МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РФ
Федеральное государственное бюджетное образовательное
учреждение высшего образования
«Томский государственный университет систем управления и
радиоэлектроники»
Кафедра автоматизированных систем управления

А.А. МИЦЕЛЬ

ЭВРИСТИЧЕСКИЕ МЕТОДЫ ОПТИМИЗАЦИИ

Методические указания по выполнению лабораторных работ для студентов направления подготовки

09.04.01 – Информатика и вычислительная техника (магистратура)

ТОМСК – 2024

УДК 519.852(075.8)
ББК 22.183я73
М701

Мицель А.А. Эвристические методы оптимизации: Методические указания по выполнению лабораторных работ/А.А. Мицель. – Томск: ТУСУР , 2024. – 59 с.

В пособии приводится описание 5 лабораторныхработ по дисциплине «Эвристические методы оптимизации» (кодирование параметров в генетических алгоритмах, поиск минимума функции одной переменной на множестве целых чисел, поиск минимума функции одной переменной на множестве вещественных чисел с использованием кода Грэя, поиск минимума функции одной переменной с вещественным кодированием чисел, поиск минимума функции многих переменных с вещественным кодированием чисел), приводятся варианты и порядок выполнения работ. Приведены примеры программ в пакете Mathcad. В качестве индивидуального задания предусмотрено выполнение шестой лабораторной работы

Пособие подготовлено для студентов, обучающихся по направлению 09.04.01 – «информатика и вычислительная техника (магистратура)».

УДК 519.852(075.8)
ББК 22.183я73
М701

СОДЕРЖАНИЕ

Лабораторная работа №1

КОДИРОВАНИЕ ПАРАМЕТРОВ ЗАДАЧИ В ГЕНЕТИЧЕСКИХ АЛГОРИТМАХ

Цель работы:

1. Научиться кодировать и декодировать целые и вещественные числа.

1.1 Бинарное кодирование

Успех в решении задачи с помощью ГА напрямую зависит от способа кодирования её параметров. Кодирование определяет то, каким образом будут выглядеть хромосомы. Хромосомы могут представлять из себя битовые строки, вещественные числа, перестановки элементов, список правил и практически любую структуру данных. В классическом ГА, представленном Холландом [1], производится бинарное кодирование параметров. Например, целые числа из интервала от 0 до 31 можно представить последовательностями нулей и единиц, используя их представление в двоичной системе счисления. Число 0 при этом записывается как 00000, а число 31 – как 11111. В данном случае хромосомы приобретают вид двоичных последовательностей, состоящих из 5 битов, т.е. цепочками длиной 5.

Итак, будем считать, что каждая переменная x_i кодируется определенным фрагментом хромосомы, состоящим из фиксированного количества генов (см. рис. 1.1). Рядом стоящие фрагменты не отделяют друг от друга какими-либо маркерами, тем не менее, при декодировании хромосомы в вектор переменных на протяжении всего моделируемого периода эволюции используется одна и та же маска картирования.

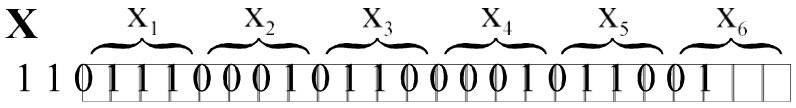

Рис. 1.1. Пример кодирования переменных задачи

Хромосомы генерируются случайным образом, путем последовательного заполнения разрядов (генов), сразу в бинарном виде. Таблица 1.1 воспроизводит в полном объеме процедуру декодирования фрагмента 4-х битовой хромосомы в проекцию вектора переменных $x_i \in [a_i, b_i]$ [2].

Таблица 1.1

Декодирование фрагментов хромосом в проекции вектора переменных [2].

Десятичное значение сдвига	Двоично-десятичный код	Код Грея	Вещественное значение координаты
0	0000	0000	a
1	0001	0001	a+1(b-a)/15
2	0010	0011	a+2(b-a)/15
3	0011	0010	a+3(b-a)/15
4	0100	0110	a+4(b-a)/15
5	0101	0111	a+5(b-a)/15

6	0110	0101	a+6(b-a)/15
7	0111	0100	a+7(b-a)/15
8	1000	1100	a+8(b-a)/15
9	1001	1101	a+9(b-a)/15
10	1010	1111	a+10(b-a)/15
11	1011	1110	a+11(b-a)/15
12	1100	1010	a+12(b-a)/15
13	1101	1011	a+13(b-a)/15
14	1110	1001	a+14(b-a)/15
15	1111	1000	b

В общем случае число интервалов будет равно 2^L-1, где L – число битов (при L=4 получим $2^4-1=15$, при L=5 получим $2^5-1=31$).

Строка $c=(c_1,...,c_L),\ c_j\in\{0,1\},\ j=1,...,L$, закодированная с помощью двоичного кода, представляет целое число $v\in\{0,...,2^L-1\}$

$$v=\sum_{j=1}^{L}c_j 2^{(L-j)}. \tag{1.1}$$

Точки разбиения интервала $[a,b]$ при двоичном кодировании определяются по формуле

$a+i\cdot(b-a)/(2^{L-1}-1),\ i=1,...,2^{L-1}-1.$

Преобразование строки $c=(c_1,...,c_L),\ c\in\{0,1\}$, закодированной двоичным кодом, в строку Грэя $g=(g_1,...,g_L)$ осуществляется по формуле [3]

$$g_k=\begin{cases}c_1,\ \textit{если}\ \ k=1,\\ c_{k-1}\oplus c_k,\ \textit{если}\ \ k>1,\end{cases} \tag{1.2}$$

где $\oplus$ – означает сложение по модулю 2. В случае бинарного и тернарного сложения по модулю 2 результаты приведены в таблице 2.2

Таблица 2.2 Сложение по модулю 2

Бинарное сложение				Тернарное сложение			
x	y	$a\oplus b$		x	y	z	$a\oplus b$
0	0	0		0	0	0	0
1	0	1		1	0	0	1
0	1	1		0	1	0	1
1	1	0		1	1	0	0
				0	0	1	1
				1	0	1	0

		0	1	1	0
		1	1	1	1

Общее правило сложения по модулю 2: результат равен 0, если нет операндов, равных 1, либо число операндов, равных 1 равно чётному количеству.

От кода Грея переходим к двоично-десятичному коду с помощью обратного преобразования по формуле

$$c_k = \sum_{i=1}^{k} g_i , \tag{1.3}$$

где суммирование выполняется по модулю 2. Далее, переходим к натуральным целым числам. Отношение полученного числа к максимальному числу, доступному для кодирования данным количеством разрядов фрагмента (в табл. 2.1 – число 15) и дает искомое значение сдвига переменной относительно левой границы a допустимого диапазона ее изменения ($b-a$).

1.2 Вещественное кодирование

Вектор параметров целевой функции $f(x),\ x \in D$ имеет компоненты $x_1, x_2, ..., x_n$, которые представляют собой искомые переменные задачи. Если вектор состоит из одной компоненты, то задача оптимизации считается одномерной, иначе – многомерной. Для оптимизации многомерных функций удобно использовать непрерывные генетические алгоритмы, то есть те, которые используют вещественное кодирование. При таком способе кодирования хромосомы представляют собой вектор вещественных чисел. Размер хромосомы равен длине вектора решения задачи, таким образом, ген представляет собой компоненту вектора решения задачи. Значения генов принимаются из интервала, на которых определены соответствующие переменные $x_i,\ i = 1,...,n$, поэтому генетические операторы должны учитывать этот момент.

Использование вещественных чисел позволяет использовать большие домены для переменных, что сложно осуществить при бинарном представлении. Еще одно преимущество в том, что при незначительных изменениях в переменных происходят небольшие изменения значения функции. Использование вещественного кодирования очень близко к естественной формулировке многих задач, что стирает разницу между генотипом и фенотипом. Из-за отсутствия этой разницы пропадает необходимость использовать дополнительные процессы кодирования и декодирования, что увеличивает скорость работы алгоритма [1].

Алгоритм вещественного кодирования состоит в генерировании с помощью равномерного датчика случайных чисел x_i из интервала $[a_i, b_i],\ i = 1,...,n$, которые являются координатами вектора x.

1.3 Практическое задание

а) Написать в пакете Mathcad программы, описанные в таблицах 1.3, 1.4.

Таблица 1.3 - Программы для кодирования и декодирования целых чисел

Идентификатор программы	Описание	Входные параметры	Выходные параметры
Ц1	Преобразование бинарных чисел	Матрица бит в двоичной СС	Матрица бит в коде Грея
Ц2		Матрица бит в коде Грея	Матрица бит в двоичной СС
Ц3	Перевод чисел из десятичной СС в двоичную	Вектор-столбец целых чисел в десятичной СС	Матрица бит в двоичной СС
Ц4			Матрица бит в коде Грея
Ц5	Перевод чисел из двоичной	Матрица бит в	Вектор-столбец

Ц6	СС в десятичную	двоичной СС Матрица бит в коде Грея	целых чисел в десятичной СС

Таблица 1.4 - Программы для кодирования и декодирования вещественных чисел

Идентификатор программы	Описание	Входные параметры	Выходные параметры
В1	Перевод чисел из десятичной СС в двоичную	– Вектор-столбец вещественных чисел в десятичной СС, – длина битовой строки L, – левая и правая границы интервала [a;b]	Матрица бит в двоичной СС
В2			Матрица бит в коде Грея
В3	Перевод чисел из двоичной СС в десятичную	– Матрица бит в двоичной СС, – левая и правая границы интервала [a;b]	Вектор-столбец вещественных чисел в десятичной СС
В4		– Матрица бит в коде Грея, – левая и правая границы интервала [a;b]	

Полученный комплекс программ должен реализовывать треугольник преобразований (рисунок 1.2).

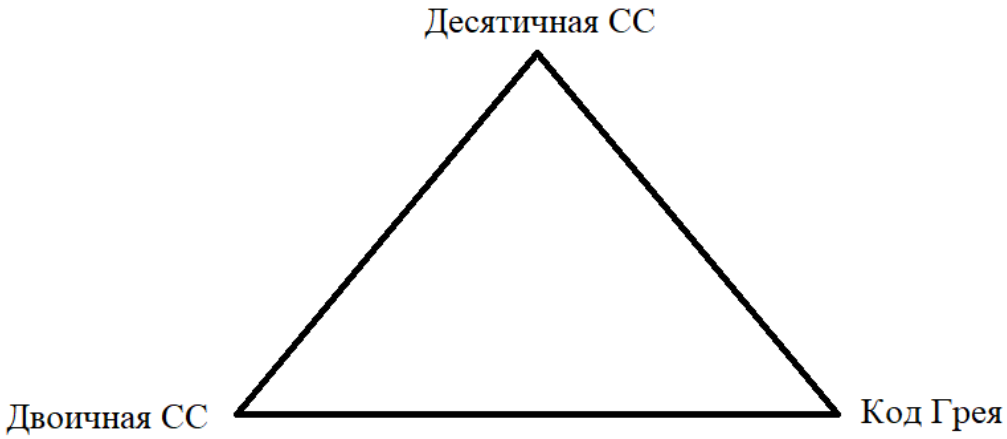

Рис.1.2 - Треугольник преобразования чисел

b) Написать в пакете Mathcad программу, генерирующую значения из заданного интервала (таблица 1.5).

Таблица 1.5 - Программа для генерации чисел в определённых интервалах

Идентификатор программы	Описание	Входной параметр	Выходной параметр

Г1	Генерация чисел в десятичной СС	атрица интервалов [a;b]	Вектор-столбец вещественных чисел в десятичной СС

Примеры программ представлены в приложении к этой лабораторной работе.

Литература

1 Holland J.H. Adaptation in Natural and Artificial Systems. – The University of Michigan Press, University of Michigan, Ann Arbor, 1975.

