Министерство образования и науки Российской Федерации
федеральное государственное бюджетное образовательное
учреждение высшего профессионального образования
«Сибирский государственный аэрокосмический университет
имени академика М.Ф. Решетнева»

Сергиенко Антон Борисович

Тестовые функции для глобальной оптимизации. v.1.8

Красноярск – 2013

Оглавление

Условные обозначения **5**

Введение **6**

1 Задачи вещественной оптимизации **7**

1.1 Функция Ackley 7

1.1.1 Описание функции 7

1.1.2 Параметры для алгоритмов оптимизации 8

1.1.3 Основная задача и подзадачи 8

1.1.4 Нахождение ошибки оптимизации 9

1.1.5 Свойства задачи 9

1.1.6 Реализация 9

1.1.7 Ссылки 10

1.2 Аддитивная потенциальная функция 10

1.2.1 Описание функции 10

1.2.2 Параметры для алгоритмов оптимизации 11

1.2.3 Основная задача и подзадачи 12

1.2.4 Нахождение ошибки оптимизации 12

1.2.5 Свойства задачи 12

1.2.6 Реализация 13

1.2.7 Ссылки 13

1.3 Мультипликативная потенциальная функция 13

1.3.1 Описание функции 13

1.3.2 Параметры для алгоритмов оптимизации 14

1.3.3 Основная задача и подзадачи 15

1.3.4 Нахождение ошибки оптимизации 15

1.3.5 Свойства задачи . . . 15
1.3.6 Реализация . . . 16
1.3.7 Ссылки . . . 16
1.4 Функция ReverseGriewank . . . 16
1.4.1 Описание функции . . . 16
1.4.2 Параметры для алгоритмов оптимизации . . . 17
1.4.3 Основная задача и подзадачи . . . 18
1.4.4 Нахождение ошибки оптимизации . . . 18
1.4.5 Свойства задачи . . . 18
1.4.6 Реализация . . . 19
1.4.7 Ссылки . . . 19
1.5 Эллиптический параболоид . . . 19
1.5.1 Описание функции . . . 19
1.5.2 Параметры для алгоритмов оптимизации . . . 20
1.5.3 Основная задача и подзадачи . . . 21
1.5.4 Нахождение ошибки оптимизации . . . 21
1.5.5 Свойства задачи . . . 22
1.5.6 Реализация . . . 22
1.6 Функция Растригина . . . 22
1.6.1 Описание функции . . . 22
1.6.2 Параметры для алгоритмов оптимизации . . . 23
1.6.3 Основная задача и подзадачи . . . 24
1.6.4 Нахождение ошибки оптимизации . . . 24
1.6.5 Свойства задачи . . . 25
1.6.6 Реализация . . . 25
1.6.7 Ссылки . . . 25
1.7 Функция Розенброка . . . 26
1.7.1 Описание функции . . . 26
1.7.2 Параметры для алгоритмов оптимизации . . . 27
1.7.3 Основная задача и подзадачи . . . 27
1.7.4 Нахождение ошибки оптимизации . . . 27
1.7.5 Свойства задачи . . . 28

1.7.6 Реализация 28
1.7.7 Ссылки 29

2 Задачи бинарной оптимизации **30**

2.1 Сумма всех элементов бинарного вектора 30
2.1.1 Описание функции 30
2.1.2 Основная задача и подзадачи 31
2.1.3 Нахождение ошибки оптимизации 31
2.1.4 Свойства задачи 32
2.1.5 Реализация 32

Заключение **33**

Литература **34**

Условные обозначения

$a \in A$ — элемент a принадлежит множеству A.

$\bar{x}$ — обозначение вектора.

$\arg f(x)$ — возвращает аргумент x, при котором функция принимает значение $f(x)$.

$Random(X)$ — случайный выбор элемента из множества X с равной вероятностью.

$Random\left(\{x^i \mid p^i\}\right)$ — случайный выбор элемента x^i из множества X, при условии, что каждый элемент $x^i \in X$ имеет вероятность выбора равную p^i, то есть это обозначение равнозначно предыдущему.

$random(a, b)$ — случайное действительное число из интервала $[a; b]$.

$int(a)$ — целая часть действительного числа a.

$\mu(X)$ — мощность множества X.

Замечание. Оператор присваивания обозначается через знак «=», так же как и знак равенства.

Замечание. Индексация всех массивов в документе начинается с 1. Это стоит помнить при реализации алгоритма на C-подобных языках программирования, где индексация начинается с нуля.

Замечание. Вызывание трех функций: $Random(X)$, $Random\left(\{x_i \mid p_i\}\right)$, $random(a, b)$ – происходит каждый раз, когда по ходу выполнения формул, они встречаются. Если формула итерационная, то нельзя перед ее вызовом один раз определить, например, $random(a, b)$ как константу и потом её использовать на протяжении всех итераций неизменной.

Замечание. Надстрочный индекс может обозначать как возведение в степень, так и индекс элемента. Конкретное обозначение определяется в контексте текста, в котором используется формула с надстрочным индексом.

Замечание. Если у нас имеется множество векторов, то подстрочный индекс обозначает номер компоненты конкретного вектора, а надстрочный индекс обозначает номер вектора во множестве, например, $\bar{x}^i \in X$ $(i = \overline{1, N})$, $\bar{x}^i_j \in \{0; 1\}$, $(j = \overline{1, n})$. В случае, если вектор имеет свое обозначение в виде подстрочной надписи, то компоненты вектора проставляются за скобками, например, $(\bar{x}_{max})_j = 0$ $(j = \overline{1, n})$.

Замечание. При выводе матриц и векторов элементы могут разделяться как пробелом, так и точкой с запятой, то есть обе записи $\begin{pmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \end{pmatrix}^{\mathrm{T}}$ и $(1; 1; 1; 1; 1; 1; 1; 1; 1)^{\mathrm{T}}$ допустимы.

Замечание. При выводе множеств элементы разделяются только точкой с запятой, то есть допустима только такая запись: $\{1; 1; 1; 1; 1; 1; 1; 1; 1\}^{\mathrm{T}}$.

Введение

В данном документе рассмотрено множество тестовых функций, которые можно использовать для проведения исследований алгоритмов оптимизации. К каждой функции дано подробное описание, график (если это возможно), свойств и параметров, которые позволят единообразно проводить сравнения разных алгоритмов оптимизации во избежания несостыковок с точки зрения разного понимания нахождения ошибки, точности работы алгоритмом.

Данный документ представляет его версию **1.2** от 17 декабря 2013 г.

Последнюю версию документа можно найти по адресу:

https://github.com/Harrix/HarrixTestFunctions

Тестовые функции реализованы на языке C++ в библиотеке **HarrixMathLibrary** в разделе «Тестовые функции для оптимизации», которую можно найти по адресу:

https://github.com/Harrix/HarrixMathLibrary.

Все библиографические материалы, которые используются в документе, приведены в виде скриншотов и скринов в папке **_Biblio** на https://github.com/Harrix/HarrixTestFunctions.

С автором можно связаться по адресу sergienkoanton@mail.ru или http://vk.com/harrix.

Сайт автора, где публикуются последние новости: http://blog.harrix.org/, а проекты располагаются по адресу http://harrix.org/.

Глава 1

Задачи вещественной оптимизации

1.1 Функция Ackley

1.1.1 Описание функции

Идентификатор: MHL_TestFunction_Ackley.

Наименование: Функция Ackley.

Тип: Задача вещественной оптимизации.

