МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
федеральное государственное бюджетное образовательное учреждение высшего образования
«Тольяттинский государственный университет»

Институт математики, физики и информационных технологий
(наименование института полностью)

Кафедра «Прикладная математика и информатика»
(наименование кафедры)
01.03.02 Прикладная математика и информатика
(код и наименование направления подготовки, специальности)
Системное программирование и компьютерные технологии
(направленность (профиль)/специализация

БАКАЛАВРСКАЯ РАБОТА

на тему: «Сравнительный анализ эвристических алгоритмов минимизации функций нескольких переменных»

Студент З.Г. Гадоев _____
(И.О. Фамилия) (личная подпись)
Руководитель О.В. Лелонд _____
(И.О. Фамилия) (личная подпись)
Консультанты Н.В. Андрюхина _____
(И.О. Фамилия) (личная подпись)

Допустить к защите

Заведующий кафедрой к.т.н., доцент, А.В. Очеповский _____
(ученая степень, звание, И.О. Фамилия) (личная подпись)
«_____»_____20_____г.

Тольятти 2019

АННОТАЦИЯ

Тема бакалаврской работы: - «Сравнительный анализ эвристических алгоритмов минимизации функций нескольких переменных».

В данной бакалаврской работе исследуются способы поиска экстремумов функций от нескольких переменных с использованием генетических алгоритмов.

В работе предложены несколько вариантов генетических алгоритмов, свойства которых исследуются с помощью функций Растрыгина, Швефеля, Грайвенка. Для визуализации процесса поиска экстремумов функции разработано собственное программное обеспечение.

Структура бакалаврской работы представлена введением, тремя главами, заключением, списком литературы.

Во введении описывается актуальность проводимого исследования, дается краткая характеристика проделанной работы.

В первой главе проводится обзор эвристических алгоритмов поиска экстремумов функций нескольких переменных.

Во второй главе описывается математический аппарат эвристических алгоритмов основанных на эволюционных вычислениях. Здесь также приводится описание предложенного варианта генетического алгоритма.

В третьей главе описывается программное обеспечения для тестирования различных конфигураций генетического алгоритма на функциях Растрыгина, Швефеля, Грайвенка (задача поиска глобальных минимумов).

Выводы по проделанной работе представлены в заключении.

В работе использовано 4 таблицы, 15 рисунков, список литературы содержит 20 литературных источников. Общий объем выпускной квалификационной работы составляет 41 страниц.

ABSTRACT

The subject of the bachelor's work is "Comparative analysis of heuristic algorithms for minimizing functions of several variables."

In this bachelor's work, methods of searching for extremal functions from several variants using genetic algorithms are investigated.

The paper proposes several variants of genetic algorithms that are investigated using the functions of Rastrygin, Swedefel, Griyvenka. To visualize the extremum search process, the developers of the software itself are used. For the visualization the process of finding extrema function developed its own software

The structure of the bachelor's work is presented by the introduction, three chapters, conclusion, list of references.

The introduction describes the relevance of the study, provides a brief description of the work done.

The first chapter reviews the heuristic algorithms for searching for extremums of functions of several variables.

The second chapter describes the mathematical apparatus of the heuristic algorithms based on evolutionary computations. Here is also a description of the proposed variant of the genetic algorithm.

The third chapter describes the software for the implementation of the proposed genetic algorithm and the results of testing the algorithm on the functions of Rastrygin, Schwefel, Griyvenok.

The conclusions of the done work are presented in the conclusion.

The work contains 4 tables and 15 figures, the list of references contains 20 literary sources. The total volume of final qualifying work is 41 pages.

ОГЛАВЛЕНИЕ

ВВЕДЕНИЕ

Данное исследование посвящено изучению эвристических алгоритмов оптимизации, основанных на эволюционных вычислениях.

Решение задач оптимизации заключается в нахождении входных параметров, при которых функция принимает либо максимальное, либо минимальное значение. В задачах оптимизации данную исследуемую функцию называют целевой.

Для решения задач оптимизации существует множество математических методов, которые можно разделить 2 группы:

- градиентные методы, основанные на понятии производной;
- стохастические методы (в качестве примера можно привести метод Монте-Карло).

Данные методы позволяют находить экстремумы функции, но не всегда можно быть уверенным, что они глобальные. Нахождение локального решения вместо глобального называется проблемой преждевременной сходимости. При использовании точных методов оптимизации из-за сложности математического аппарата резко возрастают временные затраты на поиск решения.

Для преодоления данных недостатков методов оптимизации исследователи ищут пути по совершенствованию старых и поиску новых подходов к решению оптимизационных задач.

Одним из новых подходов для решения оптимизационных задач, предложенных Джонном Холландом, является метаэвристический генетический алгоритм. Генетические алгоритмы основаны на принципах естественного отбора Дарвина и относятся к стохастическим методам оптимизации.

Используемые в генетическом алгоритме подходы стремительно развиваются: появляются новые вариации операторов скрещивания, отбора особей в новую популяцию, отбора родителей и т.д. При этом все

существующие операторы могут сочетаться в разных комбинациях, порождая все новые и новые конфигурации генетических алгоритмов. Также у каждого оператора есть свои параметры, например у оператора мутации – вероятность мутации и т.д. Это еще больше увеличивает количество возможных конфигураций генетического алгоритма.

Поэтому актуальной задачей является разработка программного средства для сравнения различных конфигураций генетического алгоритма.

Цель данной работы – разработать программное обеспечение для сравнения разных конфигураций генетического алгоритма при решении задачи оптимизации функции нескольких переменных.

Поставленная цель достигается путем решения следующих задач:

- анализ принципов работы генетического алгоритма;

- выбор тестовых функций, на которых будет оцениваться работа различных конфигураций генетического алгоритма;

- разработка программного обеспечения для визуализации процесса поиска глобальных экстремумов функции;

- тестирование разработанного программного обеспечения.

1 АНАЛИЗ СОСТОЯНИЯ ВОПРОСА

К задачам оптимизации относится поиск экстремумов (максимумов и минимумов) заданной функции, которую в теории оптимизации называют целевой.

Практически всегда целевая функция представляет собой сложную зависимость от набора входных параметров. Решением задачи оптимизации называют набор входных параметров, удовлетворяющих условиям задачи, при которых функция целевая функция достигает своего максимального или минимального значения.

Таким образом, основными элементами задачи оптимизации являются

- целевая функция;
- управляющие (входные) параметры целевой функции;
- ограничения задачи оптимизации, накладываемые на значения целевой функции и входных параметров.

Ограничения задачи оптимизации бывают различных видов.

1. Общие ограничения. К ним относятся ограничения, накладываемые на значения входных параметров целевой функции, например (1.1):

$$x \geq 0, \text{ где } x = (x_1, ..., x_n). \quad (1.1)$$

2. Специальные ограничения. К ним относятся ограничения, накладываемые на значения функций, которые зависят от входных параметров, например (1.2):

$$g_i(x) \leq 0, \text{ где } x = (x_1, ..., x_n), \quad (1.2)$$

где g_i – i-ая функция, имеющая ограничения, i – индекс для нумерации ограничений.

3. Ограничения типа неравенств. К ним относятся ограничения, накладываемые на значения целевой функции f, например (1.3):

$$f_i(x) \geq 0, \text{ где } x = (x_1, ..., x_n), \quad (1.3)$$

где i – индекс для нумерации ограничений.

4. Ограничения типа равенств. Пример представлен в (1.4):

$$f_i(x) = 0, \text{ где } x = (x_1, ..., x_n), \quad (1.4)$$

где i – индекс для нумерации ограничений.

5. Активные ограничения. К ним относятся ограничения, задающие границы для решения, пример представлен на 1.5:

$$x_{i\min} \le x_i \le x_{i\max}. \quad (1.5)$$

Набор рассмотренных выше ограничений при решении конкретной задачи оптимизации формирует допустимую область, которая может обладать разными свойствами.

Для того чтобы решить оптимизационную задачу, необходимо найти оптимальное решение из допустимой области. При этом решение, представляющее из себя набор значений x_i (1.6), называется допустимым решением, если оно удовлетворяет всем ограничениям задачи:

$$x = (x_1, ..., x_n). \quad (1.6)$$

Задача оптимизации является допустимой лишь в том случае, если множество допустимых решений является непустым, т.е. существует по меньшей мере одно допустимое решение, в противном случае задача оптимизации является недопустимой.

Для того чтобы решение x являлось оптимальным решением задачи оптимизации, оно должно удовлетворять ниже перечисленным условиям.

1. Решение должно являться допустимым.
2. При данном решении значение целевой функции должно достигать глобального экстремума.

При решении задач с использованием стохастических методов могут применяться следующие критерии для остановки поиска оптимального решения:

- изменение значений $x_1, ..., x_n$ за последние несколько итераций работы алгоритма меньше заданного порогового значения;

- изменение значения целевой функции за последние несколько итераций работы алгоритма меньше заданного порогового значения.

Таким образом, для того чтобы некоторую задачу можно было представить в виде задачи оптимизации, необходимо определить ее основные элементы: целевая функция, параметры и ограничения.

Так как генетический алгоритм основан на принципах теории Дарвина, в нем применяются термины, близкие к понятиям принципов эволюции, но несущие в себе математический смысл. Рассмотрим основные понятия генетических алгоритмов.

- Функция приспособленности $f_{\text{пр}}$ – целевая функция решаемой с помощью генетического алгоритма задачи оптимизации ($f_{np} \to \min$ или $f_{np} \to \max$).
- Особь O_i – набор входных параметров функции приспособленности, один из вариантов решения задачи оптимизации:

$$O_i = (x_1, ..., x_n), \tag{1.7}$$

где n – количество входных параметров задачи оптимизации.

- Хромосома – вектор параметров, состоящий или из вещественных генов или из бинарных генов.
- Вещественный ген – значение одного из параметров, формирующих вариант решения задачи оптимизации (другими словами, один из параметров особи). В этом случае особь O_i состоит из генов в количестве n: $x_1, ..., x_n$.
- Бинарный ген – бит закодированного в двоичном виде одного из параметров варианта решения задачи оптимизации (другими словами, бит закодированного в двоичном виде одного из параметров особи). В этом случае гены особи O_i будут представлены бинарной строкой, полученной путем конвертирования значений $x_1, ..., x_n$ в двоичный вид и объединения полученных подстрок. Длина двоичной подстроки для каждого параметра x_i

зависит от требуемой точности поиска генетическим алгоритмом решения поставленной оптимизационной задачи (чем выше точность требуется, тем длинней будет двоичная строка).

- Приспособленность – значение целевой функции для рассматриваемой особи O_i:

$$f_{пр}(O_i) = f_{пр}(x_1,...,x_n). \quad (1.8)$$

- Популяция – набор особей на текущей итерации генетического алгоритма, несколько вариантов решения задачи оптимизации:

$$Pop(t) = (O_1,...,O_N), \quad (1.9)$$

где t – номер итерации выполнения генетического алгоритма, N – размер популяции, который остается постоянным в начале каждой итерации генетического алгоритма.