2 Вороновский Г.К. Генетические алгоритмы, искусственные нейронные сети и проблемы виртуальной реальности / Г. К. Вороновский, К. В. Махотило, С. Н. Петрашев, С. А. Сергеев.– Х.: ОСНОВА, 1997.– 112 с.

3 Herrera F., Lozano M., J.L. Verdegay. Tackling Real-Coded Genetic Algorithms: Operators and Tools for Behavioural Analysis // Artificial Intelligence Review, vol. 12 Issue 4, Aug. 1, 1998. – P. 265- 319.

Приложение к лабораторной работе №1

Примеры программ в пакете Mathcad

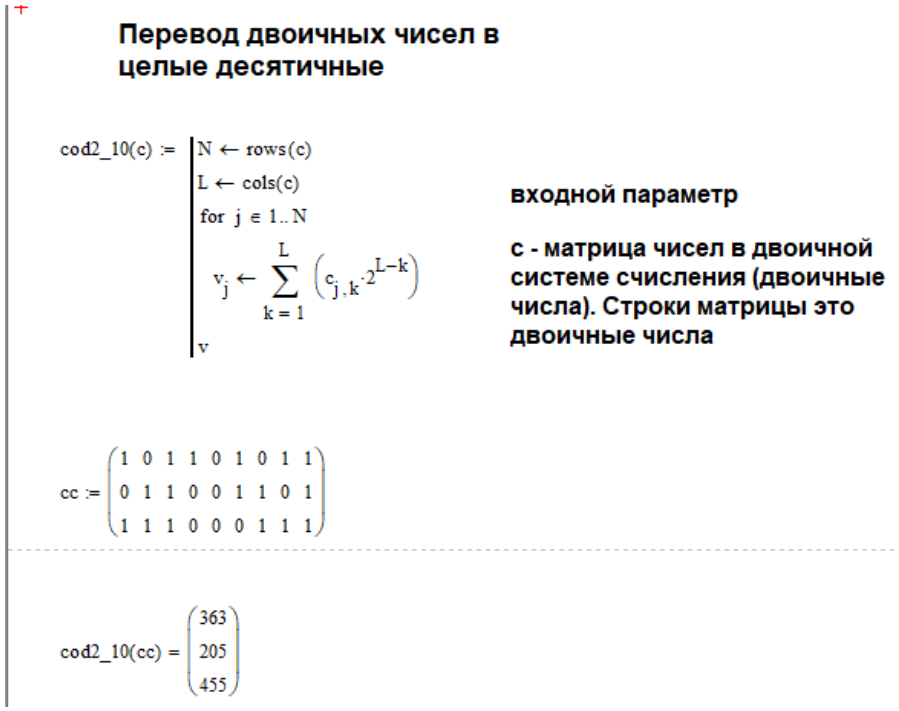

Рис. П.1 - Пример программы декодирования двоичных чисел в целые десятичные

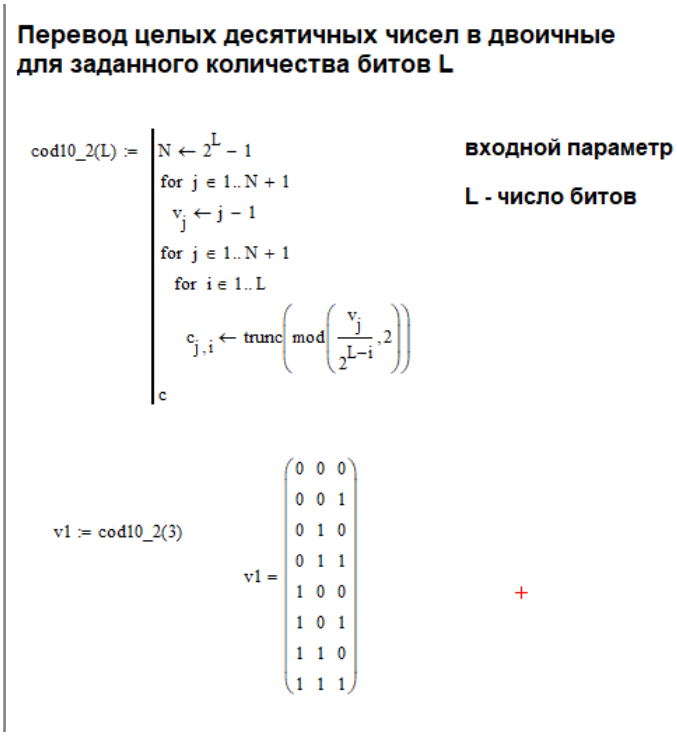

Рис. П.2 - Пример программы кодирования десятичных целых чисел в двоичные

```
bin_to_grey(c) := | N ← rows(c) − 1
                  | L ← cols(c) − 1
                  | for i ∈ 0..N
                  |   g_{i,0} ← c_{i,0}
                  |   for j ∈ 1..L
                  |     g_{i,j} ← 0 if (c_{i,j−1} ⊕ c_{i,j}) = 0
                  |     g_{i,j} ← 1 otherwise
                  | g
```

$$\text{bin\_to\_grey}\left(\begin{pmatrix}0&0&1&1\\0&1&1&0\\1&0&0&1\end{pmatrix}\right)=\begin{pmatrix}0&0&1&0\\0&1&0&1\\1&1&0&1\end{pmatrix}$$

Рис. П.3 - Пример программы Ц1

```
grey_to_bin(g) := | N ← rows(g) − 1
                  | L ← cols(g) − 1
                  | for i ∈ 0..N
                  |   for j ∈ 0..L
                  |     c_{i,j} ← 0 if (Σ_{k=0}^{j} g_{i,k} = 0) ∨ mod([(Σ_{k=0}^{j} g_{i,k})], 2) = 0
                  |     c_{i,j} ← 1 otherwise
                  | c
```

Комментарий:
Можно ли как-то избавиться от ИЛИ в условии, при котором c(i,j)=0?
Подсказка: можно

Рис. П.4 - Пример программы Ц2

```
dec_to_bin(v) := | L ← rows(v) − 1
                 | N ← trunc(log(max(v), 2))
                 | for i ∈ 0..L
                 |   for j ∈ 0..N
                 |     c_{i,j} ← trunc(mod(v_i / 2^{N−j}, 2))
                 | c
```

$$\text{dec\_to\_bin}\left(\begin{pmatrix}4\\8\\10\\5\end{pmatrix}\right)=\begin{pmatrix}0&1&0&0\\1&0&0&0\\1&0&1&0\\0&1&0&1\end{pmatrix}$$

Рис. П.5 - Пример программы Ц3

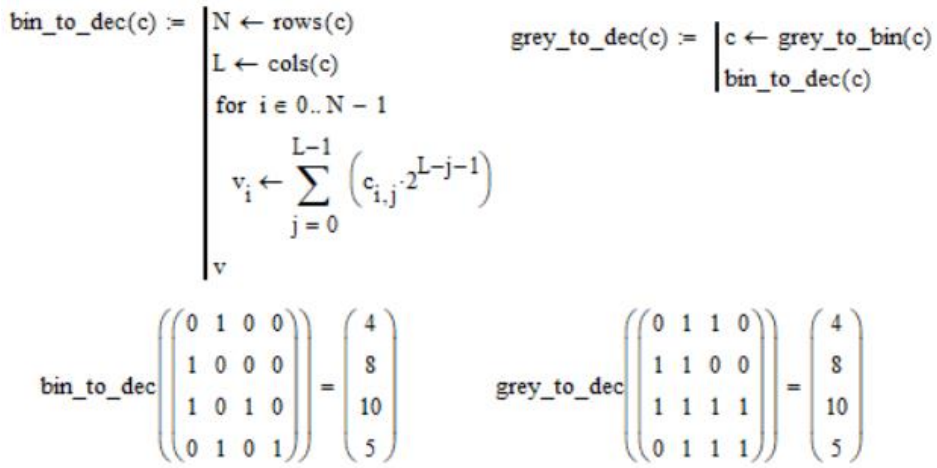

Рис. П.6 - Пример программ Ц5 и Ц6

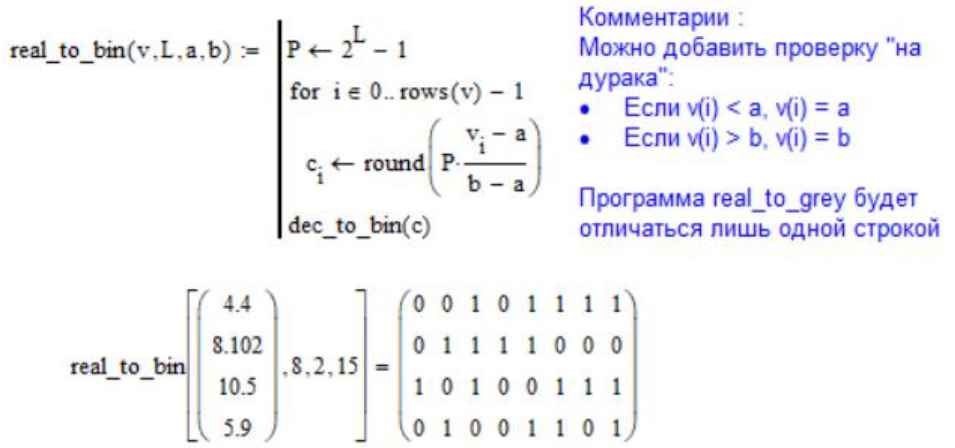

Рис. П.7 - Пример программы В1

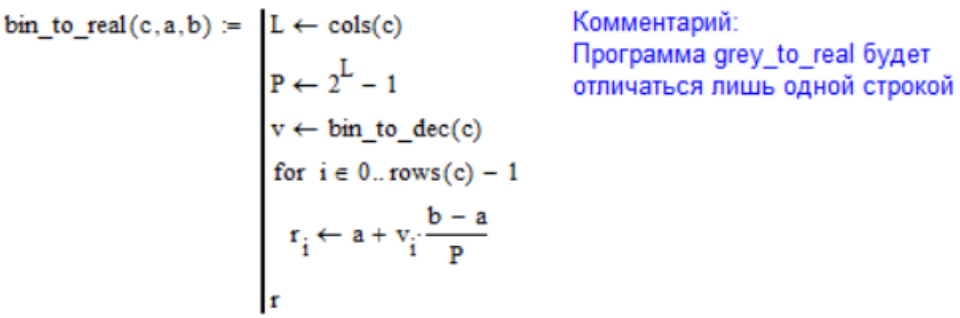

Рис. П.8 - Пример программы В3

$$\text{gen\_real}(\text{borders}) := \left| \begin{array}{l} N \leftarrow \text{rows}(\text{borders}) - 1 \\ \text{for } i \in 0..N \\ \quad v_i \leftarrow \text{borders}_{i,0} + \text{rnd}(1) \cdot \left(\text{borders}_{i,1} - \text{borders}_{i,0}\right) \\ v \end{array} \right.$$

$$\text{gen\_real}\left(\begin{pmatrix} 0 & 100 \\ 0 & 1000 \\ 2 & 3 \\ 4 & 4.1 \end{pmatrix}\right) = \begin{pmatrix} 14.731 \\ 988.508 \\ 2.119 \\ 4.001 \end{pmatrix}$$

Рис. П.9 - Пример программы Г1

Вещественное кодирование чисел

ORIGIN := 1

n := 3

$$\text{xmin} := \begin{pmatrix} -2.5 \\ 2.6 \\ -5 \end{pmatrix} \qquad \text{xmax} := \begin{pmatrix} 2.5 \\ 10.6 \\ 5 \end{pmatrix}$$

$$\text{xkk}(a,b) := \left| \begin{array}{l} n \leftarrow \text{rows}(a) \\ \text{for } i \in 1..n \\ \quad \text{xk}_i \leftarrow a_i + \text{rnd}(1) \cdot \left(b_i - a_i\right) \\ \text{xk} \end{array} \right.$$