Формула (целевая функция):

$$f(\bar{x}) = 20 + e - 20e^{-0.2\sqrt{\frac{1}{n}\sum_{i=1}^{n}\bar{x}_i^2}} - e^{\frac{1}{n}\sum_{i=1}^{n}cos(2\pi\cdot\bar{x}_i)}, \text{ где} \tag{1.1}$$

$\bar{x} \in X$, $\bar{x}_j \in [Left_j; Right_j]$, $Left_j = -5$, $Right_j = 5$, $j = \overline{1,n}$.

Обозначение: $\bar{x}$ — вещественный вектор;

n — размерность вещественного вектора.

Решаемая задача оптимизации: $\bar{x}_{min} = \arg\min_{\bar{x}\in X} f(\bar{x})$.

Точка минимума: $\bar{x}_{min} = (0, 0, \ldots, 0)^{\mathrm{T}}$, то есть $(\bar{x}_{min})_j = 0$ $(j = \overline{1,n})$.

Минимум функции: $f(\bar{x}_{min}) = 0$.

График: Рисунок 1.1 нас 8 стр.

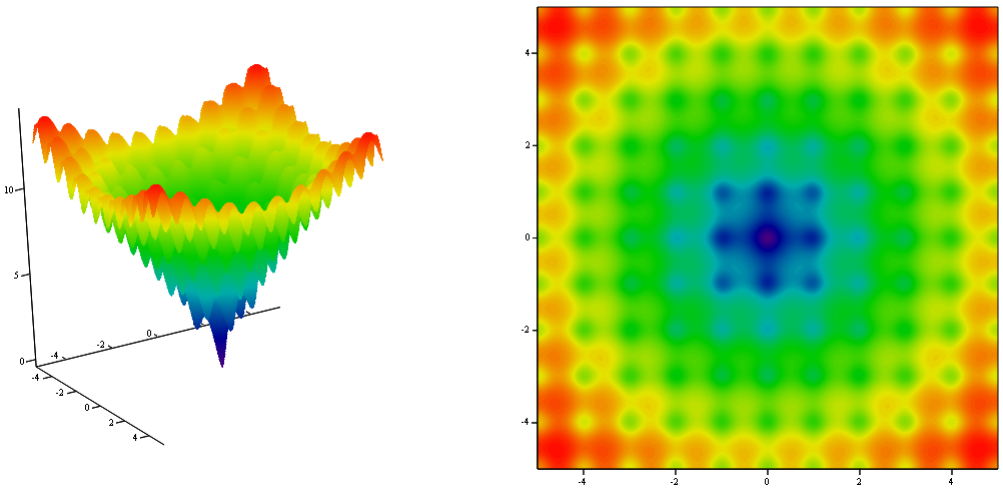

Рисунок 1.1. Функция Ackley

1.1.2 Параметры для алгоритмов оптимизации

Точность вычислений:	$\varepsilon = 0.025$.
Число интервалов, на которые предполагается разбивать каждую компоненту вектора $\bar{x}$ в пределах своего изменения (для алгоритмов дискретной оптимизации) :	$NumberOfParts_j = 4095$ $(j = \overline{1,n})$.
Для этого длина бинарной строки для x_j координаты равна (для алгоритмов бинарной оптимизации) :	$(k_2)_j = 12$ $(j = \overline{1,n})$.

Замечание: $NumberOfParts_j$ выбирается как минимальное число, удовлетворяющее соотношению:

$$NumberOfParts_j = 2^{(k_2)_j} - 1 \geq \frac{10\,(Right_j - Left_j)}{\varepsilon}, \text{где } (k_2)_j \in \mathbb{N}, \left(j = \overline{1,n}\right).$$

1.1.3 Основная задача и подзадачи

Изменяемый параметр:	n — размерность вещественного вектора.
Значение в основной задаче:	$n = 2$.
Подзадача №2:	$n = 3$.
Подзадача №3:	$n = 4$.
Подзадача №4:	$n = 5$.
Подзадача №5:	$n = 10$.
Подзадача №6:	$n = 20$.
Подзадача №7:	$n = 30$.

1.1.4 Нахождение ошибки оптимизации

Пусть в результате работы алгоритма оптимизации за N запусков мы нашли решения $\bar{x}^k_{submin}$ со значениями целевой функции $f\left(\bar{x}^k_{submin}\right)$ соответственно ($k = \overline{1, N}$). Используем три вида ошибок:

Надёжность:

$$R = \frac{\sum_{k=1}^{N} S\left(\bar{x}^k_{submin}\right)}{N}, \text{ где}$$

$$S\left(\bar{x}^k_{submin}\right) = \begin{cases} 1, \text{ если } \left|\left(\bar{x}^k_{submin}\right)_j - \left(\bar{x}_{min}\right)_j\right| \leq \varepsilon, j = \overline{1, n}; \\ 0, \text{ иначе.} \end{cases}$$

Ошибка по входным параметрам:

$$E_x = \frac{\sum_{k=1}^{N} \left(\frac{\sqrt{\sum_{j=1}^{n} \left(\left(\bar{x}^k_{submin}\right)_j - \left(\bar{x}_{min}\right)_j\right)^2}}{n} \right)}{N}.$$

Ошибка по значениям целевой функции:

$$E_f = \frac{\sum_{k=1}^{N} \left| f\left(\bar{x}^k_{submin}\right) - f\left(\bar{x}_{min}\right)\right|}{N}.$$

1.1.5 Свойства задачи

Условной или безусловной оптимизации:	Задача безусловной оптимизации.
Одномерной или многомерной оптимизации:	Многомерной: n.
Функция унимодальная или многоэкстремальная:	Функция многоэкстремальная.
Функция стохастическая или нет:	Функция не стохастическая.
Особенности:	Нет.

1.1.6 Реализация

Реализация функции взята из библиотеки HarrixMathLibrary в разделе «Тестовые функции для оптимизации», которую можно найти по адресу https://github.com/Harrix/HarrixMathLibrary.

Код 1.1. Код функции MHL_TestFunction_Ackley

```
double MHL_TestFunction_Ackley(double *x, int VMHL_N)
{
/*
```

```
Функция многих переменных: Ackley.
Тестовая функция вещественной оптимизации.
Входные параметры:
 x - указатель на исходный массив;
 VMHL_N - размер массива x.
Возвращаемое значение:
 Значение тестовой функции в точке x.
*/
double VMHL_Result;
double f1,f2=0;
f1=exp(-0.2*sqrt(TMHL_SumSquareVector(x,VMHL_N)/double(VMHL_N)));
for (int i=0;i<VMHL_N;i++) f2=f2+cos(2.*MHL_PI*x[i]);
f2=exp(f2/double(VMHL_N));
VMHL_Result=20.+exp(1)-20.*f1-f2;
return VMHL_Result;
}
```

1.1.7 Ссылки

Данная функция приводится в следующих источниках:

1. [1, стр. 5] — Empirical review of standard benchmark functions using evolutionary global optimization.
2. [2] — Ackley's Function.

1.2 Аддитивная потенциальная функция

1.2.1 Описание функции

Идентификатор:	MHL_TestFunction_AdditivePotential.
Наименование:	Аддитивная потенциальная функция.
Тип:	Задача вещественной оптимизации.

Формула (целевая функция):

$$f(\bar{x}) = z(\bar{x}_1) + z(\bar{x}_2), \text{ где} \tag{1.2}$$

$$z(v) = -\frac{1}{(v-1)^2 + 0.2} - \frac{1}{2(v-2)^2 + 0.15} - \frac{1}{3(v-3)^2 + 0.3},$$

$\bar{x} \in X$, $\bar{x}_j \in [Left_j; Right_j]$, $Left_j = 0$, $Right_j = 4$, $j = \overline{1,n}$, $n = 2$.

Обозначение: $\bar{x}$ — вещественный вектор;

$n = 2$ — размерность вещественного вектора.