Принцип работы генетического алгоритма заключается в стохастическом поиске решения задачи оптимизации. Причем процесс поиска носит итерационный характер [1-5]. На каждой следующей итерации своей работы генетический алгоритм производит отсеивание нежелательных (слабых) решений, а на основе лучших решений путем их комбинации алгоритм стремится получить новые решения с более высокими показателями приспособленности.

Все конфигурации генетических алгоритмов работают на основе последовательного выполнения следующих шагов [6-8].

На первом этапе производится генерация начальной популяции особей. При этом случайным образом генерируется набор (заданной размерности) вариантов решения задачи оптимизации [9-11].

На втором этапе осуществляется проверка достижения критерия остановки поиска решения задачи оптимизации. В качестве такого критерия можно использовать один из следующих вариантов [12-14].

- Все решения текущей популяции находятся в области экстремума, и новые итерации не позволяют сгенерировать решения с большей приспособленностью.

- На протяжении нескольких итераций в популяции не изменяется особь с наибольшей приспособленностью. Это означает, что текущая конфигурация генетического алгоритма выбрала оптимальное решение и не может его улучшить.

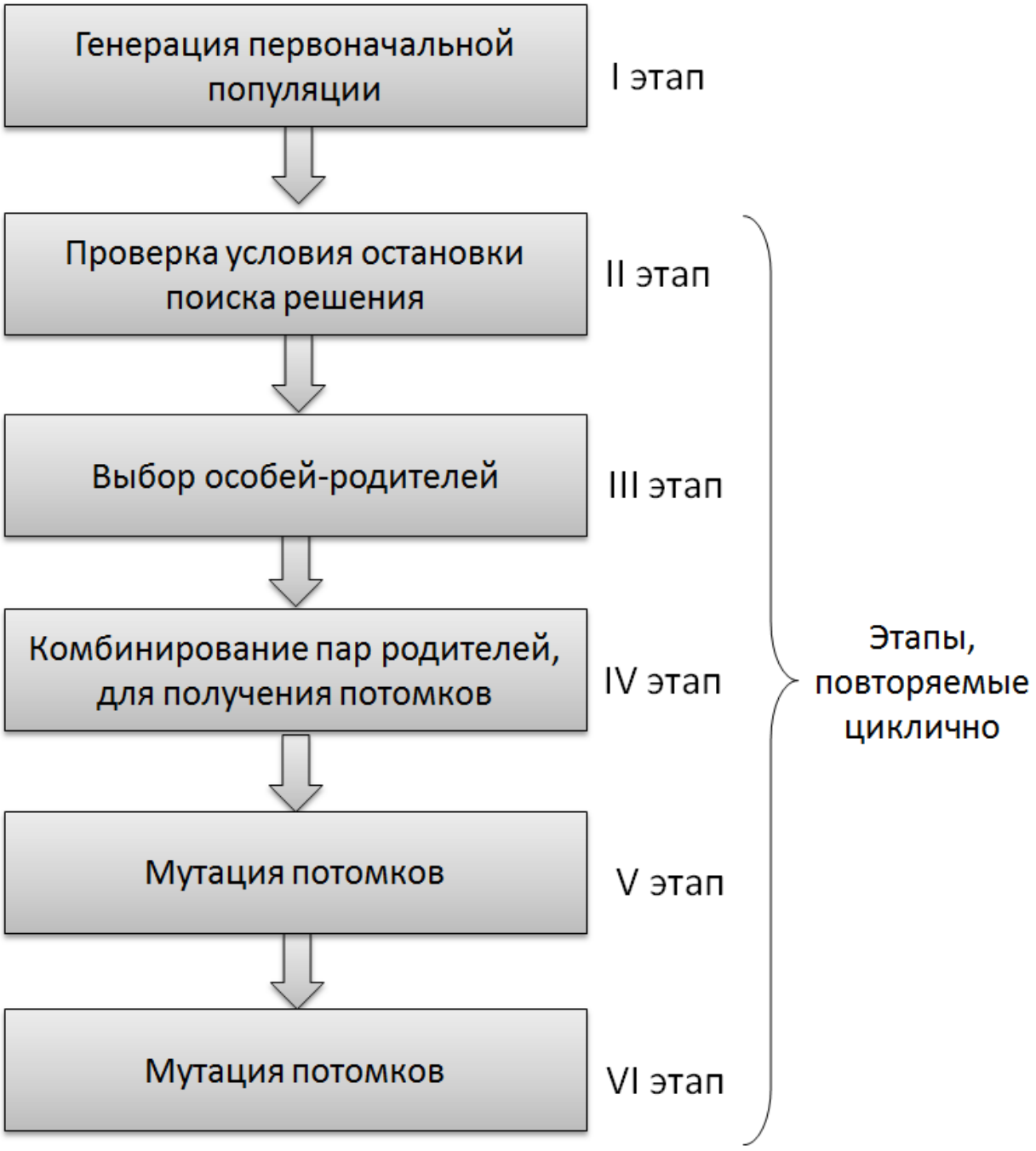


Рисунок 1.1 – Основные этапы генетического алгоритма

Если критерий поиска оптимального решения не достигнут, то к текущей популяции последовательно применяются операторы выбора родителей для скрещивания, скрещивания, мутации и отбоар особей в новую популяцию.

Общий вид генетических алгоритмов представлен на рисунке 1.1.

Генетический алгоритм является метаэвристическим, это значит, что он является основой для создания конкретных конфигураций эвристических алгоритмов. Дело в том, что существует большое количество вариантов реализации оператора выбора родителей (третий этап), оператора скрещивания (четвертый этап), оператора мутации (пятый этап) и оператора отбора особей (шестой этап). При этом у каждого варианта реализации еще существует и большое количество параметров, влияющих на процесс выполнения используемого оператора [15-18].

В связи с этим под конкретной конфигурацией генетического алгоритма принято понимать набор используемых в текущей реализации операторов и конкретные значение их параметров.

2 ВЫБОР КОНФИГУРАЦИИ ГЕНЕТИЧЕСКОГО АЛГОРИТМА

2.1 Описание конфигурации генетического алгоритма

Выберем операторы генетического алгоритма, которые будут использоваться при поиске экстремумов функции нескольких переменных.

Особью O_i в рамках решения задачи оптимизации будет являться вектор входных параметров исследуемой функции.

Если предположить, что у функции n входных параметров, то особь будет представлять собой один набор данных вида

$$O = (x_1, x_2, ..., x_n). \quad (2.1)$$

В одной популяции обычно содержится несколько решений задачи оптимизации функции. Если, например, размер популяции составляет N особей, то популяция Pop на итерации t будет вектором (2.2):

$$Pop(t) = (O_1, O_2, ..., O_N). \quad (2.2)$$

Если требуется найти глобальный минимум функции, то решаемая задача с учетом введённых обозначений будет выглядеть следующим образом:

$$f(x_1, ..., x_n) \to \min. \quad (2.3)$$

На первом этапе работы генетического алгоритма случайным образом генерируются N вариантов решений задачи оптимизации. На следующих этапах работы генетического алгоритма данные решения будут уточняться до тех пор, по не будет найдено глобальное решение задачи оптимизации.

Так как генерация первоначальной популяции происходит случайным образом, то ее можно описать следующим образом (2.4):

$$i = 1...N$$
$$\begin{cases} O_i = (x_1, ..., x_n) \\ x_1 = random(\min(x_1), \max(x_1)) \\ x_2 = random(\min(x_2), \max(x_2)) \\ ... \\ x_n = random(\min(x_n), \max(x_n)) \end{cases}, \quad (2.4)$$

где random(*A*, *B*) – функция, генерирующая случайное число в диапазоне от *A* до *B*; min(*x*) – функция, возвращающая минимальное значение области определения переменной *x*; max(*x*) – функция, возвращающая максимальное значение области определения переменной *x*.

На втором этапе выполнения генетического алгоритма производится проверка, достигнуто ли условие остановки поиска оптимального решения. Так как в процессе своей работы генетический алгоритм корректирует решения, содержащиеся в популяции, то предложено выполнять поиск до тех пор, пока все решения не локализуются в одной окрестности.

Математически это условие можно описать следующим образом:

$$\frac{\sum_{i=1}^{N} f(O_i)}{N} \le T, \quad (2.5)$$

где *f*() – целевая функция, O_i – *i*-тая особь текущей популяции, *N* – размер популяции, *T* – параметр, задающий размер окрестности.

Таким образом, если условие (2.5) выполнено, то следует остановить поиск решения.

Для того чтобы искать новые решения задачи оптимизации, необходимо на основе двух известных решений путем их комбинирования сгенерировать новые решения. Те особи, которые участвуют в генерировании новых решений задачи оптимизации, называются родителями. А те особи, которые получаются в результате комбинирования, называются потомками.

На этапе выбора родителей (III этап) предлагается применить наиболее часто используемый оператор – панмиксия. Он предполагает, что особи, которые могут стать родителями, выбираются случайным образом.

Математически панмиксию можно описать следующим образом:

$$\begin{cases} Pf = random(O_1,...,O_N) \\ Ps = random(O_1,...,O_N) \end{cases}, \quad (2.6)$$

где Pf и Ps – первый (first parent) и второй (second parent) родители соответственно, $O_1,...,O_N$ - особи текущей популяции.

Для получения особей-потомков не достаточно просто выбрать пару родителей, необходимо дополнительно определить схему комбинирования их генов. Так как мы рассматриваем задачу оптимизации функции нескольких переменных, то геном будет являться вещественное значение одной из переменных. Т.е. если значение целевой функции зависит от переменных $x_1,...,x_n$, то особь (вариант решения) будет состоять из n генов (значений этих переменных).

В более сложных случаях для комбинации генов может потребоваться их представление в бинарном виде, что накладывает дополнительные ограничения на точность поиска экстремума функции. В нашем случае этого не требуется, так как мы работаем с вещественными генами.

На четвертом этапе предлагается использовать стратегию под названием линейная рекомбинация.

При использовании данного оператора скрещивания сначала генерируется множитель рекомбинации α, который выбирается случайным образом на участке от [-0,25; 1,25]. Данный множитель генерируется один раз для каждого потомка. Затем формируется каждый i-ый ген особи-потомка D на основе генов родителей. Математически линейную рекомбинацию можно представить следующим образом:

$$i = 1...n$$
$$\begin{cases} \alpha = random(-0,25;1,25), \\ D_i = Pf_i + \alpha \cdot (Ps_i - Pf_i) \end{cases} \quad (2.7)$$

где random(*A*, *B*) – функция, генерирующая случайное число в диапазоне от *A* до *B*.