Входные данные

a - массив нижней границы чисел
b - массив верхней границы чисел

$$\text{xk} := \text{xkk}(\text{xmin}, \text{xmax}) \qquad \text{xk} = \begin{pmatrix} -2.494 \\ 4.147 \\ 0.85 \end{pmatrix}$$

Рис. П.10 - Пример программы вещественного кодирования

Перевод двоичных чисел в десятичные

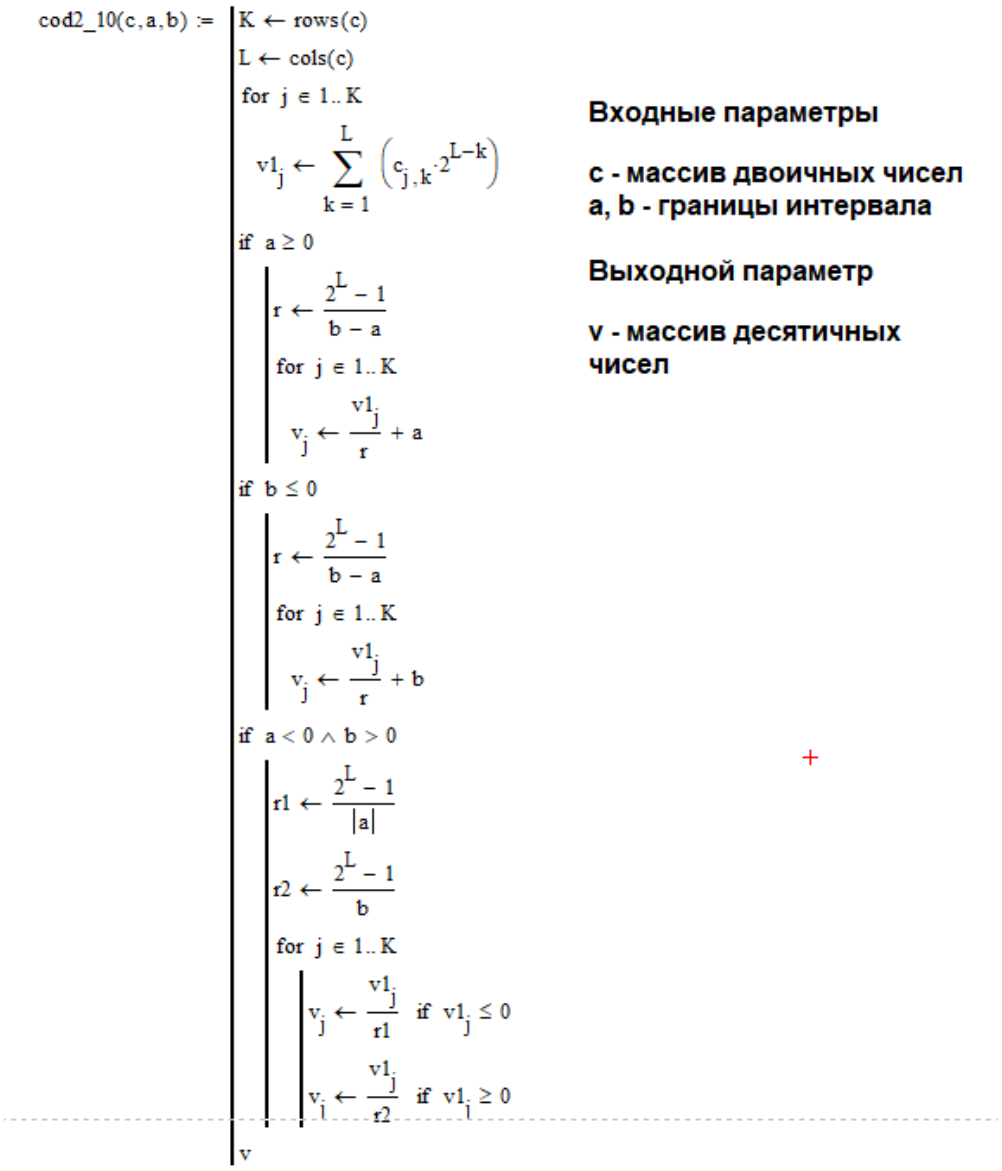

Рис. П.11 - Пример программы расчета фенотипов (декодирование)

Лабораторная работа №2

ПОИСК МИНИМУМА ФУНКЦИИ ОДНОЙ ПЕРЕМЕННОЙ НА МНОЖЕСТВЕ ЦЕЛЫХ ЧИСЕЛ

Цель работы:

2. Научиться вычислять приближенное решение задачи одномерной оптимизации на множестве целых чисел с помощью генетических алгоритмов

2.1 Основные термины генетических алгоритмов

Ген – атомарный элемент хромосомы (может быть битом, числом или неким другим объектом.

Хромосома (цепочка) – упорядоченная последовательность генов (строка из каких-либо чисел). Если эта строка представлена бинарной строкой из нулей и единиц, например, 101010, то она получена либо с использованием двоичного кодирования, либо кода Грея. Каждая позиция хромосомы называется *геном*.

Генотип(код) – упорядоченная последовательность хромосом (одно из решений).

Фенотип – набор значений, соответствующих генотипу.

Особь (индивидуум) – конкретный экземпляр генотипа (вариант решения задачи). Особи представляются хромосомами с закодированными в них множествами параметров задачи, т.е. решений. Обычно особь состоит из одной хромосомы, поэтому в дальнейшем особь и хромосома идентичные понятия.

Популяция – набор особей (набор решений задачи). В начале алгоритма случайным образом генерируется начальная популяция (набор решений). Эти решения будут становиться лучше (эволюционировать) в процессе работы алгоритма до тех пор, пока не удовлетворят условиям задачи.

Кроссинговер (кроссовер) – операция, при которой две хромосомы обмениваются своими частями. Например, 1100&1010 → 1110&1000. Здесь произошел обмен вторым геном (вторым младшим разрядом).

Мутация – случайное изменение одной или нескольких позиций в хромосоме. Например, 1010011 →1010001.

Инверсия – изменение порядка следования битов в хромосоме или в ее фрагменте. Например, 1100 → 0011.

Рекомбинация – операция, при которой две хромосомы обмениваются своими частями.

Функция приспособленности. Одним из центральных понятий является *функция приспособленности* или *функция пригодности*. Она оценивает то, насколько приспособлена данная особь в популяции, другими словами, она определяет качество особей популяции.

Функции приспособленности всегда зависит от задачи и в её роли часто выступает целевая функция [24]. Иногда оценку приспособленности проводят в две стадии. Первая стадия – это собственно оценка $f_k(x) = f_k(x_1^k, x_2^k, ..., x_n^k)$. Вторая - дополнительные преобразования. Например, ею может быть нормировка к виду $F_k = \frac{f_k - f_k^0}{f_k^1 - f_k^0}$, где f_k^1, f_k^0 – соответственно, лучший и худший показатели в текущей популяции.

2.2 Одномерная оптимизация на множестве целых чисел

Генетические алгоритмы состоят из четырех основных операторов: селекции, кроссинговера (скрещивания, репродукции), мутации, создания нового поколения. Цикл скрещивания и мутации с последующей оценкой приспособленности называется поколением. Поэтапно алгоритм процесса формирования нового поколения можно представить так:

Шаг 1. Создать начальную популяцию s_k^0, $k = 1,...,K$ из K хромосом (особей). Для

этого случайным образом генерируется K двоичных чисел из интервала $[0; 2^L - 1]$, где L – длина хромосомы. Длина битовой строки $L = 5$ (длина хромосомы).

Шаг 2. Произвести расчет пробных решений x^k, $x^k \in [a, b]$ путем декодирования двоичных чисел (начальное поколение) по формуле $x^k = \frac{e^{-1}(s_k^0)}{r} + a$, если $a \geq 0$, либо $x^k = \frac{e^{-1}(s_k^0)}{r} + b$, если $b \leq 0$. Здесь $r = \frac{2^L - 1}{b - a}$; a и b границы интервала. Если $a < 0$, $b > 0$, то для отрицательных двоичных чисел $r^- = \frac{2^L - 1}{|a|}$, $x^k = \frac{e^{-1}(s_k^0)}{r^-}$, а для положительных – $r^+ = \frac{2^L - 1}{b}$, $x^k = \frac{e^{-1}(s_k^0)}{r^+}$. Если целые числа границ интервала $|a| \neq 2^L - 1$ и $|b| \neq 2^L - 1$, то значения фенотипов x^k будут вещественными.

Оценить степень приспособленности каждой особи (рассчитать значения целевой функции $f_k(x) = f(x^k)$.

Шаг 3. Выбрать K родителей из популяции при помощи метода селекции (вероятность выбора родителя должна зависеть от степени его приспособленности).

Шаг 4. Выбрать из родительского пула пару родителей для репродукции. При помощи оператора кроссинговера получить потомка.

Шаг 5. Подвергнуть потомков мутации с вероятностью $p_m = \frac{1}{L}$.

Шаг 6. Выполнить элитарный отбор особей.

Шаг 7. Повторяем шаги 5–6, пока не будет сгенерировано новое поколение популяции, содержащее K хромосом

Шаг 8. Оценить степень приспособленности каждой особи в новой популяции.

Шаг 9. Выход, если выполняется критерий останова, иначе на шаг 3.

Шаг 10. Перейти к шагу 3, если количество поколений не превышает допустимого.

2.2.1 Оптимизация с помощью двоичного кода

Рассмотрим применение алгоритма на следующем примере. Найти минимум функции $f(x) = 2x^2 + 1$ для целочисленной переменной $x \in [-31, 31]$.

Шаг. 1

Сформируем начальную популяцию из 6 особей. Закодируем значения переменной в виде двоичных последовательностей.

Пример программы формирования начальной популяции на множестве целых чисел $x \in [-31, 31]$ в Mathcad приведен на рис. 2.1.

Формирование начальной популяции

```
ss0(a,b,L,K) := | for j ∈ 1..K
                |   z_j ← round(md(10)/10)
                |   z_j ← 0  if a ≥ 0
                |   z_j ← 1  if b ≤ 0
                | k1 ← 1  if a ≥ 0
                | k1 ← −1  if a < 0 ∨ b ≤ 0
                | for j ∈ 1..K
                |   for i ∈ 1..L
                |     c0_{j,i} ← (k1)^{z_j} · round(md(10)/10)
                | c0
```

Входные параметры

a, b - границы интервала
L - длина хромосомы
K - количество особей

Выходной параметр

c0 - массив особей начальной популяции

c0 := ss0(a,b,L,K)

$$c0 = \begin{pmatrix} 0 & 0 & 0 & 1 & 0 \\ 0 & -1 & -1 & 0 & 0 \\ 0 & 1 & 1 & 1 & 1 \\ -1 & 0 & -1 & -1 & -1 \\ 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & -1 & -1 & 0 \end{pmatrix}$$

Рис. 2.1 Пример программы формирования начальной популяции

Шаг 2. Оцениваем степень приспособленности каждой особи

x0 := cod2_10(c0,a,b)

ff0 := f(x0)

$$x0 = \begin{pmatrix} 2 \\ -12 \\ 15 \\ -23 \\ 21 \\ -6 \end{pmatrix} \qquad ff0 = \begin{pmatrix} 9 \\ 289 \\ 451 \\ 1.059 \times 10^3 \\ 883 \\ 73 \end{pmatrix}$$

$$fs0 := \frac{\sum_{m=1}^{K} ff0_m}{K} \qquad fs0 = 460.667$$

Рис. 2.2 Оценка приспособленности особей

Шаг 3. Выбрать K родителей из популяции при помощи метода селекции (вероятность выбора родителя должна зависеть от степени его приспособленности). Веса рассчитываются по формуле

$$w_s = \frac{|f(x_s)|}{\sum_{i=1}^{K} |f(x_s)|}$$

$$p_s(x) = w_s \cdot 100.$$

При поиске минимума функции вероятности можно вычислять по формуле

$$p_i = \frac{1/(0{,}1 + w_i)}{\sum_{j=1}^{K} 1/(0{,}1 + w_j)} \cdot 100, \ i = 1, \ldots, K.$$

$$p = \begin{pmatrix} 32 \\ 16 \\ 12 \\ 7 \\ 8 \\ 26 \end{pmatrix}.$$

Примеры программ показаны на рис. 2.3 и 2.4.