Решаемая задача оптимизации: $\bar{x}_{min} = \arg\min_{\bar{x} \in X} f(\bar{x})$.

Точка минимума: $\bar{x}_{min} = (2, 2)^{\mathrm{T}}$, то есть $(\bar{x}_{min})_j = 2 \ (j = \overline{1, n})$.

Минимум функции: $f(\bar{x}_{min}) = -15.606060606060606$.

График: Рисунок 1.2 нас 11 стр.

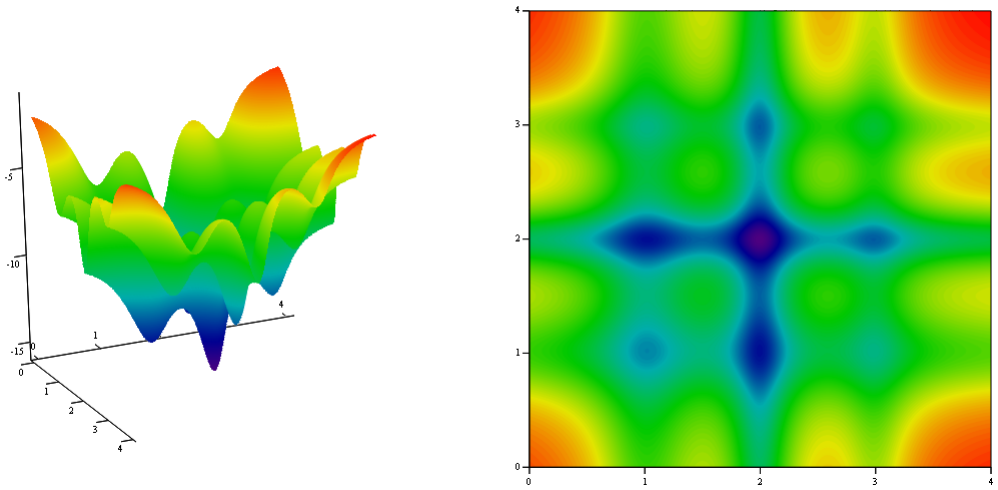

Рисунок 1.2. Аддитивная потенциальная функция

1.2.2 Параметры для алгоритмов оптимизации

Точность вычислений: $\varepsilon = 0.01$.

Число интервалов, на которые предполагается разбивать каждую компоненту вектора $\bar{x}$ в пределах своего изменения (для алгоритмов дискретной оптимизации) : $NumberOfParts_j = 4095 \ (j = \overline{1, n})$.

Для этого длина бинарной строки для x_j координаты равна (для алгоритмов бинарной оптимизации) : $(k_2)_j = 12 \ (j = \overline{1, n})$.

Замечание: $NumberOfParts_j$ выбирается как минимальное число, удовлетворяющее соотношению:

$$NumberOfParts_j = 2^{(k_2)_j} - 1 \geq \frac{10\,(Right_j - Left_j)}{\varepsilon}, \text{где } \ (k_2)_j \in \mathbb{N}, \left(j = \overline{1, n}\right).$$

1.2.3 Основная задача и подзадачи

Изменяемый параметр: n — размерность вещественного вектора.

Значение в основной задаче: $n = 2$.

1.2.4 Нахождение ошибки оптимизации

Пусть в результате работы алгоритма оптимизации за N запусков мы нашли решения $\bar{x}^k_{submin}$ со значениями целевой функции $f\left(\bar{x}^k_{submin}\right)$ соответственно ($k = \overline{1, N}$). Используем три вида ошибок:

Надёжность:

$$R = \frac{\sum_{k=1}^{N} S\left(\bar{x}^k_{submin}\right)}{N}, \text{ где}$$

$$S\left(\bar{x}^k_{submin}\right) = \begin{cases} 1, \text{ если } \left|\left(\bar{x}^k_{submin}\right)_j - \left(\bar{x}_{min}\right)_j\right| \le \varepsilon, j = \overline{1, n}; \\ 0, \text{ иначе.} \end{cases}$$

Ошибка по входным параметрам:

$$E_x = \frac{\sum_{k=1}^{N} \left(\frac{\sqrt{\sum_{j=1}^{n} \left(\left(\bar{x}^k_{submin}\right)_j - \left(\bar{x}_{min}\right)_j\right)^2}}{n} \right)}{N}.$$

Ошибка по значениям целевой функции:

$$E_f = \frac{\sum_{k=1}^{N} \left| f\left(\bar{x}^k_{submin}\right) - f\left(\bar{x}_{min}\right)\right|}{N}.$$

1.2.5 Свойства задачи

Условной или безусловной оптимизации:	Задача безусловной оптимизации.
Одномерной или многомерной оптимизации:	Многомерной: n.
Функция унимодальная или многоэкстремальная:	Функция многоэкстремальная.
Функция стохастическая или нет:	Функция не стохастическая.
Особенности:	Нет.

1.2.6 Реализация

Реализация функции взята из библиотеки HarrixMathLibrary в разделе «Тестовые функции для оптимизации», которую можно найти по адресу https://github.com/Harrix/HarrixMathLibrary.

Код 1.2. Код функции MHL_TestFunction_AdditivePotential

```
double MHL_TestFunction_AdditivePotential(double x, double y)
{
/*
Функция двух переменных: аддитивная потенциальная функция.
Тестовая функция вещественной оптимизации.
Входные параметры:
 x - первая вещественная переменная;
 y - вторая вещественная переменная.
Возвращаемое значение:
 Значение тестовой функции в точке (x,y).
*/
double VMHL_Result;
double z1=-(1./((x-1.)*(x-1.)+0.2))-(1./(2.*(x-2.)*(x-2.)+0.15))-(1./(3.*(x-3.)*(x
    -3.)+0.3));
double z2=-(1./((y-1.)*(y-1.)+0.2))-(1./(2.*(y-2.)*(y-2.)+0.15))-(1./(3.*(y-3.)*(y
    -3.)+0.3));
VMHL_Result=z1+z2;
return VMHL_Result;
}
```

1.2.7 Ссылки

Данная функция приводится в следующих источниках:

1. [3, стр. 33] — Эволюционные методы моделирования и оптимизации сложных систем.

1.3 Мультипликативная потенциальная функция

1.3.1 Описание функции

Идентификатор: MHL_TestFunction_MultiplicativePotential.

Наименование: Мультипликативная потенциальная функция.

Тип: Задача вещественной оптимизации.

Формула (целевая функция):

$$f(\bar{x}) = -z(\bar{x}_1) \cdot z(\bar{x}_2), \text{ где} \tag{1.3}$$

$$z(v) = -\frac{1}{(v-1)^2 + 0.2} - \frac{1}{2(v-2)^2 + 0.15} - \frac{1}{3(v-3)^2 + 0.3},$$

$\bar{x} \in X$, $\bar{x}_j \in [Left_j; Right_j]$, $Left_j = 0$, $Right_j = 4$, $j = \overline{1,n}$, $n = 2$.

Обозначение: $\bar{x}$ — вещественный вектор;

$n = 2$ — размерность вещественного вектора.

Решаемая задача оптимизации: $\bar{x}_{min} = \arg\min_{\bar{x} \in X} f(\bar{x})$.

Точка минимума: $\bar{x}_{min} = (2, 2)^{\mathrm{T}}$, то есть $(\bar{x}_{min})_j = 2$ $(j = \overline{1, n})$.

Минимум функции: $f(\bar{x}_{min}) = -60.8872819100091$.

График: Рисунок 1.3 нас 14 стр.

