

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.1.11**

Красноярск – 2013

# Оглавление

<b>Условные обозначения</b>	<b>6</b>
<b>Введение</b>	<b>7</b>
<b>1 Задачи вещественной оптимизации</b>	<b>8</b>
1.1 Функция Ackley . . . . .	8
1.1.1 Описание функции . . . . .	8
1.1.2 Параметры для алгоритмов оптимизации . . . . .	9
1.1.3 Основная задача и подзадачи . . . . .	9
1.1.4 Нахождение ошибки оптимизации . . . . .	10
1.1.5 Свойства задачи . . . . .	10
1.1.6 Реализация . . . . .	10
1.1.7 Ссылки . . . . .	11
1.2 Функция Гипер-эллипсоид . . . . .	11
1.2.1 Описание функции . . . . .	11
1.2.2 Параметры для алгоритмов оптимизации . . . . .	12
1.2.3 Основная задача и подзадачи . . . . .	13
1.2.4 Нахождение ошибки оптимизации . . . . .	13
1.2.5 Свойства задачи . . . . .	14
1.2.6 Реализация . . . . .	14
1.2.7 Ссылки . . . . .	14
1.3 Эллиптический параболоид . . . . .	15
1.3.1 Описание функции . . . . .	15
1.3.2 Параметры для алгоритмов оптимизации . . . . .	16
1.3.3 Основная задача и подзадачи . . . . .	16
1.3.4 Нахождение ошибки оптимизации . . . . .	16

1.3.5	Свойства задачи . . . . .	17
1.3.6	Реализация . . . . .	17
1.4	Функция Растригина . . . . .	18
1.4.1	Описание функции . . . . .	18
1.4.2	Параметры для алгоритмов оптимизации . . . . .	19
1.4.3	Основная задача и подзадачи . . . . .	19
1.4.4	Нахождение ошибки оптимизации . . . . .	19
1.4.5	Свойства задачи . . . . .	20
1.4.6	Реализация . . . . .	20
1.4.7	Ссылки . . . . .	21
1.5	Функция Розенброка . . . . .	21
1.5.1	Описание функции . . . . .	21
1.5.2	Параметры для алгоритмов оптимизации . . . . .	22
1.5.3	Основная задача и подзадачи . . . . .	22
1.5.4	Нахождение ошибки оптимизации . . . . .	23
1.5.5	Свойства задачи . . . . .	23
1.5.6	Реализация . . . . .	23
1.5.7	Ссылки . . . . .	24
1.6	Функция Развернутый гипер-эллипсоид . . . . .	24
1.6.1	Описание функции . . . . .	24
1.6.2	Параметры для алгоритмов оптимизации . . . . .	25
1.6.3	Основная задача и подзадачи . . . . .	25
1.6.4	Нахождение ошибки оптимизации . . . . .	26
1.6.5	Свойства задачи . . . . .	26
1.6.6	Реализация . . . . .	26
1.6.7	Ссылки . . . . .	27
1.7	Аддитивная потенциальная функция . . . . .	27
1.7.1	Описание функции . . . . .	27
1.7.2	Параметры для алгоритмов оптимизации . . . . .	28
1.7.3	Основная задача и подзадачи . . . . .	29
1.7.4	Нахождение ошибки оптимизации . . . . .	29
1.7.5	Свойства задачи . . . . .	29

1.7.6	Реализация . . . . .	30
1.7.7	Ссылки . . . . .	30
1.8	Мультипликативная потенциальная функция . . . . .	30
1.8.1	Описание функции . . . . .	30
1.8.2	Параметры для алгоритмов оптимизации . . . . .	31
1.8.3	Основная задача и подзадачи . . . . .	32
1.8.4	Нахождение ошибки оптимизации . . . . .	32
1.8.5	Свойства задачи . . . . .	32
1.8.6	Реализация . . . . .	33
1.8.7	Ссылки . . . . .	33
1.9	Функция ReverseGriewank . . . . .	33
1.9.1	Описание функции . . . . .	33
1.9.2	Параметры для алгоритмов оптимизации . . . . .	34
1.9.3	Основная задача и подзадачи . . . . .	35
1.9.4	Нахождение ошибки оптимизации . . . . .	35
1.9.5	Свойства задачи . . . . .	35
1.9.6	Реализация . . . . .	36
1.9.7	Ссылки . . . . .	36
1.10	Функция волна . . . . .	36
1.10.1	Описание функции . . . . .	36
1.10.2	Параметры для алгоритмов оптимизации . . . . .	37
1.10.3	Основная задача и подзадачи . . . . .	38
1.10.4	Нахождение ошибки оптимизации . . . . .	38
1.10.5	Свойства задачи . . . . .	38
1.10.6	Реализация . . . . .	39
1.10.7	Ссылки . . . . .	39
<b>2</b>	<b>Задачи бинарной оптимизации</b>	<b>40</b>
2.1	Сумма всех элементов бинарного вектора . . . . .	40
2.1.1	Описание функции . . . . .	40
2.1.2	Основная задача и подзадачи . . . . .	41
2.1.3	Нахождение ошибки оптимизации . . . . .	41

2.1.4	Свойства задачи . . . . .	42
2.1.5	Реализация . . . . .	42
<b>Заключение</b>		<b>43</b>
<b>Литература</b>		<b>44</b>

# 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.

$Random(\{x^i \mid p^i\})$  — 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 С-подобных языках программирования, где индексация начинается с нуля.

