant$$

$$\leqslant a + bV\left(x^k\right) \leqslant a + (b/(2l))\left(\nabla V\left(x^k\right), s^k\right),$$

де a,b - деякі константи, також $a \to 0$ при $\varepsilon \to 0$.

Виходячи з цього бачимо що метод з постійним γ перестає збігатись до точки мінімуму. Він дає лиш можливість лише попасти у деяку околицю мінімуму, розміри якої пропорціональні рівню похибок. Збіжність проходить зі швидкістю геометричної прогресії.

Відносні детерміновані похибки

Теорема (2). Нехай $\| r^k \| \le \alpha \left\| \nabla f(x^k) \right\|$, $\alpha < 1$, $\gamma_k \equiv \gamma$. Тоді знайдеться таке $\bar{\gamma} > 0$, що при $0 < \gamma < \bar{\gamma}$, метод градієнту $x^{k+1} = x^k - \gamma_k s^k$, $s^k = \nabla f(x^k) + r^k$

буде збігатись зі швидкістю геометрічної прогресії.

Доказ:

Візьмемо як функцію Ляпунова - $V(x) = f(x) - f(x^*)$. Тоді:

$(\nabla V(x^k), s^k) = (\nabla f(x^k), \nabla f(x^k) + r^k) \ge (1 - \alpha) \left\| \nabla f(x^k) \right\|^2 \ge (1 - \alpha) 2lV(x^k)$,

$\left\| s^k \right\|^2 \le \left\| \nabla f(x^k) \right\|^2 (1 + \alpha)^2 \le 2(1 + \alpha)^2 LV(x^k)$

Таким чином, метод стійкий к відносним похибкам, якщо їх рівень менш ніж 100%. Причиною цього очевидна - усякий напрямок з градієнтом що робить гострий кут - є напрямом зменшення функції $f(x)$ та може бути застосований сам як напрямок руху замість градієнту.

Абсолютні випадкові похибки

Нехай похибки r^k випадкові, незалежні, та $Mr^k = 0$ та $M \left\| r^k \right\|^2 \le \sigma^2$.

Теорема (3). Знайдеться таке $\bar{\gamma} > 0$, що при $\gamma^k \equiv \gamma$, де при $0 < \gamma < \bar{\gamma}$ у методі градієнту $M(f(x^k) - f^*) \le \rho(\gamma) + M(f(x^0) - f^*) q^k$, де $q < 1$, $\rho(\gamma) \to 0$ при $\gamma \to 0$. Якщо $\gamma_k \to 0$, $\sum_{k=0}^{\infty} \gamma_k = \infty$, тоді $M \left\| x^k - x^* \right\|^2 \to 0$.

Але якщо $\sum_{k=0}^{\infty} \gamma_k^2 < \infty$, $\sum_{k=0}^{\infty} \gamma_k = \infty$, то $x^k \to x^*$. Нарешті якщо $\gamma_k = \gamma / k, \gamma > (2l)^{-1}$, тоді $M(f(x^k) - f^*) \le \frac{L\sigma^2 \gamma^2}{2(2l\gamma - 1)k} + o(\frac{1}{k})$

Доказ:

Візьмемо функцію $V(x) = f(x) - f^*$.

Тоді $(\nabla V(x^k), Ms^k) = (\nabla f(x^k), \nabla f(x^k)) \geq 2lV(x^k)$,

$M\left\|s^k\right\|^2 = \left\|\nabla f(x^k)\right\|^2 + M\left\|r^k\right\|^2 \leq \sigma^2 + (\nabla V(x^k), Ms^k)$.

По-перше, звичайний варіант методу з $\gamma^k \equiv \gamma$ не забезпечує збіжність методу при таких похибках, а приводить лише в околицю мінімуму, розмірі якої пропорціональні γ. По-друге, якщо вибирати спадаючі γ_k, то можна зробити метод збіжним (у середньому при $\gamma_k \to 0$ та майже збіжним при $\sum_{k=0}^{\infty} \gamma^2_k < \infty$). По-третє, швидкість збіжності дуже повільна (близко O(1/k)), яку ніяк не покращити за будь-яких γ_k

Відносні випадкові похибки

Нехай похибки r^k аналогічні пункту з абсолютними випадковими, але їх дисперсія задовольняє $M\left\|r^k\right\|^2 \leq \alpha||\nabla f(x)||^2$.