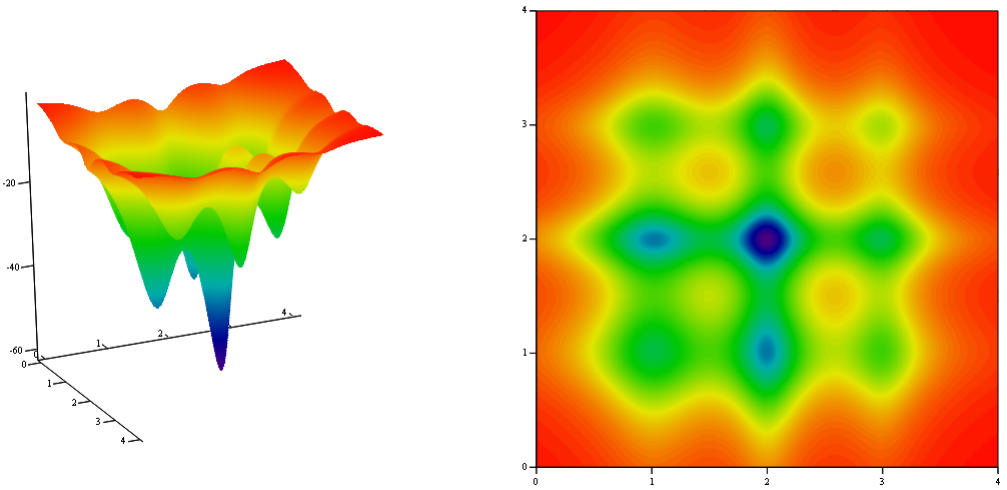

Рисунок 1.3. Мультипликативная потенциальная функция

1.3.2 Параметры для алгоритмов оптимизации

Точность вычислений: $\varepsilon = 0.01$.

Число интервалов, на которые предполагается разбивать каждую компоненту вектора $\bar{x}$ в пределах своего изменения (для алгоритмов дискретной оптимизации) : $NumberOfParts_j = 4095$ $(j = \overline{1, n})$.

Для этого длина бинарной строки для x_j координаты равна (для алгоритмов бинарной оптимизации) : $(k_2)_j = 12$ $(j = \overline{1, n})$.

Замечание: $NumberOfParts_j$ выбирается как минимальное число, удовлетворяющее соотношению:

$$NumberOfParts_j = 2^{(k_2)_j} - 1 \geq \frac{10\,(Right_j - Left_j)}{\varepsilon}, \text{где } (k_2)_j \in \mathbb{N}, \left(j = \overline{1, n}\right).$$

1.3.3 Основная задача и подзадачи

Изменяемый параметр: n — размерность вещественного вектора.

Значение в основной задаче: $n = 2$.

1.3.4 Нахождение ошибки оптимизации

Пусть в результате работы алгоритма оптимизации за N запусков мы нашли решения $\bar{x}^k_{submin}$ со значениями целевой функции $f\left(\bar{x}^k_{submin}\right)$ соответственно ($k = \overline{1, N}$). Используем три вида ошибок:

Надёжность:

$$R = \frac{\sum_{k=1}^{N} S\left(\bar{x}^k_{submin}\right)}{N}, \text{ где}$$

$$S\left(\bar{x}^k_{submin}\right) = \begin{cases} 1, \text{ если } \left|\left(\bar{x}^k_{submin}\right)_j - \left(\bar{x}_{min}\right)_j\right| \le \varepsilon, j = \overline{1, n}; \\ 0, \text{ иначе.} \end{cases}$$

Ошибка по входным параметрам:

$$E_x = \frac{\sum_{k=1}^{N} \left(\frac{\sqrt{\sum_{j=1}^{n} \left(\left(\bar{x}^k_{submin}\right)_j - \left(\bar{x}_{min}\right)_j\right)^2}}{n} \right)}{N}.$$

Ошибка по значениям целевой функции:

$$E_f = \frac{\sum_{k=1}^{N} \left| f\left(\bar{x}^k_{submin}\right) - f\left(\bar{x}_{min}\right)\right|}{N}.$$

1.3.5 Свойства задачи

Условной или безусловной оптимизации:	Задача безусловной оптимизации.
Одномерной или многомерной оптимизации:	Многомерной: n.
Функция унимодальная или многоэкстремальная:	Функция многоэкстремальная.
Функция стохастическая или нет:	Функция не стохастическая.
Особенности:	Нет.

1.3.6 Реализация

Реализация функции взята из библиотеки HarrixMathLibrary в разделе «Тестовые функции для оптимизации», которую можно найти по адресу https://github.com/Harrix/HarrixMathLibrary.

Код 1.3. Код функции MHL_TestFunction_MultiplicativePotential

```
double MHL_TestFunction_MultiplicativePotential(double x, double y)
{
/*
Функция двух переменных: мультипликативная потенциальная функция.
Тестовая функция вещественной оптимизации.
Входные параметры:
 x - первая вещественная переменная;
 y - вторая вещественная переменная.
Возвращаемое значение:
 Значение тестовой функции в точке (x,y).
*/
double VMHL_Result;
double z1=-(1./((x-1.)*(x-1.)+0.2))-(1./(2.*(x-2.)*(x-2.)+0.15))-(1./(3.*(x-3.)*(x
    -3.)+0.3));
double z2=-(1./((y-1.)*(y-1.)+0.2))-(1./(2.*(y-2.)*(y-2.)+0.15))-(1./(3.*(y-3.)*(y
    -3.)+0.3));
VMHL_Result=-z1*z2;
return VMHL_Result;
}
```

1.3.7 Ссылки

Данная функция приводится в следующих источниках:

1. [3, стр. 32] — Эволюционные методы моделирования и оптимизации сложных систем.

1.4 Функция ReverseGriewank

1.4.1 Описание функции

Идентификатор: MHL_TestFunction_ReverseGriewank.

Наименование: Функция ReverseGriewank.

Тип: Задача вещественной оптимизации.

Формула (целевая функция):

$$f(\bar{x}) = \frac{1}{\frac{x^2+y^2}{200} - cos(x)\,cos\left(\frac{y}{\sqrt{2}}\right) + 2}, \text{ где} \tag{1.4}$$

$\bar{x} \in X$, $\bar{x}_j \in [Left_j; Right_j]$, $Left_j = -10$, $Right_j = 10$, $j = \overline{1,n}$, $n = 2$.

Обозначение: $\bar{x}$ — вещественный вектор;

$n = 2$ — размерность вещественного вектора.

Решаемая задача оптимизации: $\bar{x}_{max} = \arg\max_{\bar{x} \in X} f(\bar{x})$.

Точка максимума: $\bar{x}_{max} = (0, 0)^{\mathrm{T}}$, то есть $(\bar{x}_{max})_j = 0$ $(j = \overline{1, n})$.

Максимум функции: $f(\bar{x}_{max}) = 1$.

График: Рисунок 1.4 нас 17 стр.

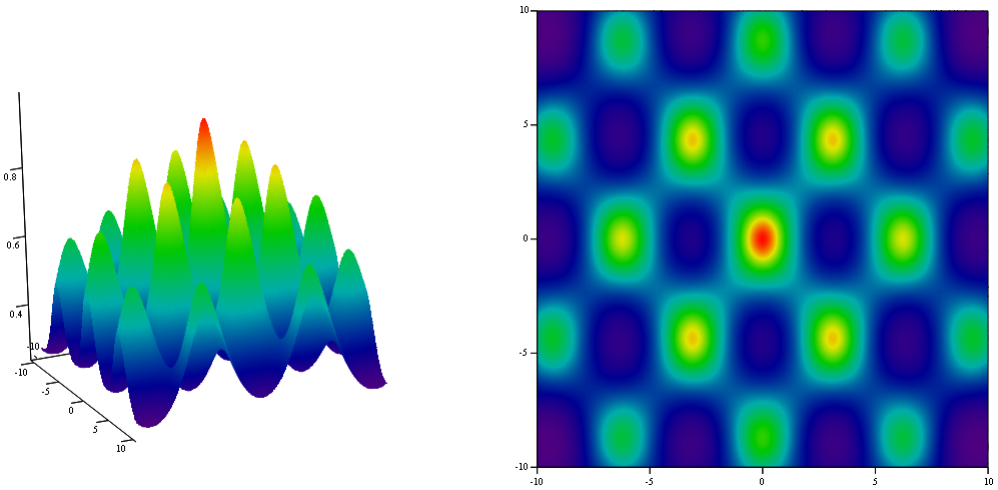

Рисунок 1.4. Функция ReverseGriewank

1.4.2 Параметры для алгоритмов оптимизации

Точность вычислений: $\varepsilon = 0.05$.