Для того чтобы защитить генетический алгоритм от попадания в локальный экстремум, при решении задачи оптимизации функции применяется мутация.

Идея мутации заключается в случайном изменении генов особей-потомков. Вместе с тем лишь малая часть особей должна участвовать в мутации для того, чтобы не нарушать процесс поиска решения. Вероятность P_m мутации особи при выполнении генетического алгоритма задается, как правило, намного меньше единицы ($P_m << 1$). В нашем случае вероятность мутации равна 10%.

Идея мутация вещественных генов заключается в выборе знака и величины изменения гены. Очевидно, что вероятность выбора знака должна быть одинаковой. При этом величину изменения гена необходимо ограничить, так как мутация не должна приводить к сильным изменениям особи. На практике величину δ изменения гена ограничивают половиной размера области изменения входного параметра, с которым данный ген связан.

Таким образом, мутация генов потомка *D* осуществляется так, как это показано в (2.8):

$$\begin{aligned} & j = 1...n \\ & \delta_j \leftarrow \text{random}([0.5 \cdot \min(x_j)], [0.5 \cdot \max(x_j)]) \\ & \alpha_j \leftarrow \text{random}(0,1) \\ & D_j = \begin{cases} D_j - \delta \ \ if \ \ \alpha_j \le 0.5 \\ D_j + \delta \ \ if \ \ \alpha_j > 0.5 \end{cases} \end{aligned} \quad , \quad (2.8)$$

где $\min(x_j)$ и $\max(x_j)$ - минимальное и максимальное значения области определения параметра x_j.

В результате комбинирования родителей получаются новые особи-потомки, но при этом особи-родители также остаются в популяции. Это приводит к тому, что на последнем этапе генетического алгоритма размер популяции *Pop* превышает начальное значение *N*. Перед началом выполнения следующей итерации генетического алгоритма в новую популяцию *Pop_new* отбираются только лучшие с точки зрения решаемой задачи ($f(x_1,...,x_n) \to \min$) особи.

На последнем этапе выполнения генетического алгоритма выполняется сортировка текущей популяции таким образом, чтобы меньшие значения индексов соответствовали лучшим решениям. Так мы получаем отсортированную популяцию *Pop_sort*.

Далее особи из популяции *Pop_sort* переносятся в новую популяцию *Pop_new* до тех пор, пока последняя не достигнет первоначального размера. Математически это можно записать так (2.9):

$$\begin{cases} i = 1...N \\ Pop\_new_i = Pop\_sort_i \end{cases}, \quad (2.9)$$

где *N* – размер популяции, *Pop_sort* – отсортированная текущая популяция размера >N, *Pop_new* – новая популяция размера *N*, *i* – индекс особи.

2.2 Функции для тестирования работы генетических алгоритмов

Генетический алгоритм является метаэвристическим. Это означает, что под генетическим понимают большое множество алгоритмов, формируемых путем различного сочетания операторов отбора родителей, скрещивания, мутации, отбора особей в новую популяцию.

Для изучения свойств любой конфигурации генетического алгоритма можно использовать одну или несколько тестовых функций, которые будут представлены ниже [19-20]. Каждая функция обладает своими особенностями: в некоторых из них присутствует несколько точек глобальных экстремумов; в других – большое количество локальных экстремумов, затрудняющих поиск глобального минимума (или максимума); в других функциях рядом находятся области, в которых значение функции практически не меняется и области с резким изменением значения целевой функции.

Рассмотрим наиболее известные тестовые функции.

- Сферическая функция (2.10):

$$f(x_1,...,x_n)=\sum_{i=1}^{n}x_i^2 . \qquad (2.10)$$

При тестировании генетического алгоритма на данной функции решают задачу поиска глобального минимума, который достигается при $x_i=0\,(i=1...n)$. Область изменения входных параметров при тестировании ограничена диапазоном $x\in(-5{,}12;5{,}12)$. Минимальное значение функции $f(x_1,\ldots,x_n)$ равно 0. Локальные минимумы отсутствуют.

- Гиперэллипсоидная функция с параллельными осями (2.11):

$$f(x_1,...,x_n)=\sum_{i=1}^{n}ix_i^2 \ . \qquad (2.11)$$

Данную функцию используют для проверки у генетического алгоритма способности осуществлять поиск глобального минимума, который достигается при $x_i=0\,(i=1...n)$. Область изменения входных параметров при тестировании ограничена диапазоном $x\in(-5{,}12;5{,}12)$. Минимальное значение функции $f(x_1,\ldots,x_n)$ равно 0. Данная функция является выпуклой и унимодальной.

- Повернутая гиперэллипсоидная функция (2.12):

$$f(x_1,...,x_n)=\sum_{i=1}^{n}\sum_{j=1}^{i}x_j^2\ . \qquad (2.12)$$

Данную функцию используют для тестирования конфигурации генетического алгоритма на задаче поиска глобального минимума, который достигается при $x_i=0\ (i=1...n)$. Область изменения входных параметров при тестировании ограничена диапазоном $x\in(-65,536;65,536)$. Минимальное значение функции $f(x_1,\ldots,x_n)$ равно 0. Эта функция получена поворотом гиперэллипсоидной функции, поэтому она является выпуклой и унимодальной.

- Ступенчатая функция (2.13):

$$f(x_1,...,x_n)=\sum_{i=1}^{n}\left|round(x_i)\right|\ , \qquad (2.13)$$

где round() – функция, округляющее число до целой части.

При тестировании генетического алгоритма на данной функции решают задачу поиска глобального минимума, который достигается при $x_i=0\ (i=1...n)$. Область изменения входных параметров при тестировании ограничена диапазоном $x\in(-5,12;5,12)$. Минимальное значение функции $f(x_1,\ldots,x_n)$ равно 0.

- Седло Розенброка (2.14) :

$$f(x_1,...,x_n)=\sum_{i=1}^{n-1}(100(x_{i+1}-x_i^2)^2+(x_i-1)^2)\ . \qquad (2.14)$$

Данную функцию используют для проверки у генетического алгоритма способности осуществлять поиск глобального минимума, который достигается при $x_i=0\ (i=1...n)$. Область изменения входных параметров при тестировании ограничена диапазоном $x\in(-2,048;2,048)$. Минимальное значение функции $f(x_1,\ldots,x_n)$ равно 0. Примечательно, что глобальный

минимум находится внутри параболической сильно вытянутой поверхности, что осложняет задачу по его поиску.

- Функция с гауссовским распределением случайной величины (2.15):

$$f(x_1,...,x_n) = \sum_{i=1}^{n-1}(x_i^4 + gauss(0,1)), \quad (2.15)$$

где *gauss*() – функция, генерирующая случайное число в соответствии с законом нормального распределения, первый параметр функции – математическое ожидание, второй – дисперсия.

Область изменения входных параметров при тестировании генетического алгоритма ограничена диапазоном $x \in (-1,28;1,28)$.

- Функция Растрыгина (2.16):

$$f(x_1,...,x_n) = 10n + \sum_{i=1}^{n}(x_i^2 - 10\cos(2\pi x_i)). \quad (2.16)$$

Функция Растрыгина используется для тестирования конфигурации генетического на задаче поиска глобального минимума, который достигается при $x_i = 0\,(i = 1...n)$. Область изменения входных параметров при тестировании ограничена диапазоном $x \in (-5,12;5,12)$ Минимальное значение функции $f(x_1,\ldots,x_n)$ равно 0. Примечательно, что данная функция имеет большое количество локальных минимумов, осложняющих работу генетического алгоритма.

- Функция Швефеля (2.17):

$$f(x_1,...,x_n) = 418,9829n + \sum_{i=1}^{n}(-x_i \cdot \sin\sqrt{|x_i|}). \quad (2.17)$$

При тестировании генетического алгоритма на данной функции решают задачу поиска глобального минимума, который достигается при $x_i = 420,9829\,(i = 1...n)$. Область изменения входных параметров при тестировании ограничена диапазоном $x \in (-500;50)$. Минимальное значение функции $f(x_1,\ldots,x_n)$ равно 0. Примечательно, что данная функция имеет

дополнительный локальный минимум, осложняющий работу генетического алгоритма.

- Функция Грайвенка (2.18):

$$f(x_1,...,x_n) = 1 + \sum_{i=1}^{n} \frac{x_i^2}{4000} - \prod_{i=1}^{n} \cos\left(\frac{x_i}{\sqrt{i}}\right). \quad (2.18)$$

Данную функцию используют для проверки у генетического алгоритма способности осуществлять поиск глобального минимума, который достигается при $x_i = 0\,(i = 1...n)$. Область изменения входных параметров при тестировании ограничена диапазоном $x \in (-600;600)$. Минимальное значение функции $f(x_1,\ldots,x_n)$ равно 0. Примечательно, что у функции есть большое количество распределенных локальных минимумов, осложняющих работу генетического алгоритма.

- Функция Эккли (2.19):

$$f(x_1,...,x_n) = 20 + e - 20 \cdot \exp\left(-0,2\sqrt{\frac{1}{n}\sum_{i=1}^{n} x_i^2}\right) - \exp\left(\frac{1}{n}\sum_{i=1}^{n} \cos(2\pi x_i)\right). \quad (2.19)$$

Функцию Эккли используют для тестирования конфигурации генетического алгоритма на задаче поиска глобального минимума, который достигается при $x_i = 0\,(i = 1...n)$. Область изменения входных параметров при тестировании ограничена диапазоном $x \in (-30;30)$ Минимальное значение функции $f(x_1,\ldots,x_n)$ равно 0.

- Функция суммы различных степеней (2.20):

$$f(x_1,...,x_n) = \sum_{i=1}^{n} |x_i|^{i+1}. \quad (2.20)$$

При тестировании генетического алгоритма на данной функции решают задачу поиска глобального минимума, который достигается при $x_i = 0\,(i = 1...n)$. Область изменения входных параметров при тестировании

ограничена диапазоном $x \in (-1;1)$. Минимальное значение функции $f(x_1,\ldots,x_n)$ равно 0.

- Функция Михалевича (2.21):

$$f(x_1,...,x_n) = -\sum_{i=1}^{n}\left(\sin(x_i)\cdot\sin^{20}\left(\frac{i}{\pi}x_i^2\right)\right). \qquad (2.21)$$

Данную функцию используют для проверки у генетического алгоритма способности осуществлять поиск глобального минимума. Минимальное значение функции при n=5: $f(x_1,\ldots,x_n)$=–4,69, при n=10: $f(x_1,\ldots,x_n)$=–9,66. Область изменения входных параметров при тестировании ограничена диапазоном $x \in [0;\pi]$. Примечательно, что рельеф функции представлен набором оврагов и плато, что осложняет работу генетического алгоритма.