Теорема (4). При будь-якому α існує таке $\bar{\gamma}$ таке, що при $\gamma_k \equiv \gamma,\ 0 < \gamma < \bar{\gamma}$ у методі градієнту буде задовольняти $M\left\|x^k - x^*\right\|^2 \leq cq^k,\ q < 1$

Бачимо, що наявність випадкових відносних похибок будь-якого рівню не призводить до порушення збіжності.

Висновки

Отже, залежно від типу похибок їх наявність може або зберігати або порушувати збіжність градієнтного методу. Іноді збіжність можна відновити за рахунок зміни та регулювання довжини шагу

Похибки	Вплив
Без похибок	Має збіжність до мінімуму, та лінійну швидкість збіжності
Абсолютні детерміновані похибки	Може знизити швидкість сходження, сходиться до околиці мінімуму, розміри якої пропорціональні до рівню похибок
Відносні детерміновані похибки	Метод стійкий к відносним похибкам, якщо їх рівень менш ніж 100%, має швидкість геометричної прогресії.
Абсолютні випадкові похибки	Звичайний варіант методу з $\gamma^k \equiv \gamma$ не забезпечує збіжність методу при таких похибках, а приводить лише в околицю мінімуму, розмірі якої пропорціональні γ, швидкість збіжності дуже повільна (близко O(1/k)).
Відносні випадкові похибки	Наявність випадкових відносних похибок будь-якого рівню не призводить до порушення збіжності, має лінійну швидкість збіжності.

РОЗДІЛ 3. Практична реалізація градієнтного методу за наявністю похибок

3.1 Опис алгоритму

Розглянемо задачу мінімізації функції за наявністю похибок, в якості методу розв'язку виберемо градієнтний метод. Нехай робоча область функції (заданий інтервал) розбита на кілька точок. Вибрано точки локальних мінімумів. Після цього всі координати передаються функції Розенброка $f(x, y) = (1 - x)^2 + 100 * (y - x^2)^2$ як аргументи і вибирається аргумент, що дає найменше значення. Потім при наявності різних типів похибки будемо використовувати метод градієнтного спуску та аналізувати отримані результати.

3.2 Програмна реалізація градієнтного методу за наявністю похибок

Приклад 1:

Дано:

Використовуємо функцію без похибок з обмеженою кількістю ітерацій

Вивід функції:

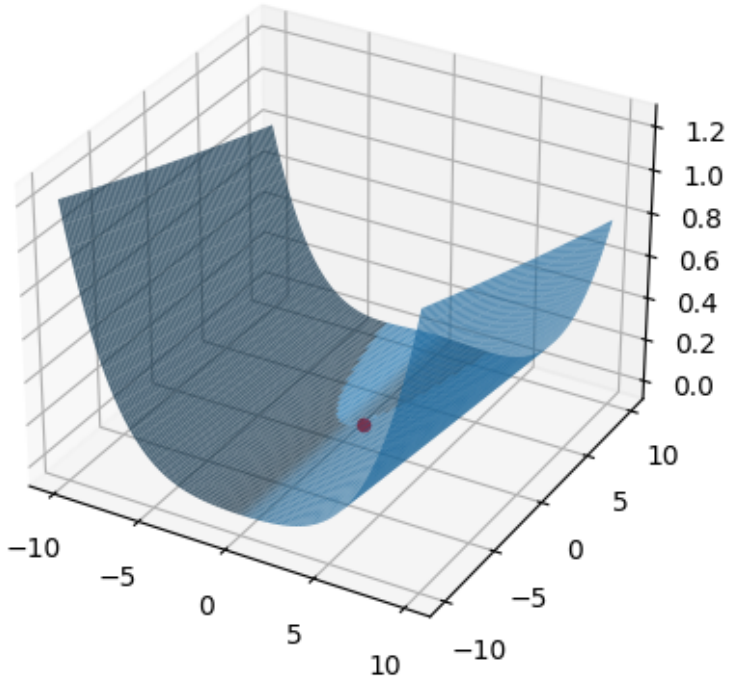

Рис. 3.2: Вивід графіку для градієнтного спуску на функції

Function with noise type: No noise

Координата мінімуму функції :

min x: 0.9711087314571791, min y: 0.4377888011196459,

Кількість ітерацій:

max iterations: 100,

Час роботи функції:

elapsed time: 0.7833459377288818s

Приклад 2:

Дано:

Використовуємо функцію з абсолютною детермінованою похибкою з обмеженою кількістю ітерацій

Вивід функції:

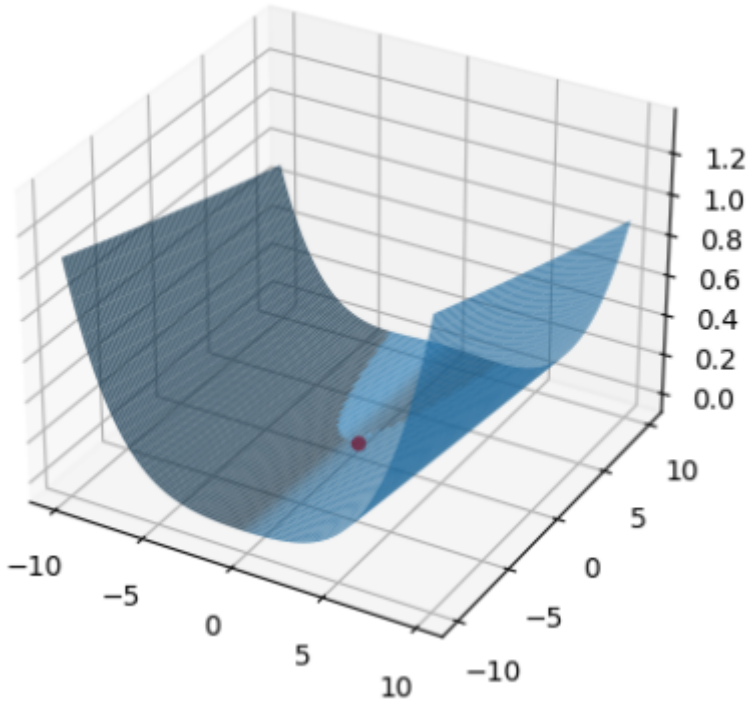

Рис. 3.3: Вивід графіку для градієнтного спуску на функції.

Function with noise type: absolute_deterministic

Координата мінімуму функції :

min x: 0.42357886921272814, min y: -0.2818356698891487,

Кількість ітерацій

max iterations: 150,

Час роботи функції:

elapsed time: 0.79986572265625s

Приклад 3:

Дано:

Використовуємо функцію з абсолютною випадковою похибкою з обмеженою кількістю ітерацій

Вивід функції:

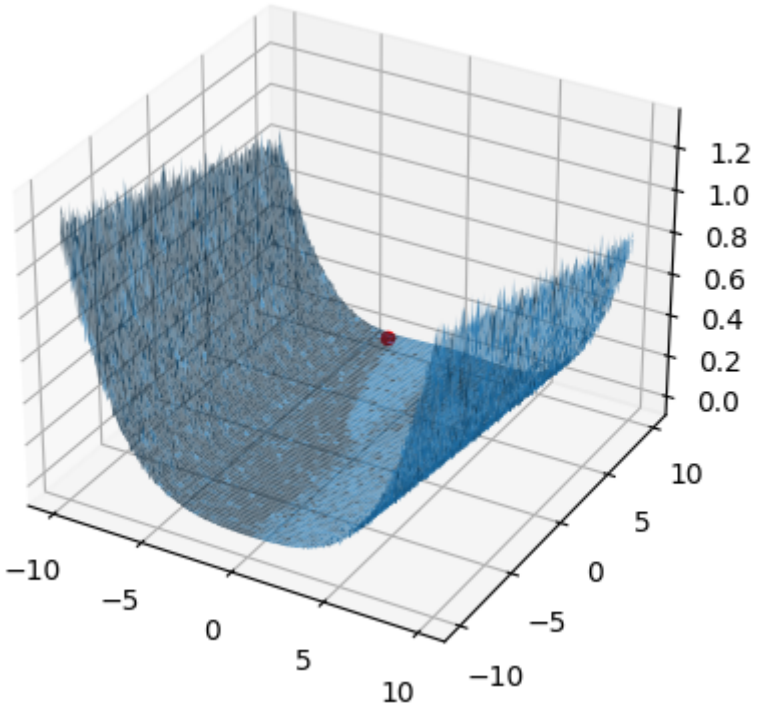

Рис. 3.4: Вивід графіку для градієнтного спуску на функції

Function with noise type: absolute_random

Координата мінімуму функції:

min x: -3.4331004250904034, min y: 9.399941577819797,

Кількість ітерацій:

max iterations: 100,

Час роботи функції:

elapsed time: 1.553809642791748s

Приклад 4:

Дано:

Використовуємо функцію з відносною детермінованою похибкою з обмеженою кількістю ітерацій

Вивід функції:

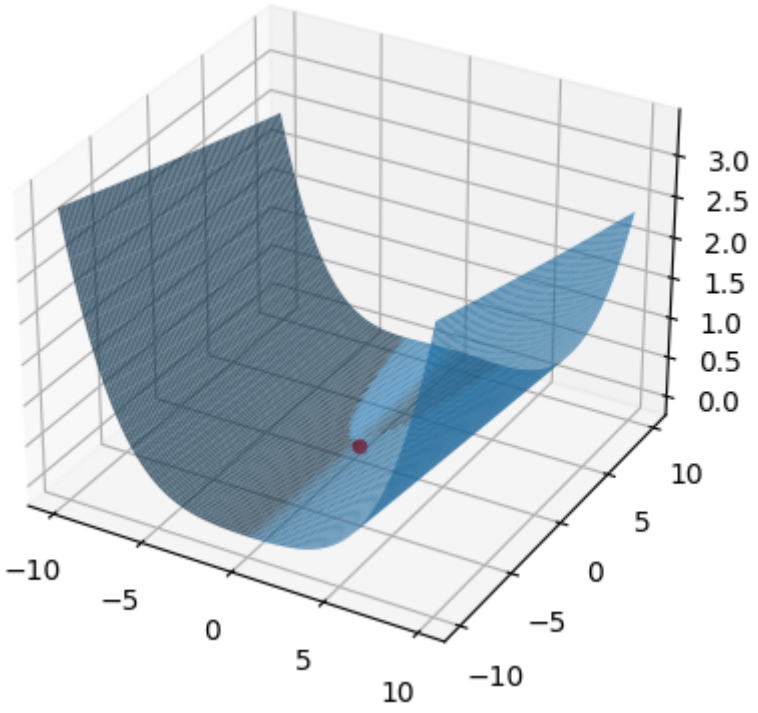

Рис. 3.4: Вивід графіку для градієнтного спуску на функції

Координата мінімуму функції:

Function with noise type: relative_deterministic

min x: 0.41357886921272813, min y: -0.2818356698891487

Кількість ітерацій:

max iterations: 200

Час роботи функції:

elapsed time: 0.8064403533935547s

Приклад 5:

Дано:

Використовуємо функцію з відносною випадковою похибкою з обмеженою кількістю ітерацій

Вивід функції:

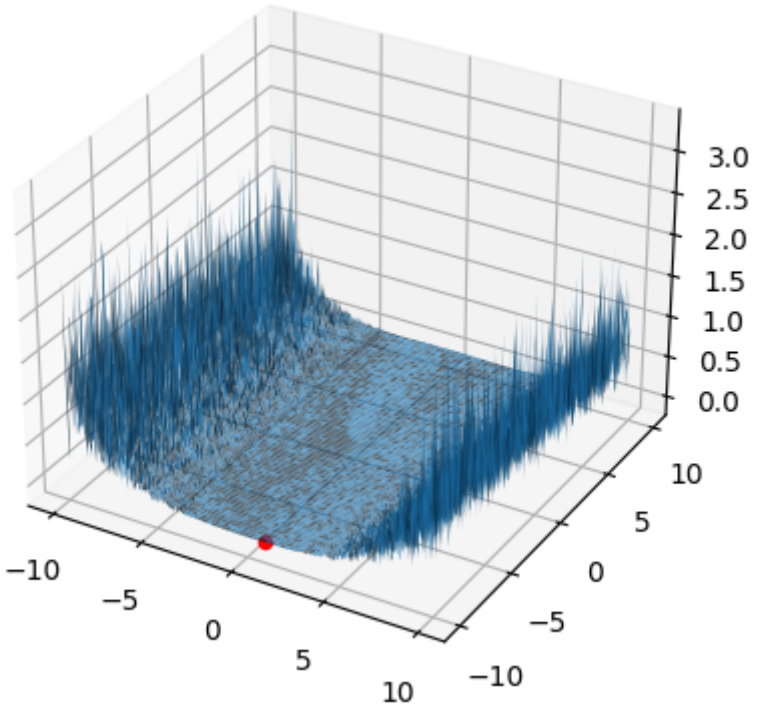

Рис. 3.4: Вивід графіку для градієнтного спуску на функції

Function with noise type: relative_random

Координата мінімуму функції:

min x: 0.867199641339196, min y: -9.922137245012872

Кількість ітерацій:

max iterations: 100

Час роботи функції:

elapsed time: 1.7106127738952637s

ВИСНОВОК

В Кваліфікаційній роботі були розглянуті джерела та типи похибок. Потім були розглянуті та порівнянні між собою методи мінімізації за наявністю похибок. Було розроблено програму, що реалізує порівняння роботи методу градієнтного пошуку за наявністю похибок та без них. Також було наведено та проаналізовано основні методи вирішення проблематики завдання. Було зроблено алгоритм що реалізує градієнтний метода при наявності та без наявності похибок і зроблено порівняльній аналіз.

Однією з основних цілей створення програми для чисельних методів пошуку екстремуму з оцінкою роботу методу при та без наявності похибок є зменшення витрат часу та ресурсів, пов'язаних з розробкою об'єкта.

Ефективність чисельних методів пошуку екстремуму має важливе значення для досягнення цієї мети. Вибір алгоритму залежить від специфіки задачі та наявності похибок.

Похибки, які можуть виникати під час розробки об'єкта, можуть призвести до невірних результатів або не коректної роботи програми. Ціллю програми є визначення таких похибок шляхом порівняння роботи методу та, якщо це можливо - відновлення збіжності чи пришвидшення алгоритму шляхом виправлення роботи методу при похибках.

У рамках цієї кваліфікаційної роботи розглядаються різні види вирішення проблематики. Треба було розробити програмний модуль градієнтного пошуку, що реалізує такі функції:

- Прийом вхідних параметрів та їх обмежень.
- Мінімізація цільової функції одним із методів градієнтного пошуку та додавання випадкової похибки, отримання вектора параметрів.
- Порівняння отриманого вектора параметрів із похибкою та без похибки

Завдання виконано в повному обсязі та задовольняє всім потребам.

Основна мета, а саме : визначити вплив похибок на градієнтний метод була досягнута.

СПИСОК ЛІТЕРАТУРИ

1. Р. Габасов, Ф.М. Кирилова Методи оптимізації. - Мінськ: Б ГУ, 1975.

2. А. Е. Кононюк Математика. К.1. - К.: КМТ, 2009.

3. А. Е. Кононюк Математика. К.2. - К.: КМТ, 2009.

4. А. Е. Кононюк Математика. К.1, ч.1 - К.: Освіта України , 2010.

5. А. Е. Кононюк Математика. К.1, ч.2 - К.: Освіта України , 2010.

6. А. Е. Кононюк Математика. К.2, ч.1 - К.: Освіта України , 2011.

7. А. Е. Кононюк Математика. К.2, ч.2 - К.: Освіта України , 2011.

8. А. Е. Кононюк Математика. К.2, ч.3 - К.: Освіта України , 2011.

9. А. Е. Кононюк Математика. К.3, ч.1 - К.: Освіта України , 2011.

10. А. Е. Кононюк Математика. К.3, ч.2 - К.: Освіта України , 2011.

11. А.Т. Яровий Методи оптимізації та варіаційне числення: Навчально-методичний посібник для студентів спеціальності “Математика” / А.Т.Яровий, Є.М.Страхов, 2017р. - С. 84-86

12. А.Т. Яровий Методи оптимізації: Навчально-методичний посібник для студентів спеціальностей “Математика” та “Прикладна математика”/ А.Т.Яровий, Є.М.Страхов, 2020р. - С. 68-71

13. Д.М. Хіммельблау Прикладне нелінійне програмування, 1975р. - С. 98-107

ДОДАТКИ

Додаток А

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

# Constants
RADIUS = 10
GLOBAL_EPSILON = 1e-9
CENTRE = (GLOBAL_EPSILON, GLOBAL_EPSILON)
NUM_POINTS = 200
STEP = RADIUS / NUM_POINTS

# Different types of noise
NOISE_TYPES = {
  'absolute_deterministic': lambda x: x + 0.3,
  'absolute_random': lambda x: x + np.random.normal(0, 0.1, (1,)),
  'relative_deterministic': lambda x: x + 0.3 * x,
  'relative_random': lambda x: x + np.random.normal(0, 0.1, (1,)) * x,
}