Число интервалов, на которые предполагается разбивать каждую компоненту вектора $\bar{x}$ в пределах своего изменения (для алгоритмов дискретной оптимизации) : $NumberOfParts_j = 4095$ $(j = \overline{1, n})$.

Для этого длина бинарной строки для x_j координаты равна (для алгоритмов бинарной оптимизации) : $(k_2)_j = 12$ $(j = \overline{1, n})$.

Замечание: $NumberOfParts_j$ выбирается как минимальное число, удовлетворяющее соотношению:

$$NumberOfParts_j = 2^{(k_2)_j} - 1 \geq \frac{10\,(Right_j - Left_j)}{\varepsilon}, \text{где } (k_2)_j \in \mathbb{N}, \left(j = \overline{1, n}\right).$$

1.4.3 Основная задача и подзадачи

Изменяемый параметр: n — размерность вещественного вектора.

Значение в основной задаче: $n = 2$.

1.4.4 Нахождение ошибки оптимизации

Пусть в результате работы алгоритма оптимизации за N запусков мы нашли решения $\bar{x}^k_{submin}$ со значениями целевой функции $f\left(\bar{x}^k_{submin}\right)$ соответственно ($k = \overline{1, N}$). Используем три вида ошибок:

Надёжность:

$$R = \frac{\sum_{k=1}^{N} S\left(\bar{x}^k_{submin}\right)}{N}, \text{ где}$$

$$S\left(\bar{x}^k_{submin}\right) = \begin{cases} 1, \text{ если } \left|\left(\bar{x}^k_{submin}\right)_j - \left(\bar{x}_{min}\right)_j\right| \le \varepsilon, j = \overline{1, n}; \\ 0, \text{ иначе.} \end{cases}$$

Ошибка по входным параметрам:

$$E_x = \frac{\sum_{k=1}^{N} \left(\frac{\sqrt{\sum_{j=1}^{n} \left(\left(\bar{x}^k_{submin}\right)_j - \left(\bar{x}_{min}\right)_j\right)^2}}{n} \right)}{N}.$$

Ошибка по значениям целевой функции:

$$E_f = \frac{\sum_{k=1}^{N} \left| f\left(\bar{x}^k_{submin}\right) - f\left(\bar{x}_{min}\right)\right|}{N}.$$

1.4.5 Свойства задачи

Условной или безусловной оптимизации:	Задача безусловной оптимизации.
Одномерной или многомерной оптимизации:	Многомерной: n.
Функция унимодальная или многоэкстремальная:	Функция многоэкстремальная.
Функция стохастическая или нет:	Функция не стохастическая.
Особенности:	Нет.

1.4.6 Реализация

Реализация функции взята из библиотеки HarrixMathLibrary в разделе «Тестовые функции для оптимизации», которую можно найти по адресу https://github.com/Harrix/HarrixMathLibrary.

Код 1.4. Код функции MHL_TestFunction_ReverseGriewank

```
double MHL_TestFunction_ReverseGriewank(double x, double y)
{
/*
Функция двух переменных: функция ReverseGriewank.
Тестовая функция вещественной оптимизации.
Входные параметры:
 x - первая вещественная переменная;
 y - вторая вещественная переменная.
Возвращаемое значение:
 Значение тестовой функции в точке (x,y).
*/
double VMHL_Result;

VMHL_Result = 1./((x*x+y*y)/200.-cos(x)*cos(y/sqrt(2.))+2.);

return VMHL_Result;
}
```

1.4.7 Ссылки

Так и не смог найти нормальный источник для этой функции. По внешнему виду похожа на функцию Гриванка, которую возвели в -1 степень. Откуда то у меня находится со студенческих времен.

1.5 Эллиптический параболоид

1.5.1 Описание функции

Идентификатор: MHL_TestFunction_ParaboloidOfRevolution.

Наименование: Эллиптический параболоид.

Тип: Задача вещественной оптимизации.

Формула (целевая функция):

$$f(\bar{x}) = \sum_{i=1}^{n} \bar{x}_i^2, \text{ где} \tag{1.5}$$

$\bar{x} \in X$, $\bar{x}_j \in [Left_j; Right_j]$, $Left_j = -2$, $Right_j = 2$, $j = \overline{1,n}$.

Обозначение: $\bar{x}$ — вещественный вектор;

n — размерность вещественного вектора.

Решаемая задача оптимизации: $\bar{x}_{min} = \arg\min_{\bar{x} \in X} f(\bar{x})$.

Точка минимума: $\bar{x}_{min} = (0, 0, \ldots, 0)^{\mathrm{T}}$, то есть $(\bar{x}_{min})_j = 0$ $(j = \overline{1, n})$.

Минимум функции: $f(\bar{x}_{min}) = 0$.

График: Рисунок 1.5 нас 20 стр.

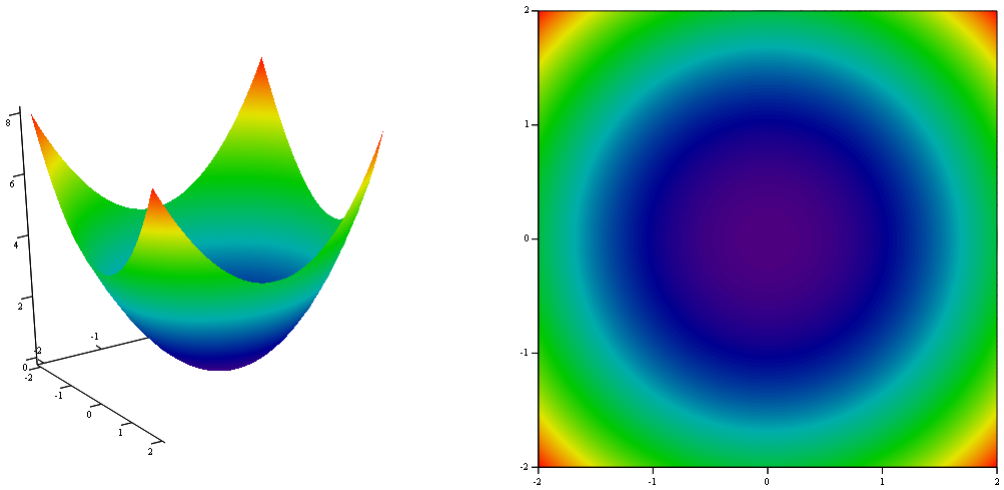

Рисунок 1.5. Эллиптический параболоид

1.5.2 Параметры для алгоритмов оптимизации

Точность вычислений: $\varepsilon = 0.01$.

Число интервалов, на которые предполагается разбивать каждую компоненту вектора $\bar{x}$ в пределах своего изменения (для алгоритмов дискретной оптимизации) : $NumberOfParts_j = 4095$ $(j = \overline{1, n})$.