- Функция Бранинса (2.22):

$$f(x_1,x_2) = \left(x_2 - \frac{5{,}1}{4\pi^2}x_1^2 + \frac{5}{\pi}x_1 - 6\right)^2 + 10\left(1-\frac{1}{8\pi}\right)\cos(x_1) + 10. \qquad (2.22)$$

Функцию Бранинса используют для тестирования конфигурации генетического на задаче поиска нескольких точек глобального минимума, которыми являются $(x_1,x_2) = (-\pi, 12.275)$, $(x_1,x_2) = (\pi, 2.275)$, $(x_1,x_2) = (9.42478, 2.475)$. Область изменения входных параметров при тестировании ограничена диапазоном $x_1 \in [-5;10], x_2 \in [0;15]$. Минимальное значение функции $f(x_1,x_2)$ равно 0,397887. От генетического алгоритма требуется найти все 3 точки глобального минимума.

- Функция Изома (2.23):

$$f(x_1,x_2) = -\cos(x_1)\cos(x_2)\exp\left(-(x_1-\pi)^2 + (x_2-\pi)^2\right). \qquad (2.23)$$

При тестировании генетического алгоритма на данной функции решают задачу поиска глобального минимума, который достигается при $x_1 = -\pi, x_2 = \pi$. Область изменения входных параметров при тестировании

ограничена диапазоном $x_1 \in [-100;100], x_2 \in [-100;100]$. Минимальное значение функции $f(x_1, x_2)$ равно –1.

Примечательно, что у данной унимодальной функции минимум находится в малой области по сравнению размером поискового пространства. Это усложняет работу генетического алгоритма.

- Функция Голдстейна (2.24):

$$f(x_1, x_2) = [1 + (1 + x_1 + x_2)^2 \cdot (19 - 14x_1 + 3x_1^2 - 14x_2 + 6x_1x_2 + 3x_2^2)] \times \\ \times [30 + (2x_1 - 3x_2)^2 (18 - 32x_1 + 12x_1^2 + 48x_2 - 36x_1x_2 + 27x_2^2)] \quad (2.24)$$

Данную функцию используют для проверки у генетического алгоритма способности осуществлять поиск глобального минимума, который достигается при $x_1 = 0, x_2 = -1$. Область изменения входных параметров при тестировании ограничена диапазоном $x_1 \in [-2;2], x_2 \in [-2;2]$. Минимальное значение функции $f(x_1, x_2)$ равно 3.

- Функция шестигорбая (2.25):

$$f(x_1, x_2) = \left(4 - 2{,}1x_1^2 + \frac{x_1^4}{3}\right)x_1^2 + x_1x_2 + (-4 - 4x_2^2) \ . \quad (2.25)$$

Эту функцию используют для тестирования конфигурации генетического алгоритма на задаче поиска нескольких точек глобального минимума, которыми являются точки $(x_1, x_2) = (-0.0898, 0.7126)$ и $(x_1, x_2) = (0,0898, -0,7126)$. Область изменения входных параметров при тестировании ограничена диапазоном $x_1 \in [-3;3], x_2 \in [-2;2]$. Минимальное значение функции $f(x_1,x_2)$ равно –1,0316. От генетического алгоритма требуется найти обе точки глобального минимума.

2.3 Вычислительный эксперимент для тестирования конфигурации генетического алгоритма

Для исследования эффективности описанной выше конфигурации генетического алгоритма необходимо выбрать тестовые функции, в рамках которых будут проводиться вычислительные эксперименты.

В рамках выполнения вычислительного эксперимента планируется определить

- возможность выбранной конфигурации генетического алгоритма находить глобальный экстремум функции в условия наличия большого количества локальных экстремумов;
- количество итераций, требуемых генетическому алгоритму для поиска глобального экстремума функции.

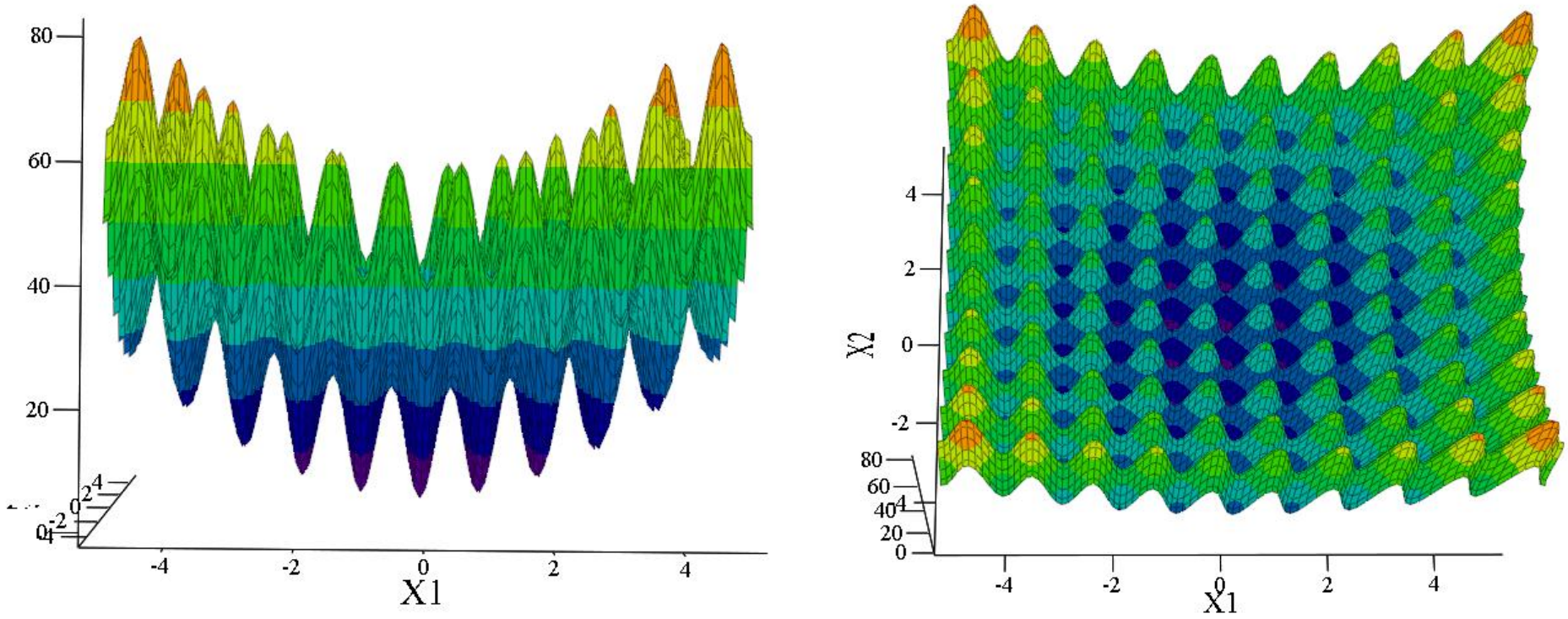
При этом, выбирая функции, необходимо учесть возможность визуализации процесса поиска решений путем отображения текущей популяции в виде набора точек на графике. Поэтому количество переменных функции должно составлять не более двух.

С учетом вышесказанного в качестве тестовых были выбраны следующие функции:

- функция Растрыгина с двумя переменными (рисунок 2.1);
- функция Швефеля с двумя переменными (рисунок 2.2);
- функция Грайвенка с двумя пременными (рисунок 2.3).

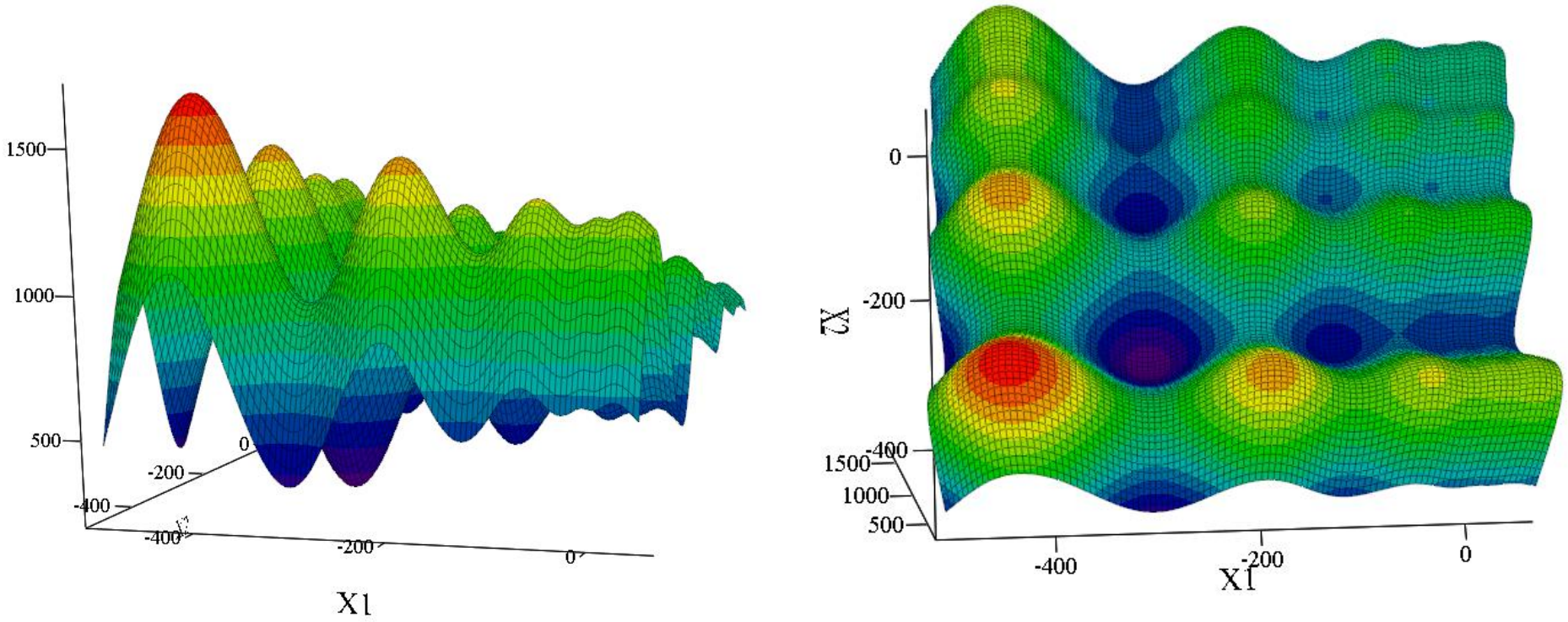
Для проведения вычислительного эксперимента требуется разработать программное обеспечение, моделирующее работу данной конфигурации генетического алгоритма и визуализирующее процесс поиска решения.

Описание разработанного программного обеспечения и результаты вычислительного эксперимента представлены в следующей главе.