**Замечание.** Вызывание трех функций:  $Random(X)$ ,  $Random(\{x_i \mid p_i\})$ ,  $random(a, b)$  — 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,  $\bar{x}^i \in X$  ( $i = \overline{1, N}$ ),  $\bar{x}_j^i \in \{0; 1\}$ , ( $j = \overline{1, n}$ ). 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  $(1 \ 1 \ 1 \ 1 \ 1 \ 1 \ 1 \ 1)^T$  и  $(1; 1; 1; 1; 1; 1; 1; 1)^T$  допустимы.

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

# 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 20 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](mailto: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

<b>Идентификатор:</b>	MHL_TestFunction_Ackley.
<b>Наименование:</b>	Функция Ackley.
<b>Тип:</b>	Задача вещественной оптимизации.

**Формула** (целевая функция):

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

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

<b>Обозначение:</b>	$\bar{x}$ — вещественный вектор; $n$ — размерность вещественного вектора.
<b>Решаемая задача оптимизации:</b>	$\bar{x}_{min} = \arg \min_{\bar{x} \in X} f(\bar{x})$ .
<b>Точка минимума:</b>	$\bar{x}_{min} = (0, 0, \dots, 0)^T$ , то есть $(\bar{x}_{min})_j = 0$ ( $j = \overline{1, n}$ ).
<b>Минимум функции:</b>	$f(\bar{x}_{min}) = 0$ .
<b>График:</b>	Рисунок 1.1 нас 9 стр.



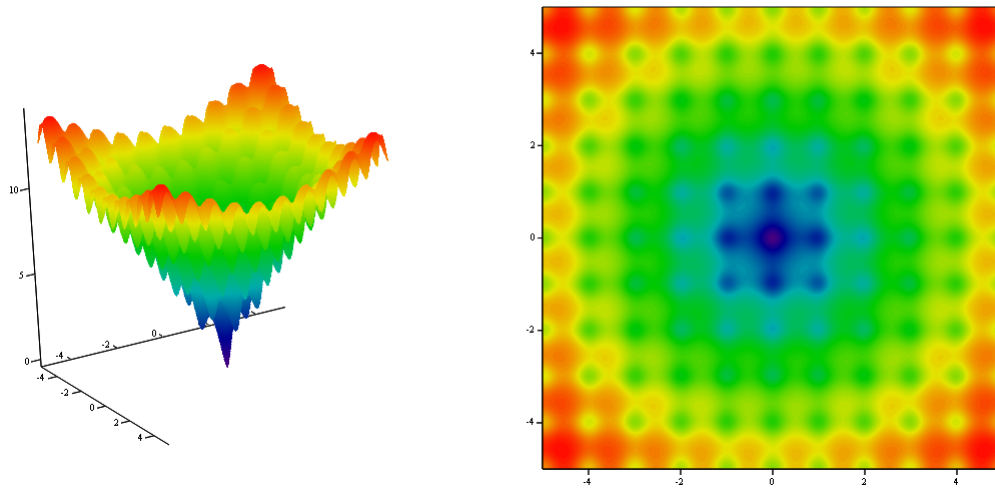


Рисунок 1.1. Функция Ackley

### 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:

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

### 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  $\bar{x}_{submin}^k$  со значениями целевой функции  $f(\bar{x}_{submin}^k)$  соответственно ( $k = \overline{1, N}$ ). Используем три вида ошибок:

**Надёжность:**

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

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

**Ошибка по входным параметрам:**

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

**Ошибка по значениям целевой функции:**

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

### 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 Функция Гипер-эллипсоид

### 1.2.1 Описание функции

<b>Идентификатор:</b>	MHL_TestFunction_HyperEllipsoid.
<b>Наименование:</b>	Гипер-эллипсоид.
<b>Тип:</b>	Задача вещественной оптимизации.
<b>Формула</b> (целевая функция):	

$$f(\bar{x}) = \sum_{i=1}^n (i \cdot \bar{x}_i)^2, \text{ где} \quad (1.2)$$

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

**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:**

$$\bar{x}_{min} = (0, 0, \dots, 0)^T, \text{ то есть } (\bar{x}_{min})_j = 0 \ (j = \overline{1, n}).$$

**Минимум функции:**

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

**График:**

Рисунок 1.2 нас 12 стр.