Для этого длина бинарной строки для x_j координаты равна (для алгоритмов бинарной оптимизации) : $(k_2)_j = 12$ $(j = \overline{1, n})$.

Замечание: $NumberOfParts_j$ выбирается как минимальное число, удовлетворяющее соотношению:

$$NumberOfParts_j = 2^{(k_2)_j} - 1 \geq \frac{10\,(Right_j - Left_j)}{\varepsilon}, \text{где } (k_2)_j \in \mathbb{N}, \left(j = \overline{1, n}\right).$$

1.5.3 Основная задача и подзадачи

Изменяемый параметр:	n — размерность вещественного вектора.
Значение в основной задаче:	$n = 2$.
Подзадача №2:	$n = 3$.
Подзадача №3:	$n = 4$.
Подзадача №4:	$n = 5$.
Подзадача №5:	$n = 10$.
Подзадача №6:	$n = 20$.
Подзадача №7:	$n = 30$.

1.5.4 Нахождение ошибки оптимизации

Пусть в результате работы алгоритма оптимизации за N запусков мы нашли решения $\bar{x}^k_{submin}$ со значениями целевой функции $f\left(\bar{x}^k_{submin}\right)$ соответственно ($k = \overline{1, N}$). Используем три вида ошибок:

Надёжность:

$$R = \frac{\sum_{k=1}^{N} S\left(\bar{x}^k_{submin}\right)}{N}, \text{ где}$$

$$S\left(\bar{x}^k_{submin}\right) = \begin{cases} 1, \text{ если } \left|\left(\bar{x}^k_{submin}\right)_j - \left(\bar{x}_{min}\right)_j\right| \leq \varepsilon, j = \overline{1, n}; \\ 0, \text{ иначе.} \end{cases}$$

Ошибка по входным параметрам:

$$E_x = \frac{\sum_{k=1}^{N} \left(\frac{\sqrt{\sum_{j=1}^{n} \left(\left(\bar{x}^k_{submin}\right)_j - \left(\bar{x}_{min}\right)_j\right)^2}}{n} \right)}{N}.$$

Ошибка по значениям целевой функции:

$$E_f = \frac{\sum_{k=1}^{N} \left| f\left(\bar{x}^k_{submin}\right) - f\left(\bar{x}_{min}\right)\right|}{N}.$$

1.5.5 Свойства задачи

Условной или безусловной оптимизации:	Задача безусловной оптимизации.
Одномерной или многомерной оптимизации:	Многомерной: n.
Функция унимодальная или многоэкстремальная:	Функция унимодальная.
Функция стохастическая или нет:	Функция не стохастическая.
Особенности:	Нет.

1.5.6 Реализация

Реализация функции взята из библиотеки HarrixMathLibrary в разделе «Тестовые функции для оптимизации», которую можно найти по адресу https://github.com/Harrix/HarrixMathLibrary.

Код 1.5. Код функции MHL_TestFunction_ParaboloidOfRevolution

```
double MHL_TestFunction_ParaboloidOfRevolution(double *x, int VMHL_N)
{
/*
Функция многих переменных: Эллиптический параболоид.
Тестовая функция вещественной оптимизации.
Входные параметры:
 x - указатель на исходный массив;
 VMHL_N - размер массива x.
Возвращаемое значение:
 Значение тестовой функции в точке x.
*/
double VMHL_Result=0;
for (int i=0;i<VMHL_N;i++) VMHL_Result+=x[i]*x[i];
return VMHL_Result;
}
```

1.6 Функция Растригина

1.6.1 Описание функции

Идентификатор:	MHL_TestFunction_Rastrigin.
Наименование:	Функция Растригина.
Тип:	Задача вещественной оптимизации.

Формула (целевая функция):

$$f(\bar{x}) = 10n + \sum_{i=1}^{n} \left(\bar{x}_i^2 - 10 \cdot \cos(2\pi \cdot \bar{x}_i)\right), \text{ где} \tag{1.6}$$

$\bar{x} \in X$, $\bar{x}_j \in [Left_j; Right_j]$, $Left_j = -5$, $Right_j = 5$, $j = \overline{1,n}$.

Обозначение:	$\bar{x}$ — вещественный вектор;
	n — размерность вещественного вектора.
Решаемая задача оптимизации:	$\bar{x}_{min} = \arg\min_{\bar{x} \in X} f(\bar{x})$.
Точка минимума:	$\bar{x}_{min} = (0, 0, \ldots, 0)^{\mathrm{T}}$, то есть $(\bar{x}_{min})_j = 0$ $(j = \overline{1,n})$.
Минимум функции:	$f(\bar{x}_{min}) = 0$.
График:	Рисунок 1.6 нас 23 стр.

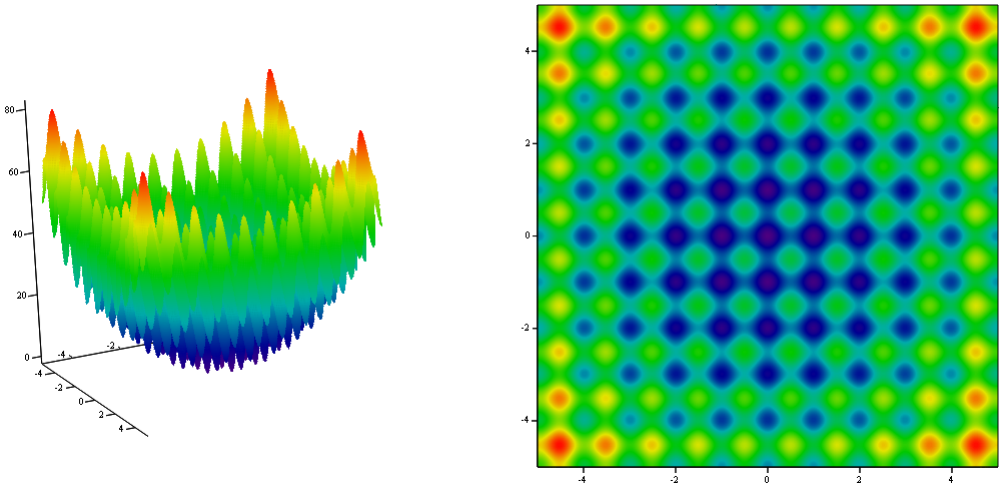

Рисунок 1.6. Функция Растригина

1.6.2 Параметры для алгоритмов оптимизации

Точность вычислений:	$\varepsilon = 0.025$.
Число интервалов, на которые предполагается разбивать каждую компоненту вектора $\bar{x}$ в пределах своего изменения (для алгоритмов дискретной оптимизации) :	$NumberOfParts_j = 4095$ $(j = \overline{1,n})$.
Для этого длина бинарной строки для x_j координаты равна (для алгоритмов бинарной оптимизации) :	$(k_2)_j = 12$ $(j = \overline{1,n})$.

Замечание: $NumberOfParts_j$ выбирается как минимальное число, удовлетворяющее соотношению:

$$NumberOfParts_j = 2^{(k_2)_j} - 1 \geq \frac{10\left(Right_j - Left_j\right)}{\varepsilon}, \text{где } (k_2)_j \in \mathbb{N}, \left(j = \overline{1,n}\right).$$

1.6.3 Основная задача и подзадачи

Изменяемый параметр:	n — размерность вещественного вектора.
Значение в основной задаче:	$n = 2$.
Подзадача №2:	$n = 3$.
Подзадача №3:	$n = 4$.
Подзадача №4:	$n = 5$.
Подзадача №5:	$n = 10$.
Подзадача №6:	$n = 20$.
Подзадача №7:	$n = 30$.