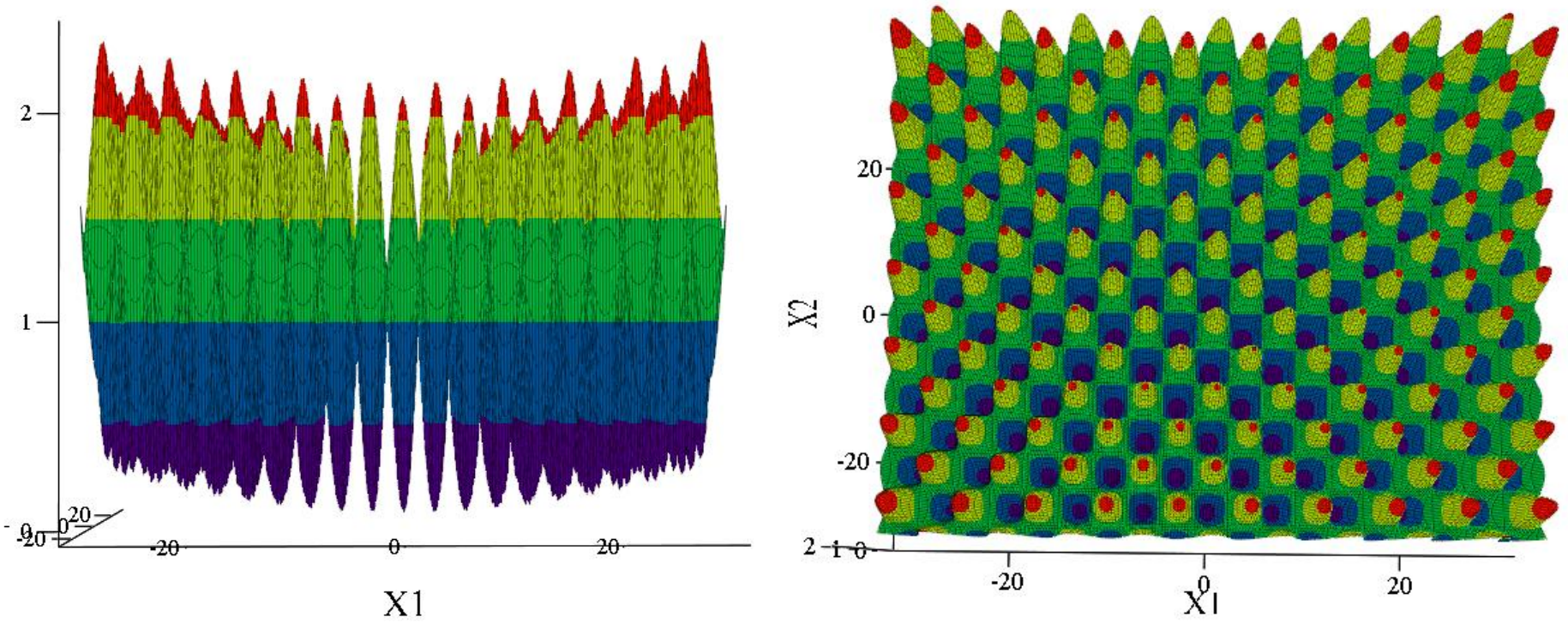
$$f(x_1, x_2) = 20 + (x_1)^2 - 10 \cdot \cos(2\pi \cdot x_1) + (x_2)^2 - 10 \cdot \cos(2\pi \cdot x_2)$$

Рисунок 2.1 – Функция Растрыгина от двух переменных



$$f(x_1, x_2) = 418,9829 \cdot 2 + (-x_1 \cdot \sin(\sqrt{|x_1|})) + (-x_2 \cdot \sin(\sqrt{|x_2|}))$$

Рисунок 2.2 – Функция Швефеля от двух переменных



$$f(x_1, x_2) = 1 + \frac{1}{4000} x_1^2 + \frac{1}{4000} x_2^2 - \cos(x_1)\cos\left(\frac{1}{2} x_2 \sqrt{2}\right)$$

Рисунок 2.3 – Функция Грайвенка от двух переменных

3 ПРОГРАММНОЕ ОБЕСПЕЧЕНИЕ ДЛЯ МОДЕЛИРОВАНИЯ ГЕНЕТИЧЕСКОГО АЛГОРИТМА

3.1 Описание программного обеспечения

В ходе выполнения бакалаврской работы для моделирования работы генетического алгоритма в командной оболочке для интерактивных вычислений Jupyter Notebook на языке программирования Python было разработано программное обеспечение, обладающее следующим функционалом:

- визуализация процесса поиска решения генетическим алгоритмом путем отображения особей текущей популяции на трехмерном графике исследуемой функции;

- программное моделирование работы конфигурации генетического алгоритма, включающее в себя случайное генерирование решений на первой итерации, остановку поиска решений при локализации особей в отдельной окрестности, использование панмиксии в качестве оператора выбора родителей, получение новых решений задачи оптимизации за счет вещественной рекомбинации генов, мутацию вещественных генов особей-потомков, элитарный отбор особей в новую популяцию;

- программное задание таких параметров работы генетического алгоритма, как исходный размер популяции, процент лучших особей, способных стать родителями, генерируемое количество пар родителей, вероятность мутации, максимальное отклонение вещественного гена от текущего значения, процент лучших особей, переходящих в новую популяцию.

Пример работы программного обеспечения представлен на рисунке 3.1.

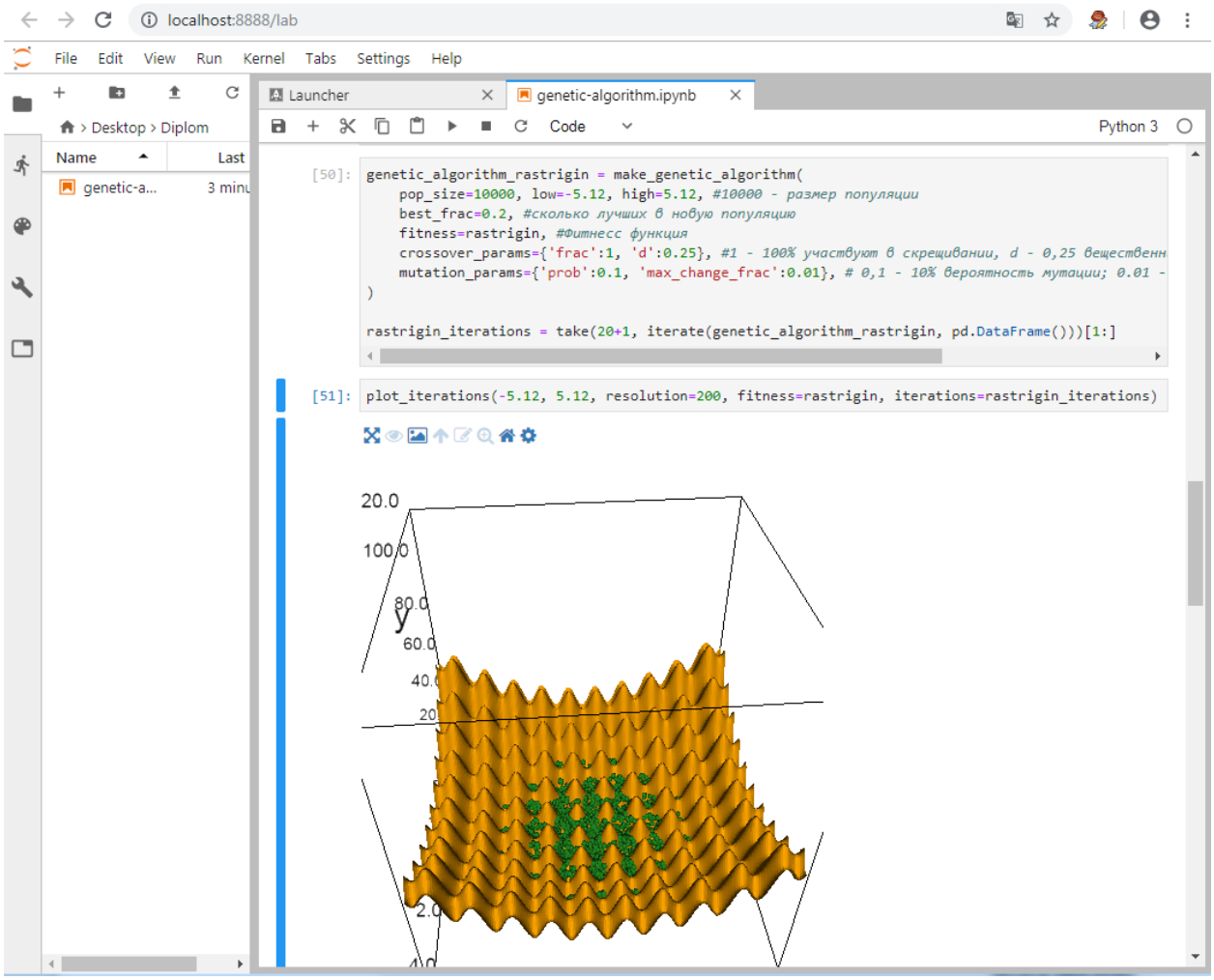


Рисунок 3.1 – Разработанное программное обеспечение для тестирования генетического алгоритма на функциях Растрыгина, Швефеля, Грайвенка

Программный код основных этапов работы генетического алгоритма приведен на рисунках 3.2-3.4.

```
def generate_population(n, low, high):
    res = pd.DataFrame(
        np.random.uniform(low, high, size=(n, 2)),
        columns=['x', 'y']
    )

    return res
```

Рисунок 3.2 – Реализация генерации популяции

```
def crossover(popul, frac, d):
    assert -0.25 <= d <= 1.25
    n = round_to(len(popul) * frac, 2)
    sampled = popul.sample(n, replace=True)
    aa = sampled.head(n // 2)
    bb = sampled.tail(n // 2)

    alpha = lambda: np.random.uniform(-d, 1 + d)

    child = lambda a, b: pd.Series([a.x + alpha() * (b.x - a.x),
                                    a.y + alpha() * (b.y - a.y)],
                                   index=['x', 'y'])

    return pd.DataFrame(
        lmapcat(juxt(child, child),
                aa.itertuples(),
                bb.itertuples()),
    )
```

Рисунок 3.3 – Реализация вещественной рекомбинации

```
def mutate(popul, prob, max_change_frac, low, high):
    def mut(indiv):
        rand_change = lambda: np.random.uniform(-max_change_frac, max_change_frac) * (high - low)
        if np.random.uniform() <= prob:
            return indiv + [rand_change(), rand_change()]
        else:
            return indiv

    return popul.apply(mut, axis=1, result_type='broadcast')
```

Рисунок 3.4 – Реализация мутации вещественных генов

Также программно были заданы функции Растрыгина, Грайвенка и Швефеля, на которых производилось тестирование генетических алгоритмов (рисунки 3.5 – 3.7).

```
def rastrigin(X, Y):
    a = 5
    return (2*a + X**2 - a * np.cos(2 * np.pi * X)
                + Y**2 - a * np.cos(2 * np.pi * Y))
```