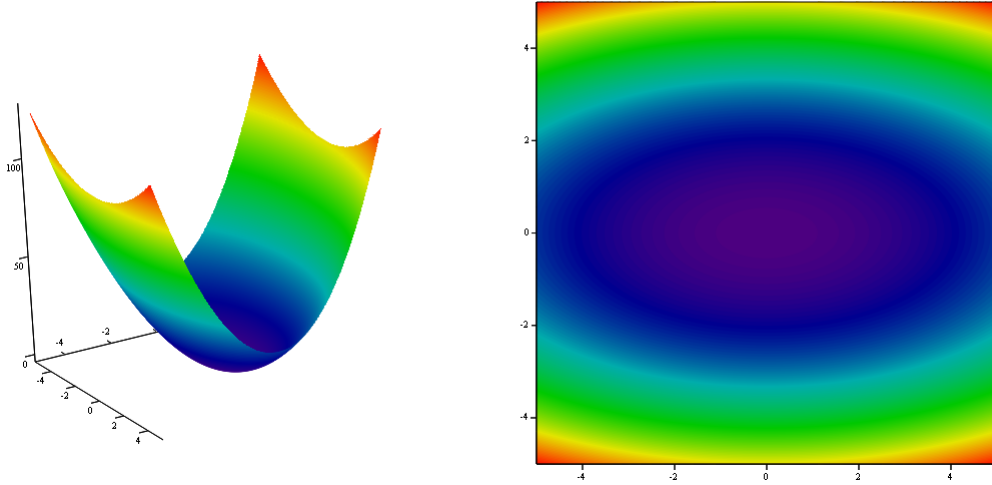


Рисунок 1.2. Гипер-эллипсоид

## 1.2.2 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:

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

### 1.2.3 Основная задача и подзадачи

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

### 1.2.4 Нахождение ошибки оптимизации

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

**Надёжность:**

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

**Ошибка по входным параметрам:**

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

**Ошибка по значениям целевой функции:**

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

### 1.2.5 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.

### 1.2.6 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_HyperEllipsoid`

```
double MHL_TestFunction_HyperEllipsoid(double *x, int VMHL_N)
{
    /*
    Функция многих переменных: Гипер-эллипсоид.
    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 += (i+1)*(i+1)*x[i]*x[i];

    return VMHL_Result;
}
```

### 1.2.7 Ссылки

В данном виде тестовую функцию в литературе не нашел. Обычно используется несколько иной вид этой функции, когда  $i$  не возводится в квадрат.

## 1.3 Эллиптический параболоид

### 1.3.1 Описание функции

**Идентификатор:** MHL\_TestFunction\_ParaboloidOfRevolution.

**Наименование:** Эллиптический параболоид.

**Тип:** Задача вещественной оптимизации.

**Формула** (целевая функция):

$$f(\bar{x}) = \sum_{i=1}^n \bar{x}_i^2, \text{ где} \quad (1.3)$$

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

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

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

**График:** Рисунок 1.3 нас 15 стр.

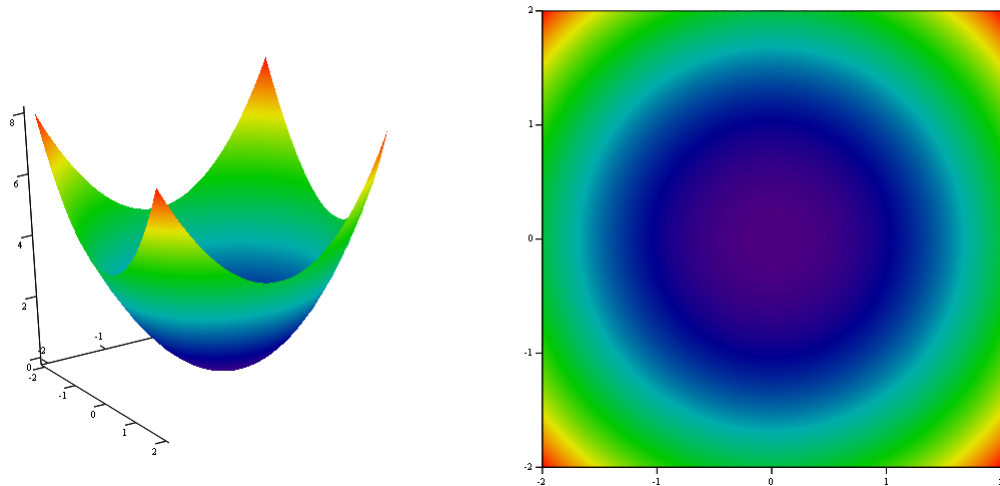


Рисунок 1.3. Эллиптический параболоид

### 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:

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

### 1.3.3 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:**  $n = 30$ .

### 1.3.4 Нахождение ошибки оптимизации

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

**Надёжность:**

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



**Ошибка по входным параметрам:**

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

**Ошибка по значениям целевой функции:**

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

### 1.3.5 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.

### 1.3.6 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\_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;
}
```

## 1.4 Функция Растригина

### 1.4.1 Описание функции

<b>Идентификатор:</b>	MHL_TestFunction_Rastrigin.
<b>Наименование:</b>	Функция Растригина.
<b>Тип:</b>	Задача вещественной оптимизации.
<b>Формула</b> (целевая функция):	

$$f(\bar{x}) = 10n + \sum_{i=1}^n (\bar{x}_i^2 - 10 \cdot \cos(2\pi \cdot \bar{x}_i)), \text{ где} \quad (1.4)$$

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

<b>Обозначение:</b>	$\bar{x}$ — вещественный вектор; $n$ — размерность вещественного вектора.
<b>Решаемая задача оптимизации:</b>	$\bar{x}_{min} = \arg \min_{\bar{x} \in X} f(\bar{x})$ .
<b>Точка минимума:</b>	$\bar{x}_{min} = (0, 0, \dots, 0)^T$ , то есть $(\bar{x}_{min})_j = 0$ ( $j = \overline{1, n}$ ).
<b>Минимум функции:</b>	$f(\bar{x}_{min}) = 0$ .
<b>График:</b>	Рисунок 1.4 нас 18 стр.

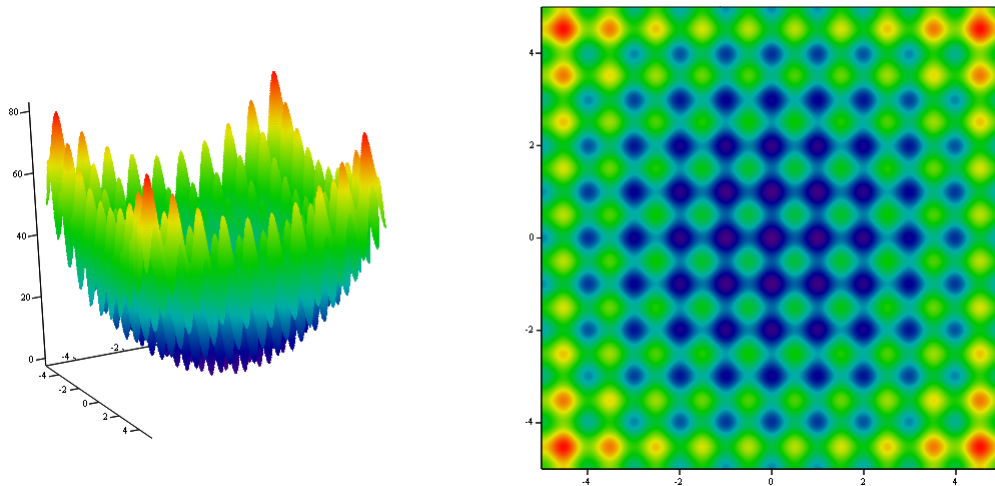


Рисунок 1.4. Функция Растригина

## 1.4.2 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:

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

## 1.4.3 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:**  $n = 30$ .

## 1.4.4 Нахождение ошибки оптимизации

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

**Надёжность:**

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

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

**Ошибка по входным параметрам:**

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

**Ошибка по значениям целевой функции:**

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

### 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\_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;
}
```