Выбор родителей с помощью рулетки

```
ruletka(w) := | n ← rows(w)
              | for j ∈ 1..n
              |   k_j ← j
              | for j ∈ 1..n
              |   r_j ← round(rnd(100))
              | for j ∈ 1..n
              |   for i ∈ 1..n
              |     if r_j ≥ Σ_{m=1}^{i} w_m
              |       | k_j ← i
              |       | continue
              | k

k := ruletka(w)
```

Выходной параметр

k - номера выбранных особей-родителей

Входной параметр

w - массив вероятностей

Рис. 2.3 Пример программы Рулетка

Результаты расчета (при каждом запуске программы результаты меняются)

$$k = \begin{pmatrix} 1 \\ 1 \\ 2 \\ 5 \\ 5 \\ 2 \end{pmatrix}$$

Выбор особей для скрещивания

```
vibor(kr, c0, a, b) := | n ← rows(kr)
                       | L ← cols(c0)
                       | for j ∈ 1..n
                       |   for i ∈ 1..L
                       |     c1[j,i] ← c0[kr_j, i]
                       | v ← cod2_10(c1, a, b)
                       | ff ← f(v)
                       | a ← sort(ff)
                       | for j ∈ 1..n
                       |   for i ∈ 1..n
                       |     kl1_i ← kr_j  if ff_j = a_i
                       | r ← 0
                       | for j ∈ 1..n/2
                       |   for i ∈ 1..L
                       |     cr[j+r,i] ← c0[kl1_j, i]
                       |     cr[j+1+r,i] ← c0[kl1_{j+1}, i]
                       |   r ← r + 1
                       | cr
```

Входные параметры

kr - массив номеров особей для скрещивания
c0 - массив особей для скрещивания в двоичном коде
a, b - границы интервала

Выходной параметр

cr - массив выбранных особей-родителей

$$c0 := \begin{pmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & -1 & -1 & -1 \\ 1 & 1 & 0 & 1 & 1 \\ -1 & -1 & 0 & -1 & 0 \\ 1 & 0 & 0 & 1 & 1 \end{pmatrix}$$

c1 := mut(c0)

Рис. 2.4. Пример программы выбора особей для скрещивания

Результаты работы программы выбора особей.

$$cr = \begin{pmatrix} 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 1 & 0 \\ 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 \end{pmatrix}; \quad xr = \begin{pmatrix} 6 \\ 6 \\ 6 \\ 16 \\ 16 \\ 16 \end{pmatrix}.$$

Шаг 4. Выбрать из родительского пула пару родителей для репродукции. При помощи оператора кроссинговера получить потомка.

Особенностью оператора скрещивания хромосом на множестве положительных и отрицательных чисел состоит в том, что если идет обмен генами хромосом, соответствующих отрицательным и положительным значениям фенотипа, то значения генов для обеих хромосом принимаются равными нулю.

Скрещивание

```
cros(cr) := | n ← rows(cr)
            | L ← cols(cr)
            | for j ∈ 1..n
            |   k_j ← 1 if Σ_{i=1}^{L} cr_{j,i} ≥ 0
            |   k_j ← −1 otherwise
            | for j ∈ 1..n
            |   for i ∈ 1..L
            |     cs_{j,i} ← cr_{j,i}
            | for j ∈ 1..n/2
            |   rs_j ← round(md(L))
            |   rs_j ← 1 if rs_j = 0
            | r ← 0
            | for j ∈ 1..n/2
            |   cs_{j+r,rs_j} ← cr_{j+1+r,rs_j}
            |   cs_{j+1+r,rs_j} ← cr_{j+r,rs_j}
            |   if k_{j+r}·k_{j+r+1} = −1
            |     cs_{j+r,rs_j} ← 0
            |     cs_{j+1+r,rs_j} ← 0
            |   r ← r + 1
            | cs

cp := cros(cr)
```

Входной параметр

cr - массив особей для скрещивания в двоичном коде

Выходной параметр

cs - массив особей потомков

Рис. 2.5 Пример программы генетического оператора Кроссинговера

Результаты расчета (при каждом запуске программы результаты меняются)

$$cp = \begin{pmatrix} 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 1 & 0 \\ 1 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 \end{pmatrix}; \quad xp = \begin{pmatrix} 6 \\ 6 \\ 22 \\ 0 \\ 16 \\ 16 \end{pmatrix}.$$ Среднее значение $\overline{f} = 357$.

Шаг 5. Подвергнуть потомков мутации с вероятностью p_m.

Мутация

```
mut(c) := | n ← rows(c)
          | L ← cols(c)
          | for j ∈ 1..n
          |   for i ∈ 1..L
          |     c1[j,i] ← c[j,i]
          | for j ∈ 1..n
          |   | k1[j] ← 1  if Σ_{i=1}^{L} c[j,i] ≥ 0
          |   | k1[j] ← −1  otherwise
          | p ← 1/L
          | for j ∈ 1..n
          |   | x ← rnd(1)
          |   | if x < p
          |   |   | k ← trunc(rnd(L)) + 1
          |   |   | c1[j,k] ← 0  if c[j,k] = 1 ∨ c[j,k] = −1
          |   |   | c1[j,k] = k1[j]  otherwise
          | c1

cm := mut(cp)
```

Входной параметр

c - массив особей для мутации в двоичном коде

Выходной параметр

c1 - массив мутированных особей

Рис. 2.6 Пример программы генетического оператора Мутация

Результаты расчета (при каждом запуске программы результаты меняются)

$$cm = \begin{pmatrix} 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 1 & 0 \\ 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 \end{pmatrix}; \quad xm = \begin{pmatrix} 6 \\ -7 \\ -5 \\ 19 \\ 16 \\ 19 \end{pmatrix}.$$ Среднее значение $\overline{f} = 243,667$.

Шаг 6. Выполнить элитарный отбор особей.
На рис. 2.7 приведен пример программы элитарного отбора особей в новую популяцию.

Элитарный отбор особей в новую популяцию

```
elitarvibor(cr, cm, a, b) :=
    n ← rows(cr)
    L ← cols(cr)
    vr ← cod2_10(cr, a, b)
    ffr ← f(vr)
    ar ← sort(ffr)
    for j ∈ 1..n
      for i ∈ 1..n
        kr_i ← j  if  ffr_j = ar_i
    vm ← cod2_10(cm, a, b)
    ffm ← f(vm)
    am ← sort(ffm)
    for j ∈ 1..n
      for i ∈ 1..n
        km_i ← j  if  ffm_j = am_i
    for j ∈ 1..n/2
      for i ∈ 1..L
        cn_j,i ← cr_krj,i
        cn_j+n/2,i ← cm_kmj,i
    cn

cn := elitarvibor(cr, cm, a, b)
```

Входные параметры

cr - массив выбранных особей-родителей

cm - массив особей после мутации (особи потомки)

a,b - границы интервала

Выходной параметр

cn - массив выбранных особей, составленный из лучших особейродителей и лучших особей потомков

Рис. 2.7 Элитарный отбор особей

Результаты расчета (при каждом запуске программы результаты меняются)

$$cn=\begin{pmatrix}0&0&1&1&0\\0&0&1&1&0\\0&0&1&1&0\\0&0&0&0&0\\0&0&0&0&0\\0&0&1&1&0\end{pmatrix};\ xn=\begin{pmatrix}6\\6\\6\\0\\0\\6\end{pmatrix};\ f(xn)=\begin{pmatrix}73\\73\\73\\1\\1\\73\end{pmatrix}.$$ Среднее значение $\overline{f}=49$.

2.3 Практическое задание

Написать программу поиска минимума функции одной переменной на множестве целых чисел с помощью генетического алгоритма. Программа может быть написана на любом алгоритмическом языке.

Входные данные:

- целевая функция;
- границы интервала поиска минимума функции $[a,b]$, где a и b – целые числа;
- количество особей в популяции K;
- количество поколений N (количество итераций);

- параметр окончания поиска ε (точность решения).

Варианты заданий

1	$f(x)=\sqrt{1+x^2}+e^{-2x},\ x\in[-7;7],\ \varepsilon=0,001$
2	$f(x)=x^4+2x^2+4x+1,\ x\in[-7;7],\ \varepsilon=0,001$
3	$f(x)=x^4-10\cdot x^3+36\cdot x^2+5\cdot x;\quad x\in[-7;7],\ \varepsilon=0,001$
4	$f(x)=\ln(1+x^2)-\sin x;\quad x\in[-7;7],\ \varepsilon=0,001$
5	$f(x)=\frac{1}{4}\cdot x^4+x^2-8\cdot x+12;\quad x\in[-7;7],\ \varepsilon=0,001$
6	$f(x)=x^3-x+1;\quad x\in[-7;7],\ \varepsilon=0,001$
7	$f(x)=2x^2+16/x;\quad x\in[0;7],\ \varepsilon=0,001$
8	$f(x)=4x^3+2x-3x^2+e^{x/2},\ x\in[0;7],\ \varepsilon=0,001$
9	$f(x)=24x-2\cdot x^2,\ x\in[0;15],\ \varepsilon=0,001$
10	$f(x)=x^4-12\cdot x^3+47\cdot x^2-60\cdot x;\ x\in[0;7],\ \varepsilon=0,001$
11	$f(x)=10x^3+3x^2+x+5;\ x\in[-7;7],\ \varepsilon=0,001$
12	$f(x)=x^2+e^{-x};\ x\in[-7;7],\ \varepsilon=0,001$
13	$f(x)=2x+e^{-x};\ x\in[-7;7],\ \varepsilon=0,001$
14	$f(x)=x^2+x+\sin x;\ x\in[-7;7],\ \varepsilon=0,001$
15	$f(x)=x^2-x+e^{-x};\ x\in[-7;7],\ \varepsilon=0,001$
16	$f(x)=-\left(2x^2-5-2^x\right);\ x\in[-7;7],\ \varepsilon=0,001$
17	$f(x)=\frac{1}{3}x^3+2x^2-5x+6;\ x\in[-7;7],\ \varepsilon=0,001$
18	$f(x)=-\left(-x^2+5x-6\right);\ x\in[-7;7],\ \varepsilon=0,001$

Лабораторная работа №3

ПОИСК МИНИМУМА ФУНКЦИИ ОДНОЙ ПЕРЕМЕННОЙ НА МНОЖЕСТВЕ ВЕЩЕСТВЕННЫХ ЧИСЕЛ С КОДИРОВАНИЕМ КОДОМ ГРЭЯ

Цель работы:

3. Научиться вычислять приближенное решение задачи одномерной оптимизации на множестве вещественных чисел с помощью генетических алгоритмов

3.1 Бинарное кодирование вещественных чисел с помощью кода Грэя

Будем считать, что переменная x представляется в виде хромосомы, состоящей из фиксированного количества генов L.

Хромосомы генерируются случайным образом, путем последовательного заполнения разрядов (генов), сразу в бинарном виде.

Число интервалов будет равно 2^L-1, где L – число битов (при L=4 получим $2^4-1=15$, при L=5 получим $2^5-1=31$).