Рисунок 3.5 – Задание функции Растрыгина

```
def griewank(X, Y):
    return 1 + X**2 / 4000 + Y**2 / 4000 - np.cos(X) * np.cos(np.sqrt(2) * Y/2)
```

Рисунок 3.6 – Задание функции Грайвенка

```
def schwefel(X, Y):
    return (-X * np.sin(np.sqrt(np.abs(X)))
            -Y * np.sin(np.sqrt(np.abs(Y))))
```

Рисунок 3.7 – Задание функции Швефеля

3.2 Результаты апробации генетического алгоритма на тестовых функциях

Опытным путем были определены параметры генетического алгоритма обеспечивающие нахождение глобальных минимумов функций Растрыгина, Швефеля, Грайвенка за наименьшее количество итераций. Оптимальные параметры выбранной конфигурации генетического алгоритма:

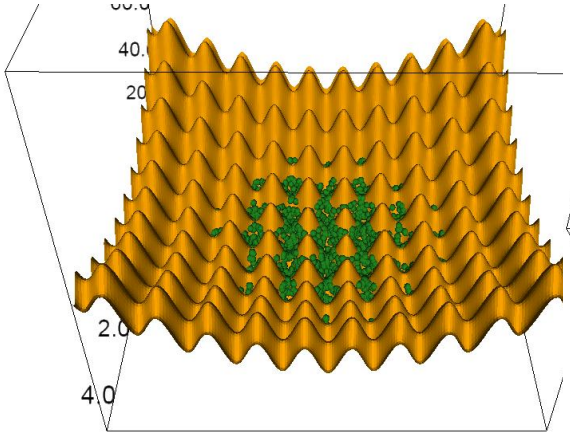
- исходный размер популяции – 1000;
- процент лучших особей, способных стать родителями, – 30%;
- генерируемое количество пар родителей – 500;
- вероятность мутации – 10%;
- максимальное отклонение вещественного гена от текущего значения – 25% области определения рассматриваемой величины;
- процент лучших особей, переходящих в новую популяцию, – 50%.

При тестировании предложенной конфигурации генетического алгоритма на функции Растрыгина с двумя переменными глобальный минимум находится за 12 итераций, на функции Швефеля с двумя переменными – за 10 итераций, на функции Грайвенка – за 42 итерации.

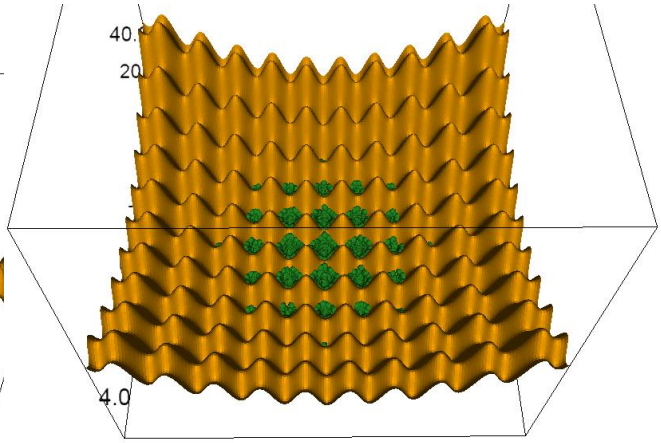
Относительно долгий поиск глобального минимума функции Грайвенка можно объяснить максимальной плотностью скопления локальных минимумов по сравнению с другими рассматриваемыми функциями.

Некоторые итерации работы генетического алгоритма на функции Растрыгина показаны на рисунке 3.8, итерации работы генетического алгоритма на функции Швефеля показаны на рисунке 3.9, а итерации работы генетического алгоритма на функции Грайвенка показаны на рисунке 3.10.

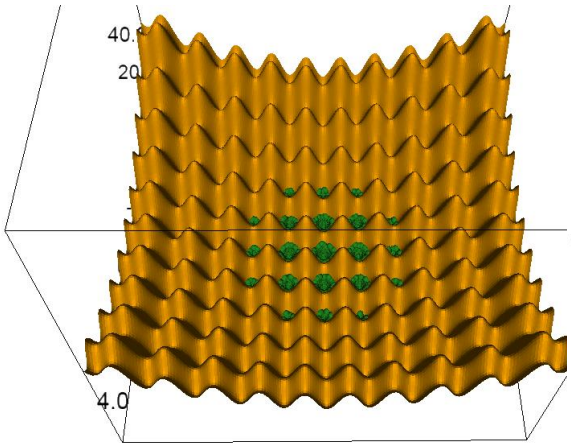
а) Итерация №2 , вид «сверху»



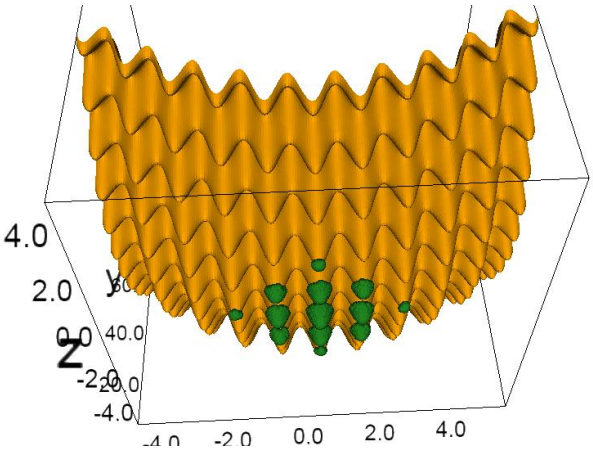
б) Итерация №4, вид «сверху»



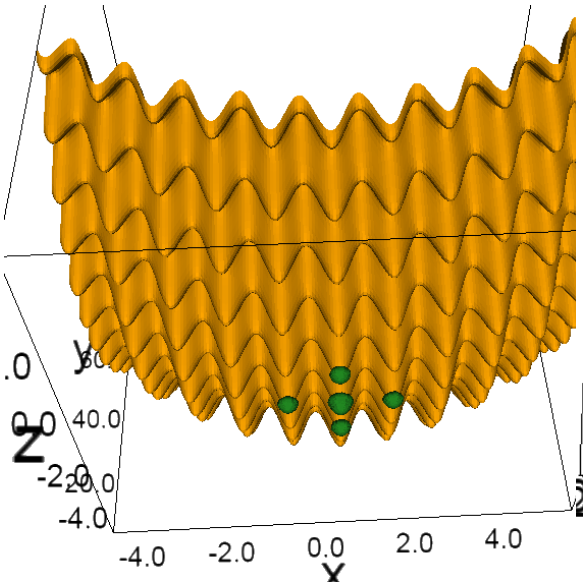
в) Итерация №6, вид «сверху»



г) Итерация №8, вид «снизу»



д) Итерация №10, вид «снизу»



е) Итерация №12, вид «сбоку»

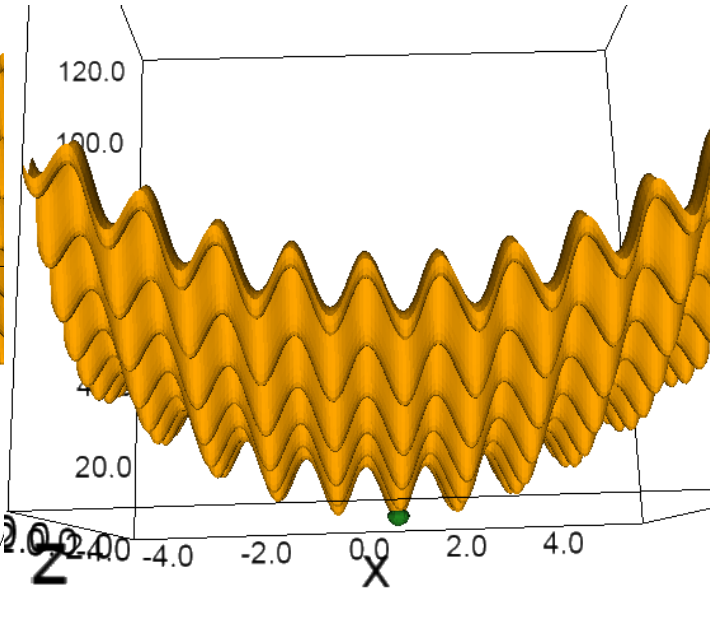
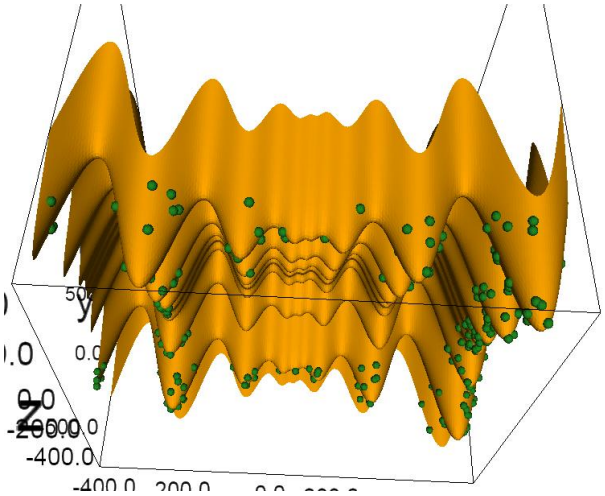
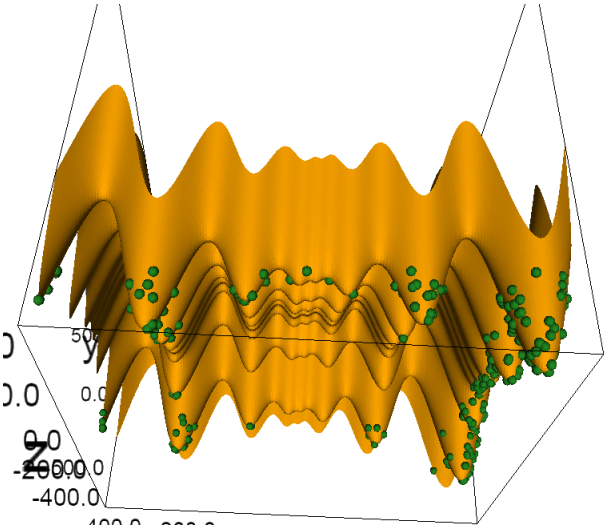


Рисунок 3.8 – Результат работы генетического алгоритма на функции Растрыгина (зеленые точки – найденные решения)