# The original function
def function(x, y, noise_type=None):
  noise_func = NOISE_TYPES.get(noise_type)
  x = noise_func(x) if noise_func else x
  y = noise_func(y) if noise_func else y
  return (1 - x) ** 2 + 100 * (y - x ** 2) ** 2
```

```
# A helper function for rotate_vector and derivative
def function_with_eps(epsilon, x, y, is_x, noise_type):
  x, y = (x + epsilon, y) if is_x else (x, y + epsilon)
  return function(x, y, noise_type)


# Calculation of the derivative
def derivative(epsilon, x, y, is_x, noise_type):
  return (function_with_eps(epsilon, x, y, is_x, noise_type) -
        function_with_eps(0, x, y, is_x, noise_type)) / epsilon


# Rotate vector by alpha
def rotate_vector(length, alpha):
  return length * np.cos(np.radians(alpha)), length * np.sin(np.radians(alpha))


# Calculate flip points
def calculate_flip_points():
  flip_points = []
  for i in range(NUM_POINTS):
    for alpha in range(360):
      x, y = rotate_vector(STEP * i, alpha)
      x += CENTRE[0]
      y += CENTRE[1]
      if derivative(GLOBAL_EPSILON, x, y, True, None) + \
          derivative(GLOBAL_EPSILON, x, y, False, None) > 0:
```

```
            flip_points.append((alpha, i - 1))
    return flip_points


# Gradient descent method
def gradient_descent(best_x, best_y, is_x, noise_type):
    iteration_count = 0
    max_iterations = 100
    descent_step = 0.1 * STEP
    value = derivative(GLOBAL_EPSILON, best_x, best_y, is_x, noise_type)

    while abs(value) > GLOBAL_EPSILON and iteration_count < max_iterations:
        best_x, best_y = (best_x - descent_step, best_y) if value > 0 else (best_x +
descent_step, best_y)
        value = derivative(GLOBAL_EPSILON, best_x, best_y, is_x, noise_type)
        iteration_count += 1

    return best_x, best_y, iteration_count


# Find minimum of the function
def find_minimum(noise_type):
    start_time = time()

    best_x, best_y = CENTRE
    for alpha, i in calculate_flip_points():
        x, y = rotate_vector(STEP * i, alpha)
        x += CENTRE[0]
        y += CENTRE[1]
```

```
        if function(x, y, noise_type) < function(best_x, best_y, noise_type):
            best_x, best_y = x, y

    for alpha in range(360):
        x, y = rotate_vector(STEP * (NUM_POINTS - 1), alpha)
        x += CENTRE[0]
        y += CENTRE[1]
        if function(x, y, noise_type) < function(best_x, best_y, noise_type):
            best_x, best_y = x, y

        best_x, best_y, max_iterations_x = gradient_descent(best_x, best_y, True, noise_type)
        best_y, best_x, max_iterations_y = gradient_descent(best_y, best_x, False, noise_type)

    elapsed_time = time() - start_time

    return best_x, best_y, max(max_iterations_x, max_iterations_y), elapsed_time



# Draw chart
def draw_chart(best_x, best_y, noise_type):
    fig = plt.figure()
    ax = fig.add_subplot(111, projection='3d')
    x = y = np.arange(-RADIUS, RADIUS, STEP)
    X, Y = np.meshgrid(x, y)
    zs = np.array([function(x, y, noise_type) for x, y in zip(np.ravel(X), np.ravel(Y))])
    Z = zs.reshape(X.shape)
    ax.plot_surface(X, Y, Z, rstride=5, cstride=5, alpha=0.7)
```

```
  ax.scatter(best_x, best_y, function(best_x, best_y, noise_type), color='red')
  plt.show()


# Main loop
def main():
  for noise_type in [None] + list(NOISE_TYPES.keys()):
    print(f"\nFunction with noise type: {noise_type if noise_type else 'No noise'}")
    min_x, min_y, max_iterations, elapsed_time = find_minimum(noise_type)
      print(f"min x: {min_x}, min y: {min_y}, max iterations: {max_iterations},
elapsed time: {elapsed_time}s")
    draw_chart(min_x, min_y, noise_type)

if __name__ == '__main__':
  main()
```