## 1.4.7 Ссылки

Данная функция приводится в следующих источниках:

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

## 1.5 Функция Розенброка

### 1.5.1 Описание функции

<b>Идентификатор:</b>	MHL_TestFunction_Rosenbrock.
<b>Наименование:</b>	Функция Розенброка.
<b>Тип:</b>	Задача вещественной оптимизации.

**Формула** (целевая функция):

$$f(\bar{x}) = \sum_{i=1}^{n-1} \left( 100(\bar{x}_{i+1} - \bar{x}_i^2)^2 + (1 - \bar{x}_i)^2 \right), \text{ где} \quad (1.5)$$

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

<b>Обозначение:</b>	$\bar{x}$ — вещественный вектор; $n$ — размерность вещественного вектора.
<b>Решаемая задача оптимизации:</b>	$\bar{x}_{min} = \arg \min_{\bar{x} \in X} f(\bar{x})$ .
<b>Точка минимума:</b>	$\bar{x}_{min} = (1, 1, \dots, 1)^T$ , то есть $(\bar{x}_{min})_j = 1$ ( $j = \overline{1, n}$ ).
<b>Минимум функции:</b>	$f(\bar{x}_{min}) = 0$ .
<b>График:</b>	Рисунок 1.5 нас 22 стр.

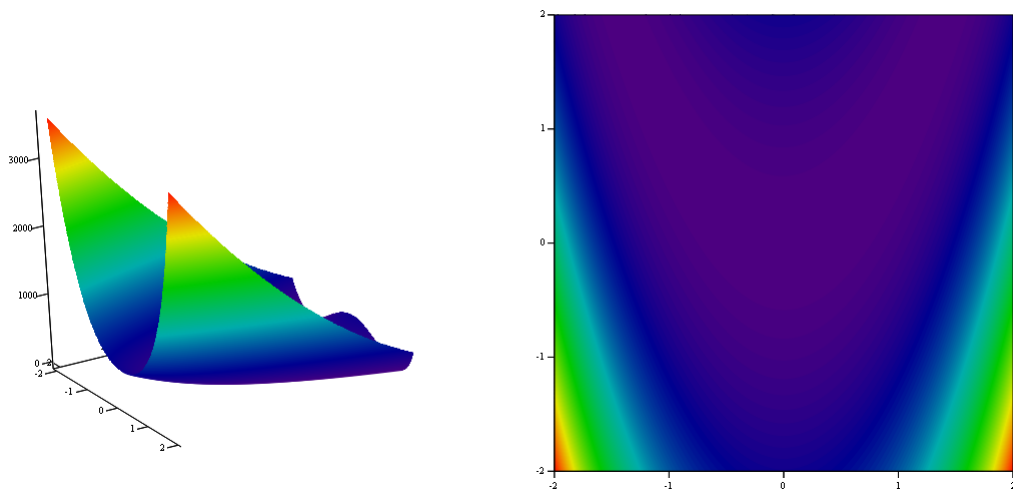


Рисунок 1.5. Функция Розенброка

## 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) :

$$NumberOfParts_j = 4095 \quad (j = \overline{1, n}).$$

**Для этого длина бинарной строки для  $x_j$  координаты равна** (для алгоритмов бинарной оптимизации) :

$$(k_2)_j = 12 \quad (j = \overline{1, n}).$$

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

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

## 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  $\bar{x}_{submin}^k$  со значениями целевой функции  $f(\bar{x}_{submin}^k)$  соответственно ( $k = \overline{1, N}$ ). Используем три вида ошибок:

**Надёжность:**

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

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

**Ошибка по входным параметрам:**

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

**Ошибка по значениям целевой функции:**

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

## 1.5.5 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.

## 1.5.6 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\_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;
}

```

## 1.5.7 Ссылки

Данная функция приводится в следующих источниках:

1. [6] — [Rosenbrock function](#).
2. [7] — [Rosenbrock Function](#).

## 1.6 Функция Развернутый гипер-эллипсоид

### 1.6.1 Описание функции

**Идентификатор:** MHL\_TestFunction\_RotatedHyperEllipsoid.

**Наименование:** Развернутый гипер-эллипсоид.

**Тип:** Задача вещественной оптимизации.

**Формула** (целевая функция):

$$f(\bar{x}) = \sum_{i=1}^n \left( \sum_{j=1}^i \bar{x}_j \right)^2, \text{ где} \quad (1.6)$$

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

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

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

**График:** Рисунок 1.6 нас 25 стр.



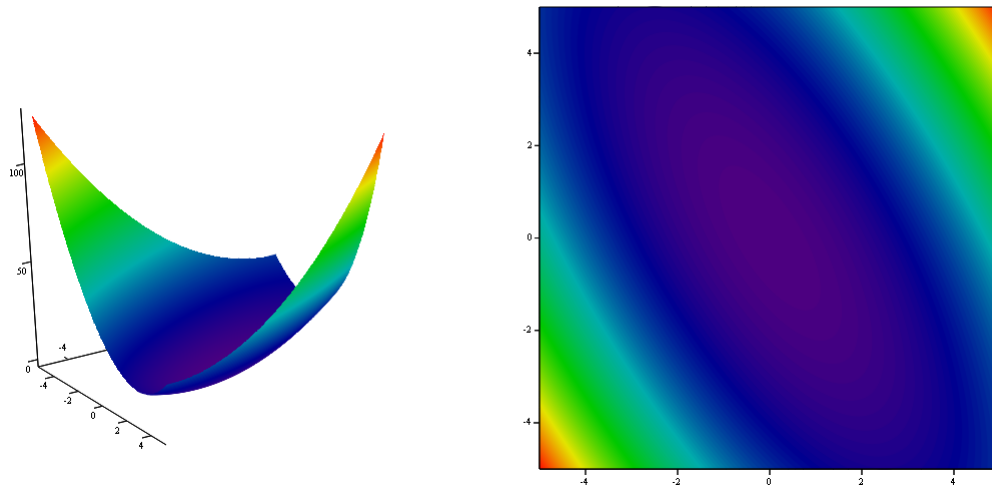


Рисунок 1.6. Развернутый гипер-эллипсоид

## 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:

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

## 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  $\bar{x}_{submin}^k$  со значениями целевой функции  $f(\bar{x}_{submin}^k)$  соответственно ( $k = \overline{1, N}$ ). Используем три вида ошибок:

**Надёжность:**

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

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

**Ошибка по входным параметрам:**

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

**Ошибка по значениям целевой функции:**

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

## 1.6.5 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.

## 1.6.6 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\_RotatedHyperEllipsoid

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

```

Функция многих переменных: Развернутый гипер-эллипсоид.
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;
double f;

for (int i=0;i<VMHL_N;i++)
{
  f=0;
  for (int j=0;j<i+1;j++)
    f += x[j];
  VMHL_Result += f*f;
}

return VMHL_Result;
}

```

### 1.6.7 Ссылки

Данная функция приводится в следующих источниках:

1. [8, стр. 4] — [GEATbx Examples. Examples of Objective Functions.](#)

Обратите внимание, что иногда под названием Rotated Hyper Ellipsoid встречается (например, [9]) неправильно записанная функция в виде:

$$f(\bar{x}) = \sum_{i=1}^n \sum_{j=1}^i \bar{x}_j^2, \text{ где}$$

Данная функция не является развернутой по своему внешнему виду, поэтому автор склонен считать эту реализацию ошибочной.

## 1.7 Аддитивная потенциальная функция

### 1.7.1 Описание функции

<b>Идентификатор:</b>	MHL_TestFunction_AdditivePotential.
<b>Наименование:</b>	Аддитивная потенциальная функция.
<b>Тип:</b>	Задача вещественной оптимизации.
<b>Формула (целевая функция):</b>	

$$f(\bar{x}) = z(\bar{x}_1) + z(\bar{x}_2), \text{ где} \quad (1.7)$$

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

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

**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:**

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

**Минимум функции:**

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

**График:**

Рисунок 1.7 нас 28 стр.

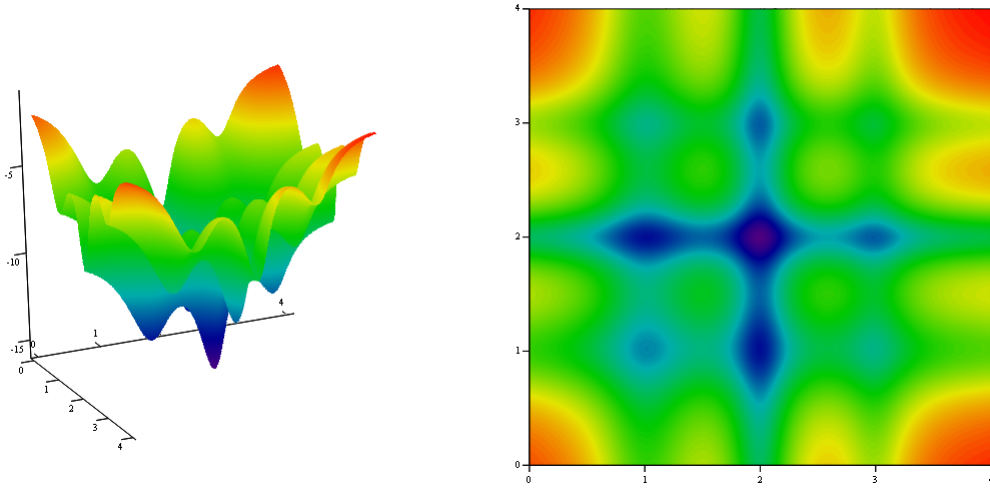


Рисунок 1.7. Аддитивная потенциальная функция

## 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:

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

### 1.7.3 Основная задача и подзадачи

<b>Изменяемый параметр:</b>	$n$ — размерность вещественного вектора.
<b>Значение в основной задаче:</b>	$n = 2$ .

### 1.7.4 Нахождение ошибки оптимизации

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

**Надёжность:**

$$R = \frac{\sum_{k=1}^N S(\bar{x}_{submin}^k)}{N}, \text{ где}$$
$$S(\bar{x}_{submin}^k) = \begin{cases} 1, & \text{если } |(\bar{x}_{submin}^k)_j - (\bar{x}_{min})_j| \leq \varepsilon, j = \overline{1, n}; \\ 0, & \text{иначе.} \end{cases}$$

**Ошибка по входным параметрам:**

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

**Ошибка по значениям целевой функции:**

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

### 1.7.5 Свойства задачи

<b>Условной или безусловной оптимизации:</b>	Задача безусловной оптимизации.
<b>Одномерной или многомерной оптимизации:</b>	Многомерной (двумерной).
<b>Функция унимодальная или многоэкстремальная:</b>	Функция многоэкстремальная.
<b>Функция стохастическая или нет:</b>	Функция не стохастическая.
<b>Особенности:</b>	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\_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;
}
```