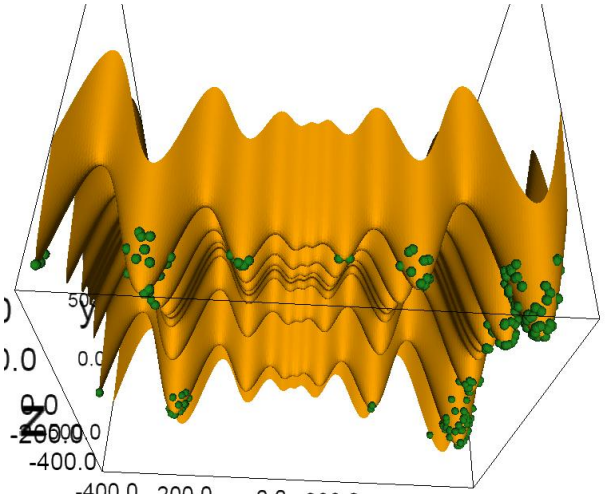
а) Итерация №2 , вид «снизу»



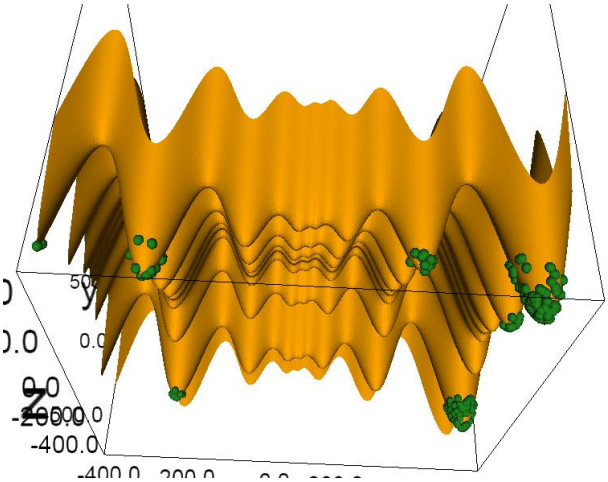
б) Итерация №3 , вид «снизу»



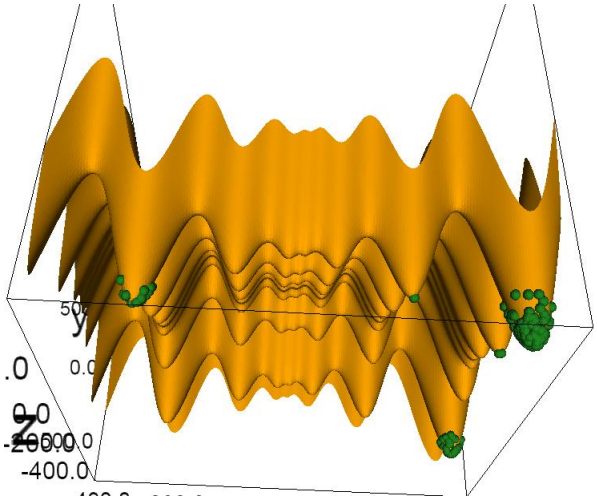
в) Итерация №4 , вид «снизу»



г) Итерация №5 , вид «снизу»



д) Итерация №6 , вид «снизу»



е) Итерация №10 , вид «снизу»

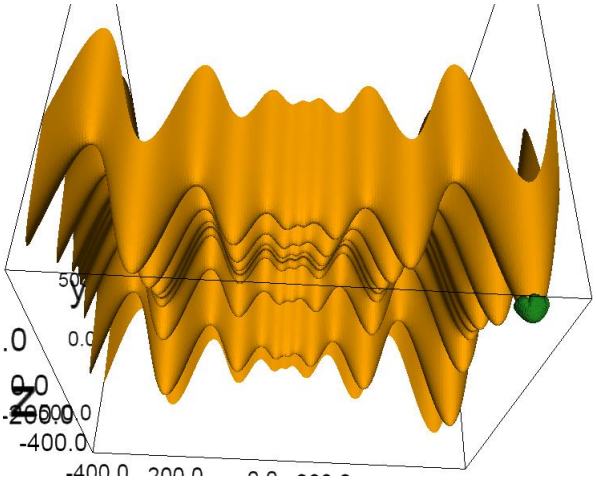
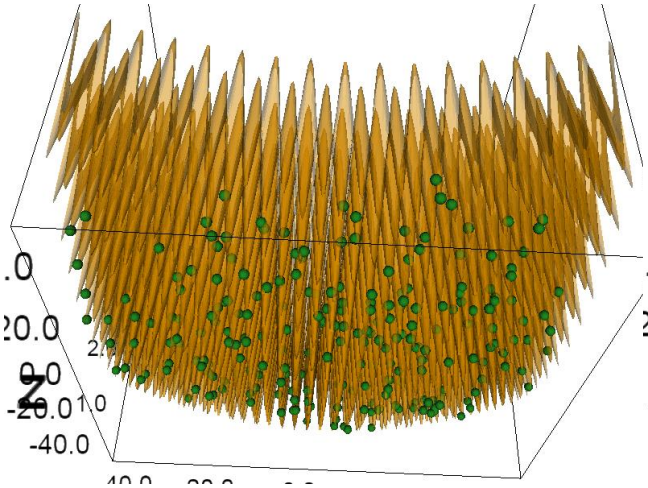
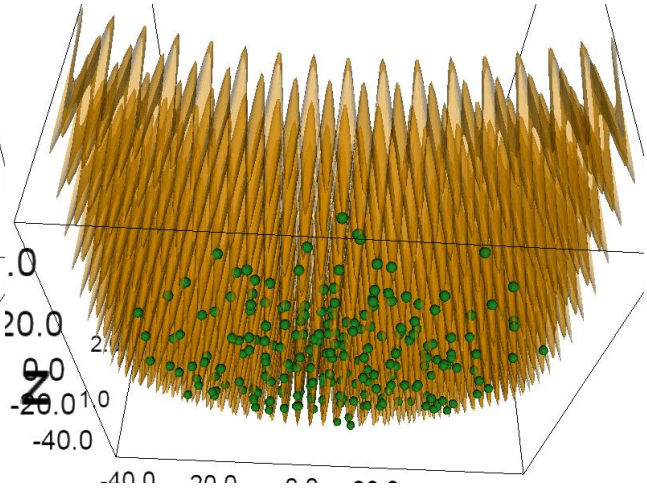


Рисунок 3.9 – Результат работы генетического алгоритма на функции Швефеля (зеленые точки – найденные решения)

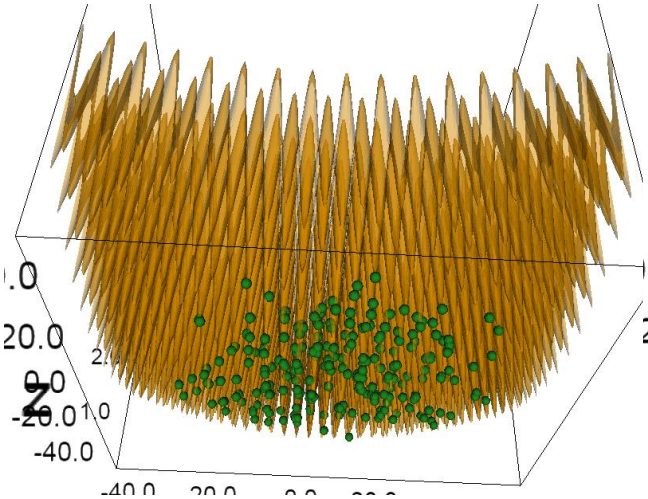
а) Итерация №2, вид



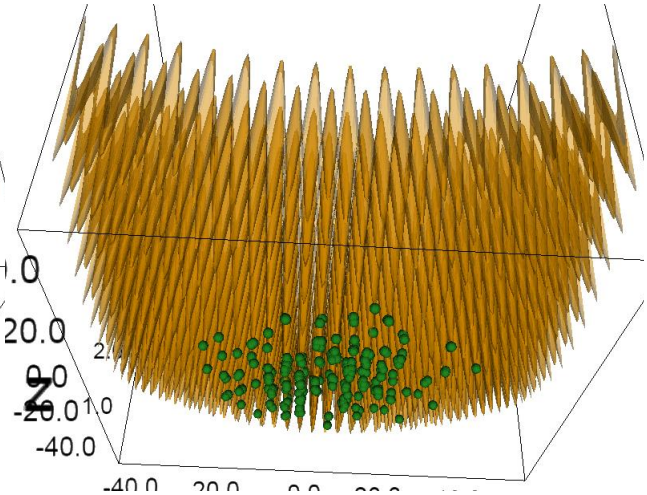
б) Итерация №5



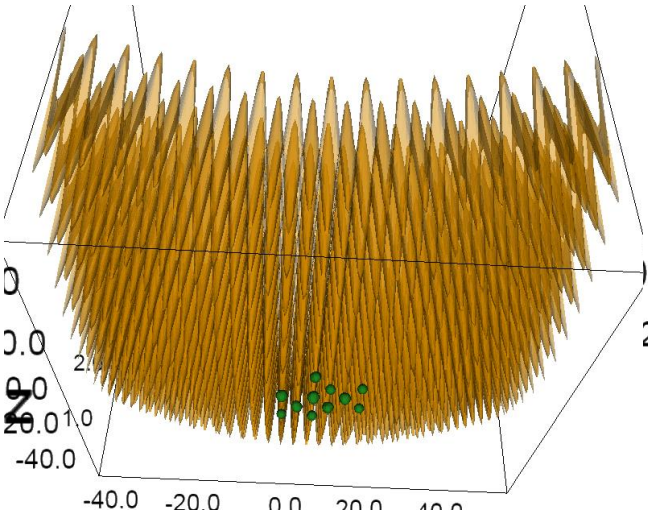
в) Итерация №10



г) Итерация №15



д) Итерация №30



е) Итерация №42

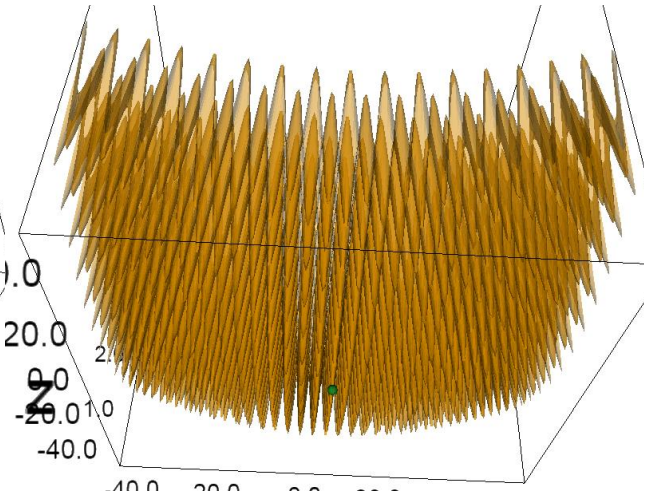


Рисунок 3.10 – Результат работы генетического алгоритма на функции Грайвенка (зеленые точки – найденные решения)

ЗАКЛЮЧЕНИЕ

В ходе выполнения бакалаврской работы были получены следующие результаты.