Строка $c=(c_1,...,c_L),\ c_j\in\{0,1\},\ j=1,...,L$, закодированная с помощью двоичного кода, представляет целое число $v\in\{0,...,2^L-1\}$

$$v=\sum_{j=1}^{L}c_j 2^{(L-j)}\ . \tag{3.1}$$

При кодировании отрицательных чисел получаем строку $c=(c_1,...,c_L)$, где

$c_j\in\{0,-1\},\ j=1,...,L$.

Значение вещественного числа из интервала $[a,b]$ при двоичном кодировании определяются по формуле $x=a+v\cdot(b-a)/(2^{L-1}-1)$ для положительных чисел v. Если $b\le 0$, то для отрицательных чисел v имеем $x=b+v\cdot(b-a)/(2^{L-1}-1)$. Если граница a отрицательная, а граница b положительная, то для отрицательных чисел v получим $x=|a|\cdot v/(2^{L-1}-1)$, а для положительных чисел v имеем $x=b\cdot v/(2^{L-1}-1)$.

Преобразование строки $c=(c_1,...,c_L),\ c\in\{0,1\}$, закодированной двоичным кодом, в строку Грэя $g=(g_1,...,g_L)$ осуществляется по формуле [3]

$$g_k=\begin{cases}c_1,\ \textit{если}\ \ k=1,\\ c_{k-1}\oplus c_k,\ \textit{если}\ \ k>1,\end{cases} \tag{3.2}$$

где $\oplus$ – означает сложение по модулю 2. В случае бинарного и тернарного сложения по модулю 2 результаты приведены в таблице 3.2

Таблица 3.2 Сложение по модулю 2

Бинарное сложение				Тернарное сложение			
x	y	$a\oplus b$		x	y	z	$a\oplus b$
0	0	0		0	0	0	0
1	0	1		1	0	0	1
0	1	1		0	1	0	1
1	1	0		1	1	0	0
				0	0	1	1

		1	0	1	0
		0	1	1	0
		1	1	1	1

Общее правило сложения по модулю 2: результат равен 0, если нет операндов, равных 1, либо число операндов, равных 1 равно чётному количеству.

От кода Грея переходим к двоично-десятичному коду с помощью обратного преобразования по формуле

$$c_k = \sum_{i=1}^{k} g_i , \qquad (3.3)$$

где суммирование выполняется по модулю 2. Далее, переходим к натуральным целым числам. Отношение полученного числа к максимальному числу, доступному для кодирования данным количеством разрядов фрагмента дает искомое значение сдвига переменной относительно левой границы a допустимого диапазона ее изменения ($b-a$).

3.2 Одномерная оптимизация на множестве вещественных чисел

Генетические алгоритмы состоят из четырех основных операторов: селекции, кроссинговера (скрещивания, репродукции), мутации, создания нового поколения. Цикл скрещивания и мутации с последующей оценкой приспособленности называется поколением.

Рассмотрим применение алгоритма на следующем примере. Найти минимум функции $f(x) = 2x^2 + 1$ для целочисленной переменной $x \in [-5,5;\ 5,5]$.

Поэтапно алгоритм процесса формирования нового поколения можно представить так:

Шаг 1. Создать начальную популяцию $s_k^0,\ k = 1,...,K$ из K хромосом (особей). Для этого случайным образом генерируется K чисел, закодированных кодом Грэя из интервала $[0;\ 2^L - 1]$, где L – длина хромосомы. Пример программы в пакете Mathcad приведен на рис. 3.1

Результат расчета для начальной популяции на интервале $[-5,5;\ 5,5]$ для $L = 5$ в коде Грэя приведен ниже. Так как в программе используется датчик случайных чисел, то при каждом запуске программы результаты меняются.

$$g0 = \begin{pmatrix} 0 & 0 & -1 & 0 & -1 \\ 0 & 1 & 1 & 0 & 1 \\ 0 & -1 & -1 & -1 & -1 \\ -1 & -1 & -1 & -1 & -1 \\ -1 & -1 & -1 & 0 & -1 \\ -1 & 0 & -1 & -1 & -1 \end{pmatrix}.$$

Формирование начальной популяции

```
ss0(a,b,L,K) := | for j ∈ 1..K
                |   z_j ← round(md(10)/10)
                |   z_j ← 0  if a ≥ 0
                |   z_j ← 1  if b ≤ 0
                | k1 ← 1  if a ≥ 0
                | k1 ← −1  if a < 0 ∨ b ≤ 0
                | for j ∈ 1..K
                |   for i ∈ 1..L
                |     c0_{j,i} ← (k1)^{z_j} · round(md(10)/10)
                | g0 ← codg(c0,a,b)
                | g0
```

Входные параметры

a, b - границы интервала
L - длина хромосомы
K - количество особей

Выходной параметр

g0 - массив особей начальной популяции

```
g0 := ss0(a,b,L,K)
```

$$g0 = \begin{pmatrix} -1 & 0 & 0 & 0 & -1 \\ -1 & -1 & -1 & 0 & -1 \\ 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 \\ -1 & -1 & 0 & -1 & 0 \\ 0 & 0 & 0 & 1 & 0 \end{pmatrix}$$

Рис. 3.1 Пример программы формирования начальной популяции

Шаг 2. Произвести расчет пробных решений x^k, $x^k \in [a, b]$ путем декодирования чисел Грэя (начальное поколение) по формуле $x^k = \frac{e^{-1}(s_k^0)}{r} + a$, если $a \geq 0$, либо $x^k = \frac{e^{-1}(s_k^0)}{r} + b$, если $b \leq 0$. Здесь $r = \frac{2^L - 1}{b - a}$; a и b границы интервала. Если $a < 0$, $b > 0$, то для отрицательных двоичных чисел $r^- = \frac{2^L - 1}{|a|}$, $x^k = \frac{e^{-1}(s_k^0)}{r^-}$, а для положительных – $r^+ = \frac{2^L - 1}{b}$, $x^k = \frac{e^{-1}(s_k^0)}{r^+}$.

Оценить степень приспособленности каждой особи (рассчитать значения целевой функции $f_k(x) = f(x^k)$.

На рис. 3.2 и 3.3 приведены примеры программ кодирования и декодирования вещественных чисел. Пример программы декодирования двоичных чисел в вещественные приведен на рис. 2.2.

Перевод двоичных чисел в код Грея

```
codg(c,a,b) := | K ← rows(c)
               | L ← cols(c)
               | for j ∈ 1..K
               |     kk_j ← 1 if Σ_{i=1}^{L} c_{j,i} ≥ 0
               |     kk_j ← −1 otherwise
               | for j ∈ 1..K
               |     g_{j,1} ← c_{j,1}
               |     for k ∈ 2..L                                   if kk_j = 1
               |         g_{j,k} ← 0 if (c_{j,k−1} + c_{j,k}) = 0 ∨ (c_{j,k−1} + c_{j,k}) = 2
               |         g_{j,k} ← 1 otherwise
               |     for k ∈ 2..L                                   otherwise
               |         g_{j,k} ← 0 if (c_{j,k−1} + c_{j,k}) = 0 ∨ (c_{j,k−1} + c_{j,k}) = −2
               |         g_{j,k} ← −1 otherwise
               | g
```

Входные параметры

c - массив двоичных чисел
a, b - границы интервала

Выходной параметр

v - массив чисел в коде Грэя

Рис. 3.2 Пример программы кодирования чисел кодом Грэя

Результат работы программы декодирования чисел Грэя представлен ниже (при каждом запуске программы результаты меняются).

$$v0 = \begin{pmatrix} -1,065 \\ 1,597 \\ -1,774 \\ -3,726 \\ -3,903 \\ -4,613 \end{pmatrix}; \quad f(v0) = \begin{pmatrix} 3,266 \\ 6,099 \\ 7,296 \\ 28,763 \\ 31,470 \\ 43,558 \end{pmatrix}.$$ Среднее значение $\overline{f} = 20,075.$

Перевод целых чисел в коде Грея в десятичные числа

```
codGr_10(g,a,b) := | K ← rows(g)
                   | L ← cols(g)
                   | for j ∈ 1..K
                   |     kk_j ← 1 if Σ_{i=1}^{L} g_{j,i} ≥ 0
                   |     kk_j ← −1 otherwise
                   | for j ∈ 1..K
                   |     for i ∈ 1..L                                  if kk_j = 1
                   |         c_{j,i} ← 0 if Σ_{k=1}^{i} g_{j,k} = 0 ∨ mod(Σ_{k=1}^{i} g_{j,k}, 2) = 0
                   |         c_{j,i} ← 1 otherwise
                   |     for i ∈ 1..L                                  otherwise
                   |         c_{j,i} ← 0 if Σ_{k=1}^{i} g_{j,k} = 0 ∨ mod(Σ_{k=1}^{i} g_{j,k}, 2) = 0
                   |         c_{j,i} ← −1 otherwise
                   | v ← cod2_10(c,a,b)
                   | v
```

Входные параметры

g - массив чисел в коде Грэя
a, b - границы интервала

Выходной параметр

v - массив десятичных чисел

Рис. 3.3 Пример программы декодирования чисел Грэя в вещественные числа

Шаг 3. Выбрать K родителей из популяции при помощи метода селекции (вероятность выбора родителя должна зависеть от степени его приспособленности). Веса рассчитываются по формуле

$$w_s = \frac{|f(x_s)|}{\sum_{i=1}^{K} |f(x_s)|}$$

$$p_s(x) = w_s \cdot 100.$$

При поиске минимума функции вероятности можно вычислять по формуле

$$p_i = \frac{1/(0{,}1 + w_i)}{\sum_{j=1}^{K} 1/(0{,}1 + w_j)} \cdot 100, \quad i = 1, \ldots, K. \qquad p = \begin{pmatrix} 43 \\ 24 \\ 20 \\ 5 \\ 5 \\ 3 \end{pmatrix}.$$

Примеры программ показаны на рис. 3.4 и 3.5.

Выбор родителей с помощью рулетки

```
ruletka(w) := | n ← rows(w)
              | for j ∈ 1..n
              |   k_j ← j
              | for j ∈ 1..n
              |   r_j ← round(rnd(100))
              | for j ∈ 1..n
              |   for i ∈ 1..n
              |     if r_j ≥ Σ_{m=1}^{i} w_m
              |       | k_j ← i
              |       | continue
              | k
```