1.6.4 Нахождение ошибки оптимизации

Пусть в результате работы алгоритма оптимизации за N запусков мы нашли решения $\bar{x}^k_{submin}$ со значениями целевой функции $f\left(\bar{x}^k_{submin}\right)$ соответственно ($k = \overline{1,N}$). Используем три вида ошибок:

Надёжность:

$$R = \frac{\sum_{k=1}^{N} S\left(\bar{x}^k_{submin}\right)}{N}, \text{ где}$$

$$S\left(\bar{x}^k_{submin}\right) = \begin{cases} 1, \text{ если } \left|\left(\bar{x}^k_{submin}\right)_j - \left(\bar{x}_{min}\right)_j\right| \leq \varepsilon, j = \overline{1,n}; \\ 0, \text{ иначе.} \end{cases}$$

Ошибка по входным параметрам:

$$E_x = \frac{\sum_{k=1}^{N} \left(\frac{\sqrt{\sum_{j=1}^{n} \left(\left(\bar{x}^k_{submin}\right)_j - \left(\bar{x}_{min}\right)_j\right)^2}}{n} \right)}{N}.$$

Ошибка по значениям целевой функции:

$$E_f = \frac{\sum_{k=1}^{N} \left| f\left(\bar{x}^k_{submin}\right) - f\left(\bar{x}_{min}\right)\right|}{N}.$$

1.6.5 Свойства задачи

Условной или безусловной оптимизации:	Задача безусловной оптимизации.
Одномерной или многомерной оптимизации:	Многомерной: n.
Функция унимодальная или многоэкстремальная:	Функция многоэкстремальная.
Функция стохастическая или нет:	Функция не стохастическая.
Особенности:	Нет.

1.6.6 Реализация

Реализация функции взята из библиотеки HarrixMathLibrary в разделе «Тестовые функции для оптимизации», которую можно найти по адресу https://github.com/Harrix/HarrixMathLibrary.

Код 1.6. Код функции MHL_TestFunction_Rastrigin

```
double MHL_TestFunction_Rastrigin(double *x, int VMHL_N)
{
/*
Функция многих переменных: функция Растригина.
Тестовая функция вещественной оптимизации.
Входные параметры:
 x - указатель на исходный массив;
 VMHL_N - размер массива x.
Возвращаемое значение:
 Значение тестовой функции в точке x.
*/
double VMHL_Result=0;
for (int i=0;i<VMHL_N;i++) VMHL_Result+=x[i]*x[i]-10.*cos(2.*MHL_PI*x[i]);
VMHL_Result+=10*VMHL_N;
return VMHL_Result;
}
```

1.6.7 Ссылки

Данная функция приводится в следующих источниках:

1. [4] — Rastrigin function.
2. [5] — Non-linear Continuous Multi-Extremal Optimization.
3. [6] — Parametric Optimization.

1.7 Функция Розенброка

1.7.1 Описание функции

Идентификатор: MHL_TestFunction_Rosenbrock.

Наименование: Функция Розенброка.

Тип: Задача вещественной оптимизации.

Формула (целевая функция):

$$f(\bar{x}) = \sum_{i=1}^{n-1} \left(100\left(\bar{x}_{i+1} - \bar{x}_i^2\right)^2 + (1 - \bar{x}_i)^2 \right), \text{ где} \tag{1.7}$$

$\bar{x} \in X$, $\bar{x}_j \in [Left_j; Right_j]$, $Left_j = -2$, $Right_j = 2$, $j = \overline{1,n}$.

Обозначение: $\bar{x}$ — вещественный вектор;

n — размерность вещественного вектора.

Решаемая задача оптимизации: $\bar{x}_{min} = \arg\min_{\bar{x} \in X} f(\bar{x})$.

Точка минимума: $\bar{x}_{min} = (1, 1, \ldots, 1)^{\mathrm{T}}$, то есть $(\bar{x}_{min})_j = 1$ $(j = \overline{1,n})$.

Минимум функции: $f(\bar{x}_{min}) = 0$.

График: Рисунок 1.7 нас 26 стр.

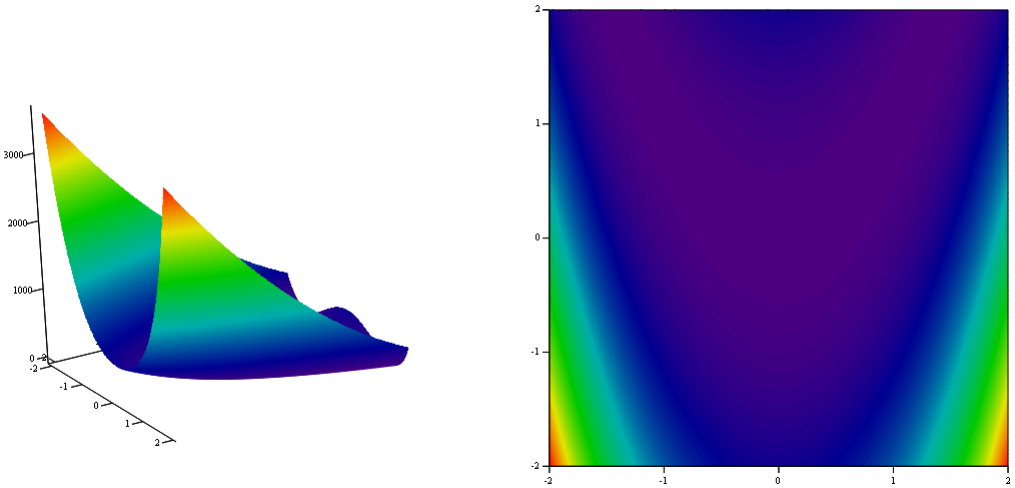

Рисунок 1.7. Функция Розенброка

1.7.2 Параметры для алгоритмов оптимизации

Точность вычислений:	$\varepsilon = 0.01$.
Число интервалов, на которые предполагается разбивать каждую компоненту вектора $\bar{x}$ в пределах своего изменения (для алгоритмов дискретной оптимизации) :	$NumberOfParts_j = 4095$ $(j = \overline{1,n})$.
Для этого длина бинарной строки для x_j координаты равна (для алгоритмов бинарной оптимизации) :	$(k_2)_j = 12$ $(j = \overline{1,n})$.

Замечание: $NumberOfParts_j$ выбирается как минимальное число, удовлетворяющее соотношению:

$$NumberOfParts_j = 2^{(k_2)_j} - 1 \geq \frac{10\,(Right_j - Left_j)}{\varepsilon}, \text{где } (k_2)_j \in \mathbb{N}, \left(j = \overline{1,n}\right).$$

1.7.3 Основная задача и подзадачи

Изменяемый параметр:	n — размерность вещественного вектора.
Значение в основной задаче:	$n = 2$.
Подзадача №2:	$n = 3$.
Подзадача №3:	$n = 4$.
Подзадача №4:	$n = 5$.
Подзадача №5:	$n = 10$.
Подзадача №6:	$n = 20$.
Подзадача №7:	$n = 30$.