1. Обзор литературных источников показал, что перспективным методом решения задач оптимизации является генетический алгоритм. Данный алгоритм относится к стохастическим методам поиска решений. Генетический алгоритм является метаэвристическим, это значит, что он является основой для создания конкретных конфигураций эвристических алгоритмов. Поэтому актуальной задачей остается исследование свойств различных конфигураций генетических алгоритмов.

2. В ходе исследования проанализированы принципы действия генетического алгоритма.

3. Предложена конфигурации генетического алгоритма, включающая в себя случайное генерирование решений на первой итерации, остановку поиска решений при локализации особей в отдельной окрестности, использование панмиксии в качестве оператора выбора родителей, получение новых решений задачи оптимизации за счет вещественной рекомбинации генов, мутацию вещественных генов особей-потомков, элитарный отбор особей в новую популяцию.

4. Разработано и протестировано программное обеспечение для визуализации процесса поиска решения генетическим алгоритмом путем отображения особей текущей популяции на трехмерном графике исследуемой функции.

5. Проведен вычислительный эксперимент, направленный на тестирование предложенной конфигурации генетического алгоритма в задачах поиска глобальных минимумов функций Растрыгина, Швефеля и Грайвенка.

6. При тестировании предложенной конфигурации генетического алгоритма на функции Растрыгина от двух переменных глобальный минимум находится за 12 итераций, на функции Швефеля от двух переменных – за 10 итераций, на функции Грайвенка – за 42 итерации.

СПИСОК ИСПОЛЬЗУЕМОЙ ЛИТЕРАТУРЫ

1. Mzili, I. Hybrid Penguins Search Optimization Algorithm and Genetic Algorithm Solving Traveling Salesman Problem [Text] / Ilyass Mzili, Mohammed Essaid Riffi, Fatiha Benzekri // Proceedings of the International Conference on Advanced Information Technology, Services and Systems (AIT2S-17) Held on April 14/15, 2017 in Tangier: Advanced Information Technology, Services and Systems. – Springer International Publishing AG 2018. – pp. 461-473

2. Karthikeyan, P. Improvement in Genetic Algorithm with Genetic Operator Combination (GOC) and Immigrant Strategies for Multicast Routing in Ad Hoc Networks [Text] / P. Karthikeyan, Subramanian Baskar // International Conference on Swarm, Evolutionary, and Memetic Computing – 4th International Conference, SEMCCO 2013, Chennai, India, December 19-21, 2013, Proceedings, Part I: Swarm, Evolutionary, and Memetic Computing. – Springer International Publishing Switzerland 2013. – pp. 481-491

3. Oliveira, S. Genetic Seam Carving: A Genetic Algorithm Approach for Content-Aware Image Retargeting [Text] / Saulo A. F. Oliveira, Francisco N. Bezerra, Ajalmar R. Rocha Neto // Iberian Conference on Pattern Recognition and Image Analysis – 7th Iberian Conference, IbPRIA 2015, Santiago de Compostela, Spain, June 17-19, 2015, Proceedings: Pattern Recognition and Image Analysis. – Springer International Publishing Switzerland 2015. – pp. 700-707

4. Smith, M. Feature Construction and Selection Using Genetic Programming and a Genetic Algorithm [Text] / Matthew G. Smith, Larry Bull // European Conference on Genetic Programming – 6th European Conference, EuroGP 2003 Essex, UK, April 14–16, 2003 Proceedings: Genetic Programming. – Springer-Verlag Berlin Heidelberg 2003. – pp. 229-237

5. Seo, K. A Comparative Study between Genetic Algorithm and Genetic Programming Based Gait Generation Methods for Quadruped Robots [Text] / Kisung Seo, Soohwan Hyun // European Conference on the Applications

of Evolutionary Computation – EvoApplicatons 2010: EvoCOMPLEX, EvoGAMES, EvoIASP, EvoINTELLIGENCE, EvoNUM, and EvoSTOC, Istanbul, Turkey, April 7-9, 2010, Proceedings, Part I: Applications of Evolutionary Computation. – Springer-Verlag Berlin Heidelberg 2010. – pp. 352-360

6. Badariah, Sh. Comparison between Genetic Algorithm and Genetic Programming Performance for Photomosaic Generation [Text] / Shahrul Badariah Mat Sah, Vic Ciesielski, Daryl D'Souza, Marsha Berry // Asia-Pacific Conference on Simulated Evolution and Learning – 7th International Conference, SEAL 2008, Melbourne, Australia, December 7-10, 2008. Proceedings: Simulated Evolution and Learning. – Springer-Verlag Berlin Heidelberg 2008. – pp. 259-268

7. Karr, Ch. Genetic Algorithm Optimization of a Filament Winding Process [Text] / Charles L. Karr, Eric Wilson, Sherri Messimer // Genetic and Evolutionary Computation Conference – Genetic and Evolutionary Computation Conference Chicago, IL, USA, July 12–16, 2003 Proceedings, Part II: Genetic and Evolutionary Computation — GECCO 2003. – Springer-Verlag Berlin Heidelberg 2003. – pp. 2406-2407

8. Nyathi, Th. Automated Design of Genetic Programming Classification Algorithms Using a Genetic Algorithm [Text] / Thambo Nyathi, Nelishia Pillay // European Conference on the Applications of Evolutionary Computation – 20th European Conference, EvoApplications 2017, Amsterdam, The Netherlands, April 19-21, 2017, Proceedings, Part II: Applications of Evolutionary Computation. – Springer International Publishing AG 2017. – pp. 224-239

9. Peng, B. An Adaptive Genetic Simulated Annealing Algorithm for QoS Multicast Routing [Text] / Bo Peng, Lei Li // International Conference on Multimedia, Computer Graphics, and Broadcasting – International Conference, MulGraB 2011, Held as Part of the Future Generation Information Technology Conference, FGIT 2011, in Conjunction with GDC 2011, Jeju Island, Korea,

December 8-10, 2011. Proceedings, Part II: Multimedia, Computer Graphics and Broadcasting. – Springer-Verlag Berlin Heidelberg 2011. – pp. 338-338

10. Xia, Yi. An Improved FastSLAM Algorithm Based on Genetic Algorithms [Text] / Yi-min Xia, Yi-min Yang // International Symposium on Information and Automation – International Symposium, ISIA 2010, Guangzhou, China, November 10-11, 2010. Revised Selected Papers: Information and Automation. – Springer-Verlag Berlin Heidelberg 2011. – pp. 296-302

11. Tian, B. A Feature Selection Algorithm for Big Data Based on Genetic Algorithm [Text] / Bo Tian, Weizhi Xiong // International Conference on Mechatronics and Intelligent Robotics – Proceedings of the International Conference on Mechatronics and Intelligent Robotics (ICMIR2017) - Volume 1: Recent Developments in Mechatronics and Intelligent Robotics. – Springer International Publishing AG 2018. – pp. 159-163

12. Smith, M. Using Genetic Programming for Feature Creation with a Genetic Algorithm Feature Selector [Text] / Matthew G. Smith, Larry Bull // International Conference on Parallel Problem Solving from Nature – 8th International Conference, Birmingham, UK, September 18-22, 2004. Proceedings: Parallel Problem Solving from Nature - PPSN VIII. – Springer-Verlag Berlin Heidelberg 2004. – pp. 1163-1171

13. Zhang, Ch. An Effective Feature Selection Scheme via Genetic Algorithm Using Mutual Information [Text] / Chunkai Zhang, Hong Hu // International Conference on Fuzzy Systems and Knowledge Discovery – Second International Conference, FSKD 2005, Changsha, China, August 27-29, 2005, Proceedings, Part II : Fuzzy Systems and Knowledge Discovery. – Springer-Verlag Berlin Heidelberg 2005. – pp. E1-E1

14. Zhang, M. Using Back Propagation Algorithm and Genetic Algorithm to Train and Refine Neural Networks for Object Detection [Text] / Mengjie Zhang, Victor Ciesielski // International Conference on Database and Expert Systems Applications – DEXA 1999: Database and Expert Systems Applications. – Springer-Verlag Berlin Heidelberg 1999. – pp. 626-635

15. Gao, G. Research on Routing Selection Algorithm Based on Genetic Algorithm [Text] / Guohong Gao, Baojian Zhang, Xueyong Li, Jinna Lv // International Conference on Intelligent Computing and Information Science – International Conference, ICICIS 2011, Chongqing, China, January 8-9, 2011. Proceedings, Part II: Intelligent Computing and Information Science. – Springer-Verlag Berlin Heidelberg 2011. – pp. 353-358

16. Lee, Z. A Hybrid Search Algorithm of Ant Colony Optimization and Genetic Algorithm Applied to Weapon-Target Assignment Problems [Text] / Zne-Jung Lee, Wen-Li Lee // International Conference on Intelligent Data Engineering and Automated Learning – 4th International Conference, IDEAL 2003, Hong Kong, China, March 21-23, 2003. Revised Papers: Intelligent Data Engineering and Automated Learning. – Springer-Verlag Berlin Heidelberg 2003. – pp. 278-285

17. Chiu, Ch. Combining Apriori Algorithm and Constraint-Based Genetic Algorithm for Tree Induction for Aircraft Electronic Ballasts Troubleshooting [Text] / Chaochang Chiu, Pei-Lun Hsu, Nan-Hsing Chiu // International Conference on Natural Computation – Second International Conference, ICNC 2006, Xi'an, China, September 24-28, 2006. Proceedings, Part I: Advances in Natural Computation. – Springer-Verlag Berlin Heidelberg 2006. – pp. 381-384

18. Yang, D. The T-Detectors Maturation Algorithm Based on Genetic Algorithm [Text] / Dongyong Yang, Jungan Chen // Australasian Joint Conference on Artificial Intelligence – 17th Australian Joint Conference on Artificial Intelligence, Cairns, Australia, December 4-6, 2004. Proceedings: AI 2004: Advances in Artificial Intelligence. – Springer-Verlag Berlin Heidelberg 2004. – pp. 1262-1268

19. Gholaminezhad, I. A Multi-Objective Relative Clustering Genetic Algorithm with Adaptive Local/Global Search based on Genetic Relatedness [Text] / Iman Gholaminezhad, Giovanni Iacca // European Conference on the Applications of Evolutionary Computation – 17th European Conference,

EvoApplications 2014, Granada, Spain, April 23-25, 2014, Revised Selected Papers: Applications of Evolutionary Computation. – Springer-Verlag Berlin Heidelberg 2014. – pp. 591-602

20. Wong, K. A technique for improving the convergence characteristic of genetic algorithms and its application to a genetic-based load flow algorithm [Text] / Kit Po Wong, An Li // Asia-Pacific Conference on Simulated Evolution and Learning – First Asia-Pacific Conference, SEAL'96 Taejon, Korea, November 9–12, 1996 Seclected Papers: Simulated Evolution and Learning. – Springer-Verlag Berlin Heidelberg 1997. – pp. 167-176