Выходной параметр

k - номера выбранных особей-родителей

Входной параметр

w - массив вероятностей

$$w = \begin{pmatrix} 12 \\ 1 \\ 45 \\ 4 \\ 31 \\ 6 \end{pmatrix}$$

kr := ruletka(w)

$$kr = \begin{pmatrix} 2 \\ 4 \\ 5 \\ 4 \\ 5 \\ 2 \end{pmatrix}$$

Рис. 3.4 Пример программы Рулетка

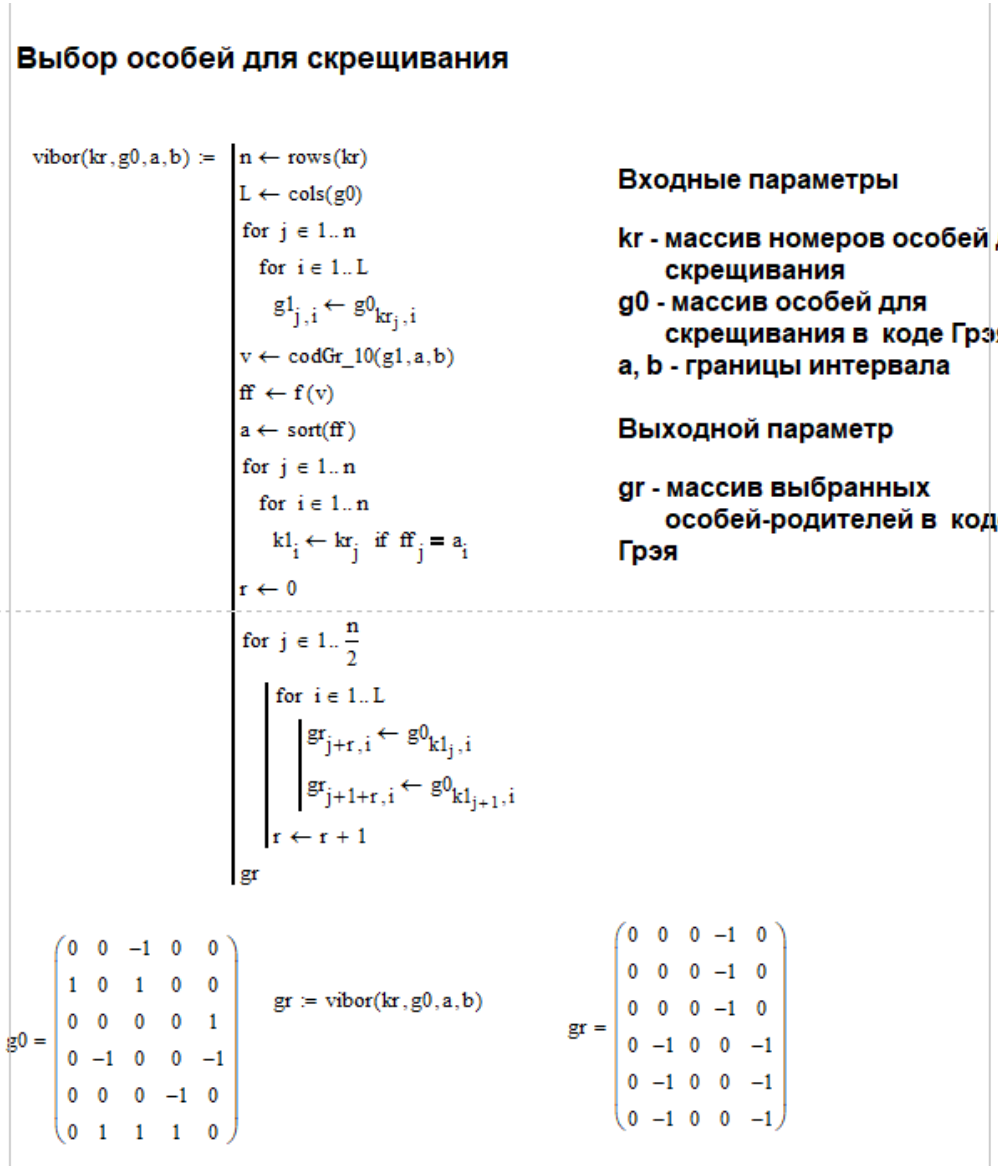

Рис. 3.5 Пример программы выбора особей для скрещивания

Шаг 4. Выбрать из родительского пула пару родителей для репродукции. При помощи оператора кроссинговера получить потомка.

Особенностью оператора скрещивания хромосом на множестве положительных и отрицательных чисел состоит в том, что если идет обмен генами хромосом, соответствующих отрицательным и положительным значениям фенотипа, то значения генов для обеих хромосом принимаются равными нулю.

Скрещивание

```
cros(gr) := | n ← rows(gr)
            | L ← cols(gr)
            | for j ∈ 1..n
            |   | k_j ← 1 if Σ_{i=1}^{L} gr_{j,i} ≥ 0
            |   | k_j ← −1 otherwise
            | for j ∈ 1..n
            |   for i ∈ 1..L
            |     gs_{j,i} ← gr_{j,i}
            | for j ∈ 1..n/2
            |   | rs_j ← round(rnd(L))
            |   | rs_j ← 1 if rs_j = 0
            | r ← 0
            | for j ∈ 1..n/2
            |   | gs_{j+r,rs_j} ← gr_{j+1+r,rs_j}
            |   | gs_{j+1+r,rs_j} ← gr_{j+r,rs_j}
            |   | if k_{j+r}·k_{j+r+1} = −1
            |   |   | gs_{j+r,rs_j} ← 0
            |   |   | gs_{j+1+r,rs_j} ← 0
            |   | r ← r + 1
            | gs

gp := cros(gr)
```

Входной параметр

cr - массив особей для скрещивания в двоичном коде

Выходной параметр

cs - массив особей потомков

Рис. 3.6 Пример программы генетического оператора скрещивания.

Результаты расчета (при каждом запуске программы результаты меняются)

$$gp = \begin{pmatrix} 0 & 0 & -1 & 0 & -1 \\ 0 & 0 & -1 & 0 & -1 \\ 0 & 0 & 0 & 0 & -1 \\ 0 & 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 & 1 \\ 0 & 1 & 1 & 0 & 1 \end{pmatrix};\ vp = \begin{pmatrix} -1,065 \\ -1,065 \\ -0,177 \\ 2,484 \\ 1,597 \\ 1,597 \end{pmatrix}.\ fp = \begin{pmatrix} 3,266 \\ 3,266 \\ 3,266 \\ 3,266 \\ 6,099 \\ 6,099 \end{pmatrix}$$

Среднее значение $\overline{f} = 4,211$.

Шаг 5. Подвергнуть потомков мутации с вероятностью $p_m = \frac{1}{L}$.

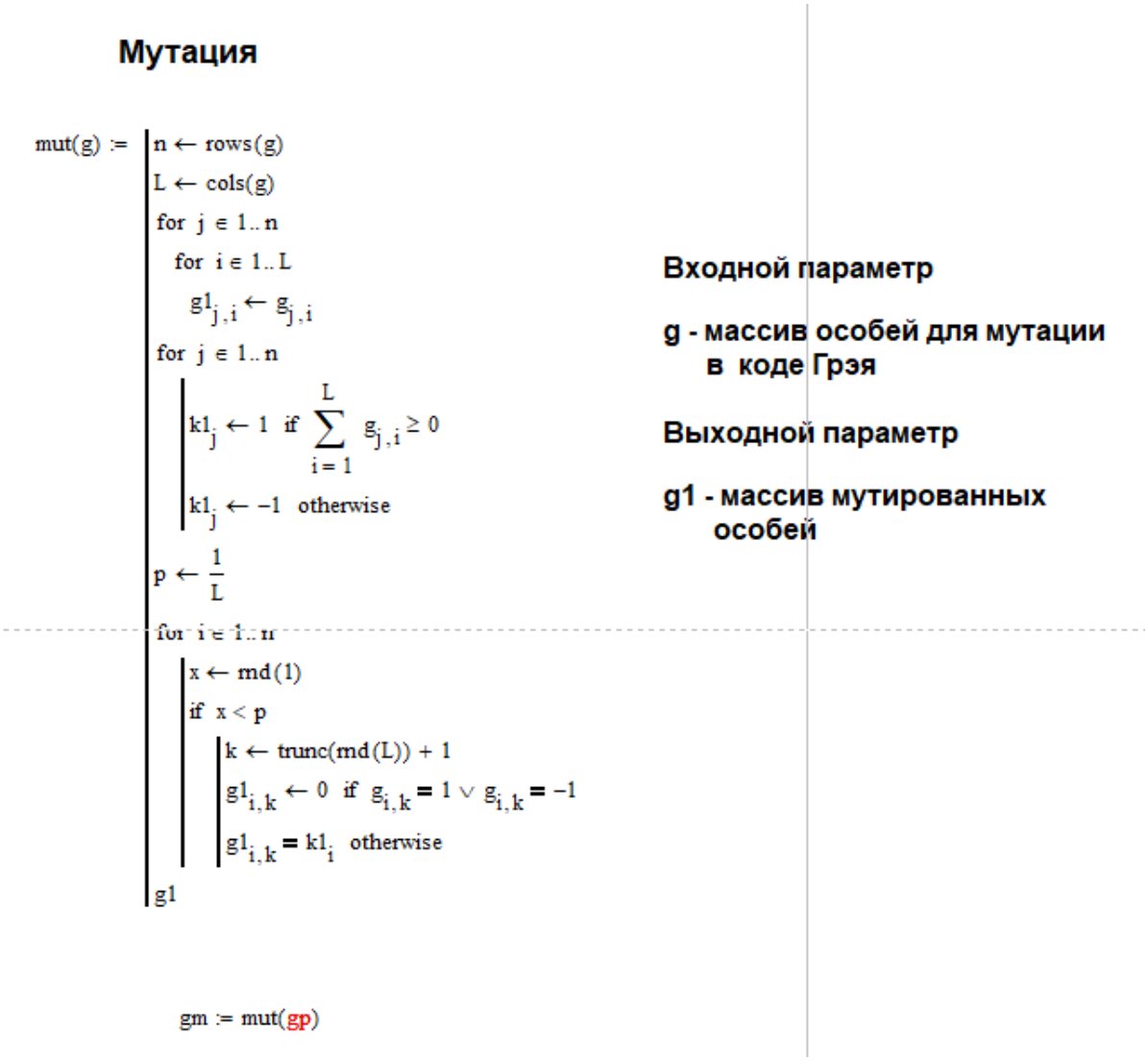

Рис. 3.7 Пример программы генетического оператора Мутация

Результаты расчета (при каждом запуске программы результаты меняются)

$$gm = \begin{pmatrix} 0 & 0 & -1 & 0 & -1 \\ 0 & 0 & -1 & 0 & -1 \\ 0 & 0 & 0 & 0 & -1 \\ 0 & 0 & 1 & 0 & 1 \\ 0 & 1 & 1 & 0 & 1 \\ 0 & 1 & 1 & 0 & 1 \end{pmatrix}; \quad vm = \begin{pmatrix} -1,065 \\ -1,065 \\ -0,177 \\ 2,484 \\ 1,597 \\ 1,597 \end{pmatrix}. \quad fm = \begin{pmatrix} 3,266 \\ 3,266 \\ 3,266 \\ 3,266 \\ 5,099 \\ 6,099 \end{pmatrix}$$

Среднее значение $\overline{f} = 4,032$.

Шаг 6. Выполнить элитарный отбор особей.
На рис. 2.7 приведен пример программы элитарного отбора особей в новую популяцию.