## 1.7.7 Ссылки

Данная функция приводится в следующих источниках:

1. [10, стр. 33] — Эволюционные методы моделирования и оптимизации сложных систем.

## 1.8 Мультипликативная потенциальная функция

### 1.8.1 Описание функции

<b>Идентификатор:</b>	MHL_TestFunction_MultiplicativePotential.
<b>Наименование:</b>	Мультипликативная потенциальная функция.
<b>Тип:</b>	Задача вещественной оптимизации.
<b>Формула (целевая функция):</b>	

$$f(\bar{x}) = -z(\bar{x}_1) \cdot z(\bar{x}_2), \text{ где} \quad (1.8)$$

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

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

**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:**

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

**Минимум функции:**

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

**График:**

Рисунок 1.8 нас 31 стр.

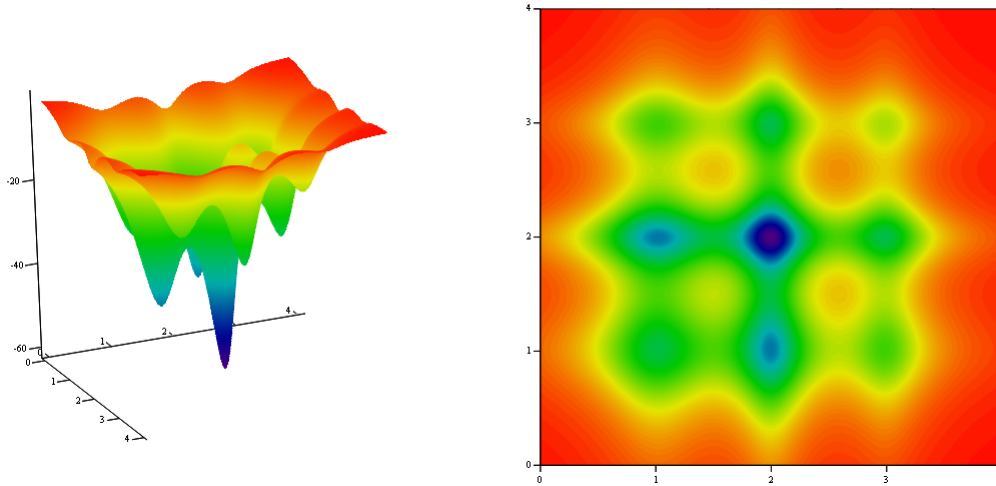


Рисунок 1.8. Мультипликативная потенциальная функция

## 1.8.2 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:

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

### 1.8.3 Основная задача и подзадачи

<b>Изменяемый параметр:</b>	$n$ — размерность вещественного вектора.
<b>Значение в основной задаче:</b>	$n = 2$ .

### 1.8.4 Нахождение ошибки оптимизации

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

**Надёжность:**

$$R = \frac{\sum_{k=1}^N S(\bar{x}_{submin}^k)}{N}, \text{ где}$$
$$S(\bar{x}_{submin}^k) = \begin{cases} 1, & \text{если } |(\bar{x}_{submin}^k)_j - (\bar{x}_{min})_j| \leq \varepsilon, j = \overline{1, n}; \\ 0, & \text{иначе.} \end{cases}$$

**Ошибка по входным параметрам:**

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

**Ошибка по значениям целевой функции:**

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

### 1.8.5 Свойства задачи

<b>Условной или безусловной оптимизации:</b>	Задача безусловной оптимизации.
<b>Одномерной или многомерной оптимизации:</b>	Многомерной: (двумерной).
<b>Функция унимодальная или многоэкстремальная:</b>	Функция многоэкстремальная.
<b>Функция стохастическая или нет:</b>	Функция не стохастическая.
<b>Особенности:</b>	Нет.