1.7.4 Нахождение ошибки оптимизации

Пусть в результате работы алгоритма оптимизации за N запусков мы нашли решения $\bar{x}^k_{submin}$ со значениями целевой функции $f\left(\bar{x}^k_{submin}\right)$ соответственно $(k = \overline{1,N})$. Используем три вида ошибок:

Надёжность:

$$R = \frac{\sum_{k=1}^{N} S\left(\bar{x}^k_{submin}\right)}{N}, \text{ где}$$

$$S\left(\bar{x}^k_{submin}\right) = \begin{cases} 1, \text{ если } \left|\left(\bar{x}^k_{submin}\right)_j - (\bar{x}_{min})_j\right| \leq \varepsilon, j = \overline{1,n}; \\ 0, \text{ иначе.} \end{cases}$$

Ошибка по входным параметрам:

$$E_x = \frac{\sum_{k=1}^{N} \left(\frac{\sqrt{\sum_{j=1}^{n} \left(\left(\bar{x}_{submin}^{k} \right)_j - \left(\bar{x}_{min} \right)_j \right)^2}}{n} \right)}{N}.$$

Ошибка по значениям целевой функции:

$$E_f = \frac{\sum_{k=1}^{N} \left| f\left(\bar{x}_{submin}^{k} \right) - f\left(\bar{x}_{min} \right) \right|}{N}.$$

1.7.5 Свойства задачи

Условной или безусловной оптимизации:	Задача безусловной оптимизации.
Одномерной или многомерной оптимизации:	Многомерной: n.
Функция унимодальная или многоэкстремальная:	Функция многоэкстремальная.
Функция стохастическая или нет:	Функция не стохастическая.
Особенности:	Нет.

1.7.6 Реализация

Реализация функции взята из библиотеки HarrixMathLibrary в разделе «Тестовые функции для оптимизации», которую можно найти по адресу https://github.com/Harrix/HarrixMathLibrary.

Код 1.7. Код функции MHL_TestFunction_Rosenbrock

```
double MHL_TestFunction_Rosenbrock(double *x, int VMHL_N)
{
/*
Функция многих переменных: функция Розенброка.
Тестовая функция вещественной оптимизации.
Входные параметры:
 x - указатель на исходный массив;
 VMHL_N - размер массива x.
Возвращаемое значение:
 Значение тестовой функции в точке x.
*/
double VMHL_Result=0;
for (int i=0;i<VMHL_N-1;i++) VMHL_Result+=100.*(x[i+1]-x[i]*x[i])*(x[i+1]-x[i]*x[i])
    +(1.-x[i])*(1.-x[i]);
return VMHL_Result;
}
```

1.7.7 Ссылки

Данная функция приводится в следующих источниках:

1. [7] — Rosenbrock function.
2. [8] — Rosenbrock Function.

Глава 2

Задачи бинарной оптимизации

2.1 Сумма всех элементов бинарного вектора

2.1.1 Описание функции

Идентификатор: MHL_TestFunction_SumVector.

Наименование: Сумма всех элементов бинарного вектора.

Тип: Задача бинарной оптимизации.

Формула (целевая функция):

$$f(\bar{x}) = \sum_{i=1}^{n} \bar{x}_i, \text{ где} \tag{2.1}$$

$\bar{x} \in X$, $\bar{x}_j \in \{0; 1\}$, $j = \overline{1, n}$.

Обозначение: $\bar{x}$ — бинарный вектор;

n — размерность бинарного вектора.

Объем поискового пространства: $\mu(X) = 2^n$.

Решаемая задача оптимизации: $\bar{x}_{max} = \arg\max_{\bar{x} \in X} f(\bar{x})$.

Точка максимума: $\bar{x}_{max} = (1, 1, \ldots, 1)^{\mathrm{T}}$, то есть $(\bar{x}_{max})_j = 1$ $(j = \overline{1, n})$.

Максимум функции: $f(\bar{x}_{max}) = n$.

Точка минимума: $\bar{x}_{min} = (0, 0, \ldots, 0)^{\mathrm{T}}$, то есть $(\bar{x}_{min})_j = 0$ $(j = \overline{1, n})$.

Минимум функции: $f(\bar{x}_{min}) = 0$.

2.1.2 Основная задача и подзадачи

Изменяемый параметр:	n — размерность бинарного вектора.
Значение в основной задаче:	$n = 20$.
Подзадача №2:	$n = 30$.
Подзадача №3:	$n = 40$.
Подзадача №4:	$n = 50$.
Подзадача №5:	$n = 60$.
Подзадача №6:	$n = 70$.
Подзадача №7:	$n = 80$.
Подзадача №8:	$n = 90$.
Подзадача №9:	$n = 100$.
Подзадача №10:	$n = 200$.

2.1.3 Нахождение ошибки оптимизации

Пусть в результате работы алгоритма оптимизации за N запусков мы нашли решения $\bar{x}^k_{submax}$ со значениями целевой функции $f\left(\bar{x}^k_{submax}\right)$ соответственно ($k = \overline{1, N}$). Используем три вида ошибок:

Надёжность:

$$R = \frac{\sum_{k=1}^{N} S\left(\bar{x}^k_{submax}\right)}{N}, \text{ где}$$

$$S\left(\bar{x}^k_{submax}\right) = \begin{cases} 1, \text{ если } \bar{x}^k_{submax} = \bar{x}_{max}; \\ 0, \text{ иначе.} \end{cases}$$

Ошибка по входным параметрам:

$$E_x = \frac{\sum_{k=1}^{N} \left(\frac{\sum_{j=1}^{n} \left| \left(\bar{x}^k_{submax}\right)_j - (\bar{x}_{max})_j \right|}{n} \right)}{N}.$$

Ошибка по значениям целевой функции:

$$E_f = \frac{\sum_{k=1}^{N} \left(\frac{\left| f\left(\bar{x}^k_{submax}\right) - f(\bar{x}_{max}) \right|}{n} \right)}{N}.$$

2.1.4 Свойства задачи

Условной или безусловной оптимизации:	Задача безусловной оптимизации.
Одномерной или многомерной оптимизации:	Многомерной: n.
Функция унимодальная или многоэкстремальная:	Функция унимодальная.
Функция стохастическая или нет:	Функция не стохастическая.
Особенности:	Нет.

2.1.5 Реализация

Реализация функции взята из библиотеки HarrixMathLibrary в разделе «Тестовые функции для оптимизации», которую можно найти по адресу https://github.com/Harrix/HarrixMathLibrary.

Код 2.1. Код функции MHL_TestFunction_SumVector

```
double MHL_TestFunction_SumVector(int *x, int VMHL_N)
{
/*
Сумма всех элементов бинарного вектора.
Тестовая функция бинарной оптимизации.
Входные параметры:
 x - указатель на исходный массив;
 VMHL_N - размер массива x.
Возвращаемое значение:
 Значение тестовой функции в точке x.
*/
double VMHL_Result=0;
for (int i=0;i<VMHL_N;i++) VMHL_Result+=x[i];
return VMHL_Result;
}
```

Заключение

В данном документе были рассмотрены множество тестовых функций, которые позволят более корректно проводить исследования в области глобальной оптимизации.

Литература

1. Dieterich Johannes M., Hartke Bernd. Empirical review of standard benchmark functions using evolutionary global optimization // CoRR. 2012. T. abs/1207.4318.
2. Ackley's Function. http://www.cs.unm.edu/~neal.holts/dga/benchmarkFunction/ackley.html.
3. Эволюционные методы моделирования и оптимизации сложных систем / Е. С. Семенкин, М. Н. Жукова, В. Г. Жуков [и др.]. Красноярск: Федеральное агентство по образованию, Сибирский федеральный университет, 2007. 310 с. http://files.lib.sfu-kras.ru/ebibl/umkd/22/u_lectures.pdf.
4. Rastrigin function. http://en.wikipedia.org/wiki/Rastrigin_function.
5. Non-linear Continuous Multi-Extremal Optimization. http://www.maths.uq.edu.au/CEToolBox/node3.html.
6. Parametric Optimization. http://www.pg.gda.pl/~mkwies/dyd/geadocu/fcnfun6.html.
7. Rosenbrock function. http://en.wikipedia.org/wiki/Rosenbrock_function.
8. Rosenbrock Function. http://www-optima.amp.i.kyoto-u.ac.jp/member/student/hedar/Hedar_files/TestGO_files/Page2537.htm.