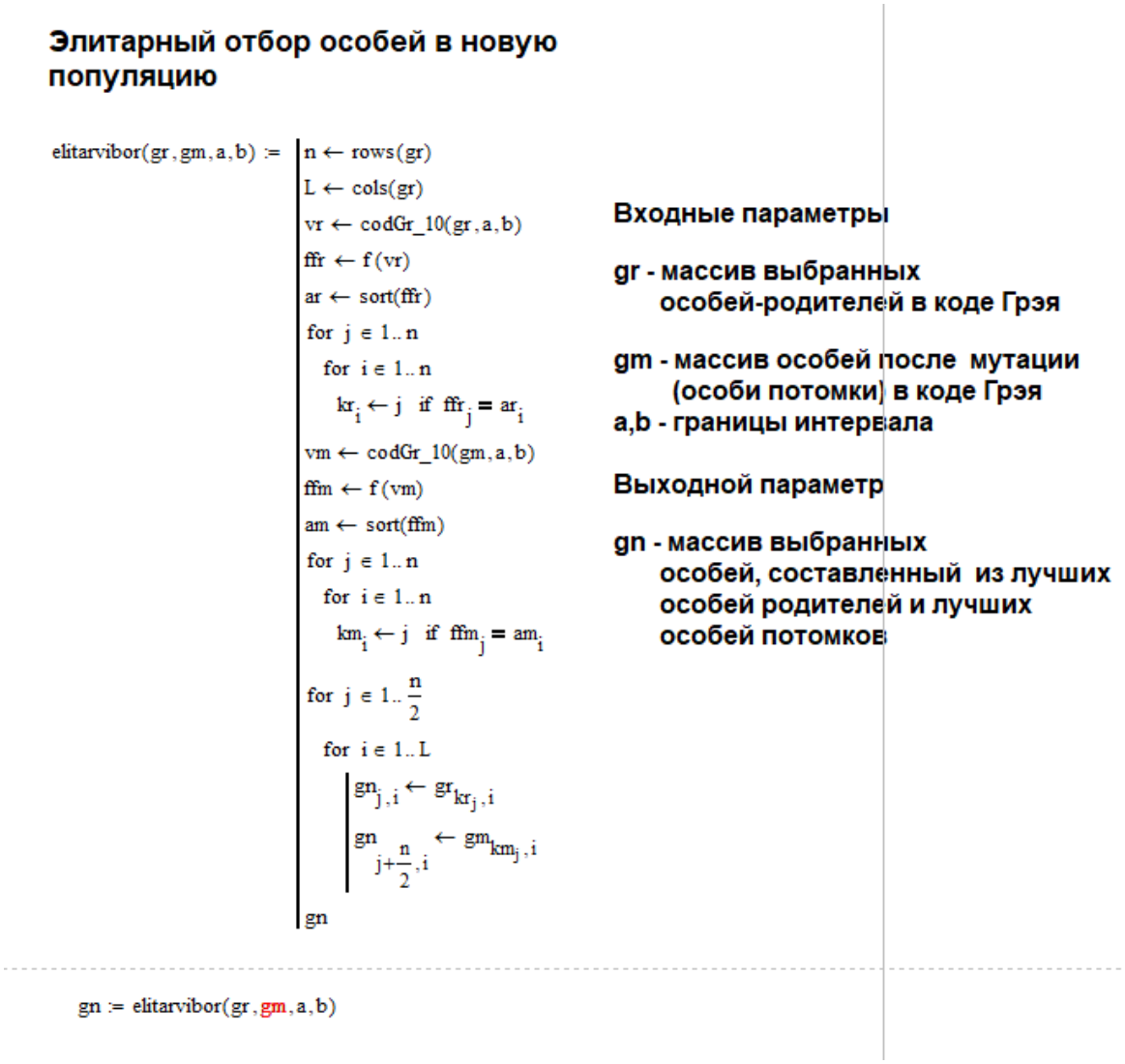

Рис. 2.7 Элитарный отбор особей

Результаты расчета (при каждом запуске программы результаты меняются)

$$gn = \begin{pmatrix} 0 & 0 & -1 & 0 & -1 \\ 0 & 0 & -1 & 0 & -1 \\ 0 & 0 & -1 & 0 & -1 \\ 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 & 1 \end{pmatrix}; \quad vn = \begin{pmatrix} -1,065 \\ -1,065 \\ -1,065 \\ 1,065 \\ 1,065 \\ 1,065 \end{pmatrix}; \quad f(vn) = \begin{pmatrix} 3,266 \\ 3,266 \\ 3,266 \\ 3,266 \\ 3,266 \\ 3,266 \end{pmatrix}.$$

Среднее значение $\overline{f} = 3,266$.

Шаг 7. Повторить шаги 5–6, пока не будет сгенерировано новое поколение популяции, содержащее K хромосом

Шаг 8. Оценить степень приспособленности каждой особи в новой популяции.

Шаг 9. Выход, если выполняется критерий останова, иначе на шаг 3.

Шаг 10. Перейти к шагу 3, если количество поколений не превышает допустимого.

3.3 Практическое задание

Написать программу поиска минимума функции одной переменной на множестве вещественных чисел с бинарным кодированием кодом Грэя. Программа может быть написана на любом алгоритмическом языке.

Входные данные:

- целевая функция;
- границы интервала поиска минимума функции $[a,b]$, где a и b – вещественные числа;
- количество особей в популяции K;
- количество поколений N (количество итераций);
- параметр окончания поиска ε (точность решения).

Варианты заданий

1	$f(x)=\sqrt{1+x^2}+e^{-2x},\ x\in[-7;7],\ \varepsilon=0,001$
2	$f(x)=x^4+2x^2+4x+1,\ x\in[-7;7],\ \varepsilon=0,001$
3	$f(x)=x^4-10\cdot x^3+36\cdot x^2+5\cdot x;\quad x\in[-7;7],\ \varepsilon=0,001$
4	$f(x)=\ln(1+x^2)-\sin x;\quad x\in[-7;7],\ \varepsilon=0,001$
5	$f(x)=\frac{1}{4}\cdot x^4+x^2-8\cdot x+12;\quad x\in[-7;7],\ \varepsilon=0,001$
6	$f(x)=x^3-x+1;\quad x\in[-7;7],\ \varepsilon=0,001$
7	$f(x)=2x^2+16/x;\quad x\in[0;7],\ \varepsilon=0,001$
8	$f(x)=4x^3+2x-3x^2+e^{x/2},\ x\in[0;7],\ \varepsilon=0,001$
9	$f(x)=24x-2\cdot x^2,\ x\in[0;15],\ \varepsilon=0,001$
10	$f(x)=x^4-12\cdot x^3+47\cdot x^2-60\cdot x;\ x\in[0;7],\ \varepsilon=0,001$
11	$f(x)=10x^3+3x^2+x+5;\ x\in[-7;7],\ \varepsilon=0,001$
12	$f(x)=x^2+e^{-x};\ x\in[-7;7],\ \varepsilon=0,001$
13	$f(x)=2x+e^{-x};\ x\in[-7;7],\ \varepsilon=0,001$
14	$f(x)=x^2+x+\sin x;\ x\in[-7;7],\ \varepsilon=0,001$
15	$f(x)=x^2-x+e^{-x};\ x\in[-7;7],\ \varepsilon=0,001$
16	$f(x)=-\left(2x^2-5-2^x\right);\ x\in[-7;7],\ \varepsilon=0,001$

17	$f(x)=\frac{1}{3}x^3+2x^2-5x+6;\ x\in[-7;7],\ \varepsilon=0,001$
18	$f(x)=-\left(-x^2+5x-6\right);\ x\in[-7;7],\ \varepsilon=0,001$

Литература

4 Holland J.H. Adaptation in Natural and Artificial Systems. – The University of Michigan Press, University of Michigan, Ann Arbor, 1975.

5 Вороновский Г.К. Генетические алгоритмы, искусственные нейронные сети и проблемы виртуальной реальности / Г. К. Вороновский, К. В. Махотило, С. Н. Петрашев, С. А. Сергеев.– Х.: ОСНОВА, 1997.– 112 с.

6 Herrera F., Lozano M., J.L. Verdegay. Tackling Real-Coded Genetic Algorithms: Operators and Tools for Behavioural Analysis // Artificial Intelligence Review, vol. 12 Issue 4, Aug. 1, 1998. – P. 265- 319.

Лабораторная работа №4

ПОИСК МИНИМУМА ФУНКЦИИ ОДНОЙ ПЕРЕМЕННОЙ С ВЕЩЕСТВЕННЫМ КОДИРОВАНИЕМ ЧИСЕЛ

Цель работы:

4. Научиться вычислять приближенное решение задачи одномерной оптимизации на множестве вещественных чисел с вещественным кодированием чисел

4.1 Алгоритм одномерной оптимизации на множестве вещественных чисел

Рассмотрим применение алгоритма на следующем примере. Найти минимум функции $f(x)=5-24x+17x^2-\frac{11}{3}x^3+\frac{1}{4}x^4$ для вещественной переменной $x\in[0;8]$.

Шаг 1. Создать начальную популяцию $x_k^0,\ k=1,...,K$ из K (особей). Для этого случайным образом генерируется K целых чисел из интервала $[0;2^L-1]$, где L – количество битов. Пример программы в пакете Mathcad приведен на рис. 4.1.

Результаты расчета для $K=6$ и $L=5$ приведены ниже

$$k0=\begin{pmatrix}0\\6\\18\\11\\26\\5\end{pmatrix};\quad v0=\begin{pmatrix}0\\1{,}548\\4{,}645\\2{,}839\\6{,}710\\1{,}290\end{pmatrix}$$

Выбор особей для начальной популяции (K особей). Номера особей определяются целыми числами длиной L битов

```
nomer(L,K) := | N ← 2^L − 1
              | for j ∈ 1..K
              |   kn_j ← j
              | for j ∈ 1..K
              |   kn_j ← round(rnd(N))
              | kn
```

Входные данные

L - количество битов
K - размер популяции (четное число)

Выходные данные

kn - номера генов в популяции

```
k0 := nomer(L,K)
```

Формирование начальной популяции (перевод целых чисел в вещественные числа

```
popul(a,b,kn,L) := | n ← rows(kn)
                   | N ← 2^L − 1
                   | for j ∈ 1..n
                   |   v0_j ← a + kn_j · (b − a)/N
                   | v0
```

Входные данные

a, b - границы интервала
kn - номера генов в популяции
L - количество битов

Выходные данные

v0 - начальная популяция

```
v0 := popul(a,b,k0,L)
```

Рис. 4.1 Пример программы формирования начальной популяции

Шаг 2.

Оценить степень приспособленности каждой особи (рассчитать значения целевой функции $f_k(x) = f(x^k)$.

$$f(v0) = \begin{pmatrix} 5 \\ -3,578 \\ 9,217 \\ 6,220 \\ 8,416 \\ -4,848 \end{pmatrix}.$$ Среднее значение $\overline{f} = 3,404$.

Шаг 3. Выбрать K родителей из популяции при помощи метода селекции (вероятность выбора родителя должна зависеть от степени его приспособленности). Веса рассчитываются по формуле

$$w_i = \frac{|f(x_i)|}{\sum_{k=1}^{K} |f(x_k)|}.$$

При поиске минимума функции вероятности можно вычислять по формулам

$$p_i = \frac{1 - w_i}{\sum_{j=1}^{K} (1 - w_j)},\ i = 1,...,K;\quad w = \begin{pmatrix} 0,173 \\ 0,181 \\ 0,151 \\ 0,167 \\ 0,155 \\ 0,174 \end{pmatrix}.$$

Примеры программ показаны на рис. 4.2 и 4.3.

Результаты расчета (при каждом запуске программы результаты меняются)

$$kr = \begin{pmatrix} 3 \\ 4 \\ 4 \\ 4 \\ 5 \\ 1 \end{pmatrix}$$

Выбор родителей с помощью рулетки

```
ruletka(w) := | n ← rows(w)
              | w1 ← sort(w)
              | for j ∈ 1..n
              |   kr_j ← j
              | for j ∈ 1..n
              |   r_j ← rnd(1)
              | for j ∈ 1..n
              |   for i ∈ 1..n
              |     if r_j ≥ Σ_{m=1}^{i} w1_m
              |       | kr_j ← i
              |       | continue
              | kr

kr := ruletka(w)
```

Входной параметр

w - массив вероятностей

Выходной параметр

k - номера выбранных особей-родителей

Рис. 4.2 Пример программы Рулетка

```
viborvs(kr, v0) := | n ← rows(v0)
                   | for j ∈ 1..n
                   |   v1_j ← v0_(kr_j)
                   | ff ← f(v1)
                   | a ← sort(f(v1))
                   | for j ∈ 1..n
                   |   for i ∈ 1..n
                   |     k1_i ← j  if ff_j = a_i
                   | r ← 0
                   | for j ∈ 1..n/2
                   |   | vr_(j+r) ← v1_(k1_j)
                   |   | vr_(j+1+r) ← v1_(k1_(j+1))
                   |   | r ← r + 1
                   | vr

vr := viborvs(kr, v0)
```

Входные параметры

kr - массив номеров особей для скрещивания
v0 - массив особей для скрещивания (начальная популяция)

Выходной параметр

vr - массив выбранных особей-родителей

Рис. 4.3 Пример программы выбора особей для скрещивания

Результаты работы программы выбора особей.

$$vr = \begin{pmatrix} 0 \\ 2{,}839 \\ 2{,}839 \\ 2{,}839 \\ 2{,}839 \\ 2{,}839 \end{pmatrix};\ f(vr) = \begin{pmatrix} 5 \\ 6{,}22 \\ 6{,}22 \\ 6{,}22 \\ 6{,}22 \\ 6{,}22 \end{pmatrix}.$$

Шаг 4. Выбрать из родительского пула пару родителей для репродукции. При помощи оператора кроссинговера получить потомка.