## 1.8.6 Реализация

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

Код 1.8. 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;
}
```

## 1.8.7 Ссылки

Данная функция приводится в следующих источниках:

1. [10, стр. 32] — Эволюционные методы моделирования и оптимизации сложных систем.

## 1.9 Функция ReverseGriewank

### 1.9.1 Описание функции

**Идентификатор:** MHL\_TestFunction\_ReverseGriewank.

**Наименование:** Функция ReverseGriewank.

**Тип:** Задача вещественной оптимизации.

**Формула** (целевая функция):

$$f(\bar{x}) = \frac{1}{\frac{\bar{x}_1^2 + \bar{x}_2^2}{200} - \cos(\bar{x}_0) \cos\left(\frac{\bar{x}_2}{\sqrt{2}}\right) + 2}, \text{ где} \quad (1.9)$$

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

**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:**

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

**Максимум функции:**

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

**График:**

Рисунок 1.9 нас 34 стр.

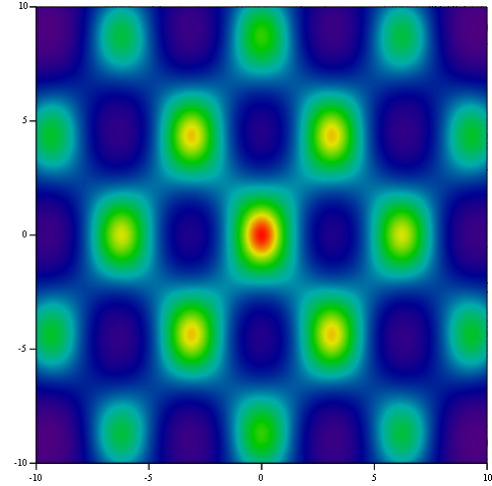
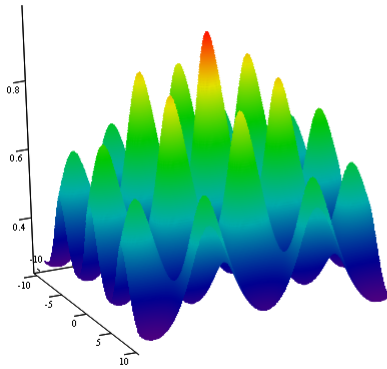


Рисунок 1.9. Функция ReverseGriewank

## 1.9.2 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:

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

### 1.9.3 Основная задача и подзадачи

<b>Изменяемый параметр:</b>	$n$ — размерность вещественного вектора.
<b>Значение в основной задаче:</b>	$n = 2$ .

### 1.9.4 Нахождение ошибки оптимизации

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

**Надёжность:**

$$R = \frac{\sum_{k=1}^N S(\bar{x}_{submin}^k)}{N}, \text{ где}$$
$$S(\bar{x}_{submin}^k) = \begin{cases} 1, & \text{если } |(\bar{x}_{submin}^k)_j - (\bar{x}_{min})_j| \leq \varepsilon, j = \overline{1, n}; \\ 0, & \text{иначе.} \end{cases}$$

**Ошибка по входным параметрам:**

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

**Ошибка по значениям целевой функции:**

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

### 1.9.5 Свойства задачи

<b>Условной или безусловной оптимизации:</b>	Задача безусловной оптимизации.
<b>Одномерной или многомерной оптимизации:</b>	Многомерной: (двумерной).
<b>Функция унимодальная или многоэкстремальная:</b>	Функция многоэкстремальная.
<b>Функция стохастическая или нет:</b>	Функция не стохастическая.
<b>Особенности:</b>	Нет.

## 1.9.6 Реализация

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

Код 1.9. 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;
}
```

## 1.9.7 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.

## 1.10 Функция волна

### 1.10.1 Описание функции

<b>Идентификатор:</b>	MHL_TestFunction_Wave.
<b>Наименование:</b>	Волна.
<b>Тип:</b>	Задача вещественной оптимизации.
<b>Формула</b> (целевая функция):	

$$f(\bar{x}) = e^{-\bar{x}_1^2} + 0.01 \cos(200 \cdot \bar{x}_1), \text{ где} \quad (1.10)$$

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

**Обозначение:**

$\bar{x}$  — вещественный вектор;

$n = 1$  — размерность вещественного вектора.

**Решаемая задача оптимизации:**

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

**Точка максимума:**

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

**Максимум функции:**

$f(\bar{x}_{max}) = 1.01$ .

**График:**

Рисунок 1.10 нас 37 стр.

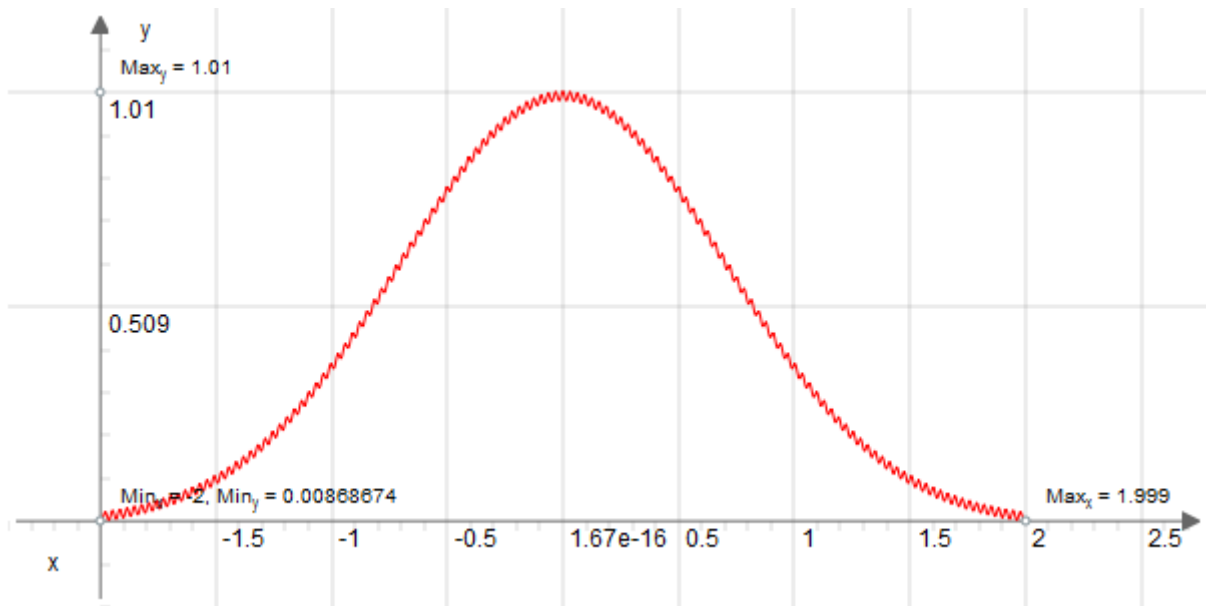


Рисунок 1.10. Волна

### 1.10.2 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:

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

### 1.10.3 Основная задача и подзадачи

<b>Изменяемый параметр:</b>	$n$ — размерность вещественного вектора.
<b>Значение в основной задаче:</b>	$n = 1$ .

### 1.10.4 Нахождение ошибки оптимизации

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

**Надёжность:**

$$R = \frac{\sum_{k=1}^N S(\bar{x}_{submin}^k)}{N}, \text{ где}$$
$$S(\bar{x}_{submin}^k) = \begin{cases} 1, & \text{если } |(\bar{x}_{submin}^k)_j - (\bar{x}_{min})_j| \leq \varepsilon, j = \overline{1, n}; \\ 0, & \text{иначе.} \end{cases}$$

**Ошибка по входным параметрам:**

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

**Ошибка по значениям целевой функции:**

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

### 1.10.5 Свойства задачи

**Условной или безусловной оптимизации:** Задача безусловной оптимизации.

**Одномерной или многомерной оптимизации:** Одномерной.

**Функция унимодальная или многоэкстремальная:** Функция многоэкстремальная.

**Функция стохастическая или нет:** Функция не стохастическая.

**Особенности:** Хотя внешне можно отнести эту функцию к стохастической, так как по поведению напоминает вид плотности нормального распределения с помехой.

### 1.10.6 Реализация

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

Код 1.10. Код функции MHL\_TestFunction\_Wave

```
double MHL_TestFunction_Wave(double x)
{
    /*
    Функция одной переменных: волна.
    Тестовая функция вещественной оптимизации.
    Входные параметры:
    x - вещественная переменная.
    Возвращаемое значение:
    Значение тестовой функции в точке (x).
    */
    double VMHL_Result;
    VMHL_Result = (exp(-x*x)+0.01*cos(200*x));
    return VMHL_Result;
}
```

### 1.10.7 Ссылки

Так и не смог найти нормальный источник для этой функции. Откуда-то у меня находится со студенческих времен.

## 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):

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

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

**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:**

$\bar{x}_{max} = (1, 1, \dots, 1)^T$ , то есть  $(\bar{x}_{max})_j = 1 (j = \overline{1, n}).$

**Максимум функции:**

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

**Точка минимума:**

$\bar{x}_{min} = (0, 0, \dots, 0)^T$ , 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

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

### 2.1.3 Нахождение ошибки оптимизации

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

**Надёжность:**

$$R = \frac{\sum_{k=1}^N S(\bar{x}_{submax}^k)}{N}, \text{ где}$$
$$S(\bar{x}_{submax}^k) = \begin{cases} 1, & \text{если } \bar{x}_{submax}^k = \bar{x}_{max}; \\ 0, & \text{иначе.} \end{cases}$$

**Ошибка по входным параметрам:**

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

**Ошибка по значениям целевой функции:**

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

## 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. Rastrigin function. [http://en.wikipedia.org/wiki/Rastrigin\\_function](http://en.wikipedia.org/wiki/Rastrigin_function).
4. Non-linear Continuous Multi-Extremal Optimization. <http://www.maths.uq.edu.au/CEToolBox/node3.html>.
5. Parametric Optimization. <http://www.pg.gda.pl/~mkwies/dyd/geadocu/fcnfun6.html>.
6. Rosenbrock function. [http://en.wikipedia.org/wiki/Rosenbrock\\_function](http://en.wikipedia.org/wiki/Rosenbrock_function).
7. Rosenbrock Function. [http://www-optima.amp.i.kyoto-u.ac.jp/member/student/hedar/Hedar\\_files/TestGO\\_files/Page2537.htm](http://www-optima.amp.i.kyoto-u.ac.jp/member/student/hedar/Hedar_files/TestGO_files/Page2537.htm).
8. Pohlheim Hartmut. GEATbx Examples. Examples of Objective Functions. 2006. [http://www.geatbx.com/download/GEATbx\\_ObjFunExpl\\_v38.pdf](http://www.geatbx.com/download/GEATbx_ObjFunExpl_v38.pdf).
9. Virtual Library of Simulation Experiments: Test Functions and Datasets. Rotated hyper-ellipsoid function. 2013. <http://www.sfu.ca/~ssurjano/rothyp.html>.
10. 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](http://files.lib.sfu-kras.ru/ebibl/umkd/22/u_lectures.pdf).