Скрещивание

```
cros(vr, α) := | n ← rows(vr)
               | for j ∈ 1..n/2
               |   | rs ← rnd(1)
               |   | m_j ← (1 + 2·α)·rs − α
               | r ← 0
               | for j ∈ 1..n/2
               |   | vs_{j+r} ← (1 − m_j)·min(vr_{j+r}, vr_{j+1+r}) + m_j·max(vr_{j+r}, vr_{j+1+r})
               |   | vs_{j+1+r} ← (1 − m_j)·max(vr_{j+r}, vr_{j+1+r}) + m_j·min(vr_{j+r}, vr_{j+1+r})
               |   | r ← r + 1
               | vs

vp := cros(vr, α)
```

Входной параметр

vr - массив особей для скрещивания
α - параметр (0<= α <=1)

Выходной параметр

cs - массив особей потомков

Рис. 4.4 Пример программы генетического оператора скрещивания.

Результаты расчета (при каждом запуске программы результаты меняются)

$$vp = \begin{pmatrix} 0{,}468 \\ 2{,}371 \\ 2{,}839 \\ 2{,}839 \\ 2{,}839 \\ 2{,}839 \end{pmatrix};\ f(vr) = \begin{pmatrix} -2{,}869 \\ 2{,}692 \\ 6{,}22 \\ 6{,}22 \\ 6{,}22 \\ 6{,}22 \end{pmatrix}.$$ Среднее значение $\overline{f} = 4{,}117$.

Шаг 5. Подвергнуть потомков мутации с вероятностью $p_m = \frac{1}{L}$.

Мутация

Мутации подвергаем последние наихудшие хромосомы

```
mutv(vs, a, b, β) := | n ← rows(vs)
                     | ff ← f(vs)
                     | aa ← sort(f(vs))
                     | for j ∈ 1..n
                     |   for i ∈ 1..n
                     |     k1_i ← j if ff_j = aa_i
                     | for j ∈ 1..n
                     |   vm_j ← vs_(k1_j)
                     | m ← n/2
                     | for i ∈ 1..m
                     |   r ← rnd(1)
                     |   g ← (vm_(i+m) − a)/(b − a)
                     |   g1 ← g
                     |   g1 ← g − g·((g − r)/g)^β if g ≥ r
                     |   g1 ← g + (1 − g)·((r − g)/(1 − g))^β otherwise
                     |   vm_(i+m) ← (1 − g1)·a + g1·b
                     | vm
```

Входной параметр

vs- массив особей для мутации

a, b - границы интервала
β - парметр (0 <= β <= 1)

Выходной параметр

vm- массив мутированных особей

Рис. 4.5 Пример программы генетического оператора Мутация

Результаты расчета (при каждом запуске программы результаты меняются)

$$vm = \begin{pmatrix} 0,468 \\ 2,371 \\ 2,839 \\ 3,892 \\ 0,533 \\ 0,191 \end{pmatrix} \cdot \quad f(vm) = \begin{pmatrix} -2,869 \\ 2,692 \\ 6,22 \\ 10,298 \\ -3,501 \\ 1,002 \end{pmatrix}. \text{ Среднее значение } \overline{f} = 2,307 .$$

Шаг 6. Выполнить элитарный отбор особей.
На рис. 4.6 приведен пример программы элитарного отбора особей в новую популяцию.

Элитарный отбор особей в новую популяцию

```
elitarvibor(vr, vm, a, b) := | n ← rows(vr)
                             | L ← cols(vr)
                             | ffr ← f(vr)
                             | ar ← sort(ffr)
                             | for j ∈ 1..n
                             |   for i ∈ 1..n
                             |     kr_i ← j if ffr_j = ar_i
                             | ffm ← f(vm)
                             | am ← sort(ffm)
                             | for j ∈ 1..n
                             |   for i ∈ 1..n
                             |     km_i ← j if ffm_j = am_i
                             | for j ∈ 1..n/2
                             |   for i ∈ 1..L
                             |     vn_{j,i} ← vr_{kr_j,i}
                             |     vn_{j+n/2,i} ← vm_{km_j,i}
                             | vn

vn := elitarvibor(vr, vm, a, b)
```

Входные параметры

vr - массив выбранных особей-родителей

vm - массив особей после мутации (особи потомки)

a,b - границы интервала

Выходной параметр

vn - массив выбранных особей, составленный из лучших особей родителей и лучших особей потомков

Рис. 4.6 Элитарный отбор особей

Результаты расчета (при каждом запуске программы результаты меняются)

$$vr = \begin{pmatrix} 0 \\ 2,839 \\ 2,839 \\ 2,839 \\ 2,839 \\ 2,839 \end{pmatrix};\ f(vr) = \begin{pmatrix} 5 \\ 6,22 \\ 6,22 \\ 6,22 \\ 6,22 \\ 6,22 \end{pmatrix}.\quad vm = \begin{pmatrix} 0,468 \\ 2,371 \\ 2,839 \\ 3,892 \\ 0,533 \\ 0,191 \end{pmatrix}.\ f(vm) = \begin{pmatrix} -2,869 \\ 2,692 \\ 6,22 \\ 10,298 \\ -3,501 \\ 1,002 \end{pmatrix}.$$

$$vn = \begin{pmatrix} 0 \\ 2,839 \\ 2,839 \\ 0,533 \\ 0,468 \\ 0,191 \end{pmatrix};\quad f(vn) = \begin{pmatrix} 5 \\ 6,22 \\ 6,22 \\ -3,501 \\ -0,269 \\ 1,002 \end{pmatrix}.$$ Среднее значение $\overline{f} = 2,307$.

Шаг 7. Повторить шаги 5–6, пока не будет сгенерировано новое поколение популяции, содержащее K хромосом

Шаг 8. Оценить степень приспособленности каждой особи в новой популяции.

Шаг 9. Выход, если выполняется критерий останова, иначе на шаг 3.

Шаг 10. Перейти к шагу 3, если количество поколений не превышает допустимого.

4.2 Практическое задание

Написать программу поиска минимума функции одной переменной на множестве вещественных чисел с вещественным кодированием. Программа может быть написана на любом алгоритмическом языке.

Входные данные:

- целевая функция;
- границы интервала поиска минимума функции $[a,b]$, где a и b – вещественные числа;
- количество особей в популяции K;
- количество поколений N (количество итераций);
- параметры алгоритма $\varepsilon = 1,\ \beta = 0,5$.
- параметр окончания поиска ε (точность решения).

Варианты заданий

1	$f(x) = \sqrt{1+x^2} + e^{-2x},\ x \in [-7;7],\ \varepsilon = 0,001$
2	$f(x) = x^4 + 2x^2 + 4x + 1,\ x \in [-7;7],\ \varepsilon = 0,001$
3	$f(x) = x^4 - 10 \cdot x^3 + 36 \cdot x^2 + 5 \cdot x;\quad x \in [-7;7],\ \varepsilon = 0,001$
4	$f(x) = \ln(1+x^2) - \sin x;\quad x \in [-7;7],\ \varepsilon = 0,001$
5	$f(x) = \frac{1}{4} \cdot x^4 + x^2 - 8 \cdot x + 12;\quad x \in [-7;7],\ \varepsilon = 0,001$
6	$f(x) = x^3 - x + 1;\quad x \in [-7;7],\ \varepsilon = 0,001$
7	$f(x) = 2x^2 + 16/x;\quad x \in [0;7],\ \varepsilon = 0,001$
8	$f(x) = 4x^3 + 2x - 3x^2 + e^{x/2},\ x \in [0;7],\ \varepsilon = 0,001$
9	$f(x) = 24x - 2 \cdot x^2,\ x \in [0;15],\ \varepsilon = 0,001$
10	$f(x) = x^4 - 12 \cdot x^3 + 47 \cdot x^2 - 60 \cdot x;\ x \in [0;7],\ \varepsilon = 0,001$
11	$f(x) = 10x^3 + 3x^2 + x + 5;\ x \in [-7;7],\ \varepsilon = 0,001$
12	$f(x) = x^2 + e^{-x};\ x \in [-7;7],\ \varepsilon = 0,001$
13	$f(x) = 2x + e^{-x};\ x \in [-7;7],\ \varepsilon = 0,001$
14	$f(x) = x^2 + x + \sin x;\ x \in [-7;7],\ \varepsilon = 0,001$
15	$f(x) = x^2 - x + e^{-x};\ x \in [-7;7],\ \varepsilon = 0,001$
16	$f(x) = -\left(2x^2 - 5 - 2^x\right);\ x \in [-7;7],\ \varepsilon = 0,001$

17	$f(x)=\frac{1}{3}x^3+2x^2-5x+6;\ x\in[-7;7],\ \varepsilon=0,001$
18	$f(x)=-\left(-x^2+5x-6\right);\ x\in[-7;7],\ \varepsilon=0,001$

Литература

7 Holland J.H. Adaptation in Natural and Artificial Systems. – The University of Michigan Press, University of Michigan, Ann Arbor, 1975.

8 Вороновский Г.К. Генетические алгоритмы, искусственные нейронные сети и проблемы виртуальной реальности / Г. К. Вороновский, К. В. Махотило, С. Н. Петрашев, С. А. Сергеев.– Х.: ОСНОВА, 1997.– 112 с.

9 Herrera F., Lozano M., J.L. Verdegay. Tackling Real-Coded Genetic Algorithms: Operators and Tools for Behavioural Analysis // Artificial Intelligence Review, vol. 12 Issue 4, Aug. 1, 1998. – P. 265- 319.

Лабораторная работа №5

ПОИСК МИНИМУМА ФУНКЦИИ МНОГИХ ПЕРЕМЕННЫХ С ВЕЩЕСТВЕННЫМ КОДИРОВАНИЕМ ЧИСЕЛ

Цель работы:

5. Научиться вычислять приближенное решение задачи многомерной оптимизации на множестве вещественных чисел с вещественным кодированием чисел

5.1 Алгоритм одномерной оптимизации на множестве вещественных чисел

Рассмотрим применение алгоритма на следующем примере. Найти минимум функции $f(x) = 2x_1^2 + x_2^2 + \sin(x_1 + x_2)$ для вещественной переменной $x \in \left[\begin{pmatrix} -5 \\ -5 \end{pmatrix}; \begin{pmatrix} 5 \\ 5 \end{pmatrix}\right]$.

Шаг 1. Создать начальную популяцию $x_k^0,\ k = 1,...,K$ из K (особей). Для этого случайным образом генерируется K целых чисел из интервала $[0;\ 2^L - 1]$, где L – количество битов. Пример программы в пакете Mathcad приведен на рис. 5.1.

Результаты расчета для K = 6 и L = 5 приведены ниже

$$k0 = \begin{pmatrix} 1 & 7 & 19 & 12 & 27 & 6 \\ 23 & 10 & 4 & 6 & 32 & 5 \end{pmatrix};$$

$$v0 = \begin{pmatrix} -4,677 & -2,742 & 1,129 & -1,129 & 3,71 & -3,065 \\ 2,419 & -1,774 & -3,710 & -3,065 & 5,323 & -3,387 \end{pmatrix}.$$

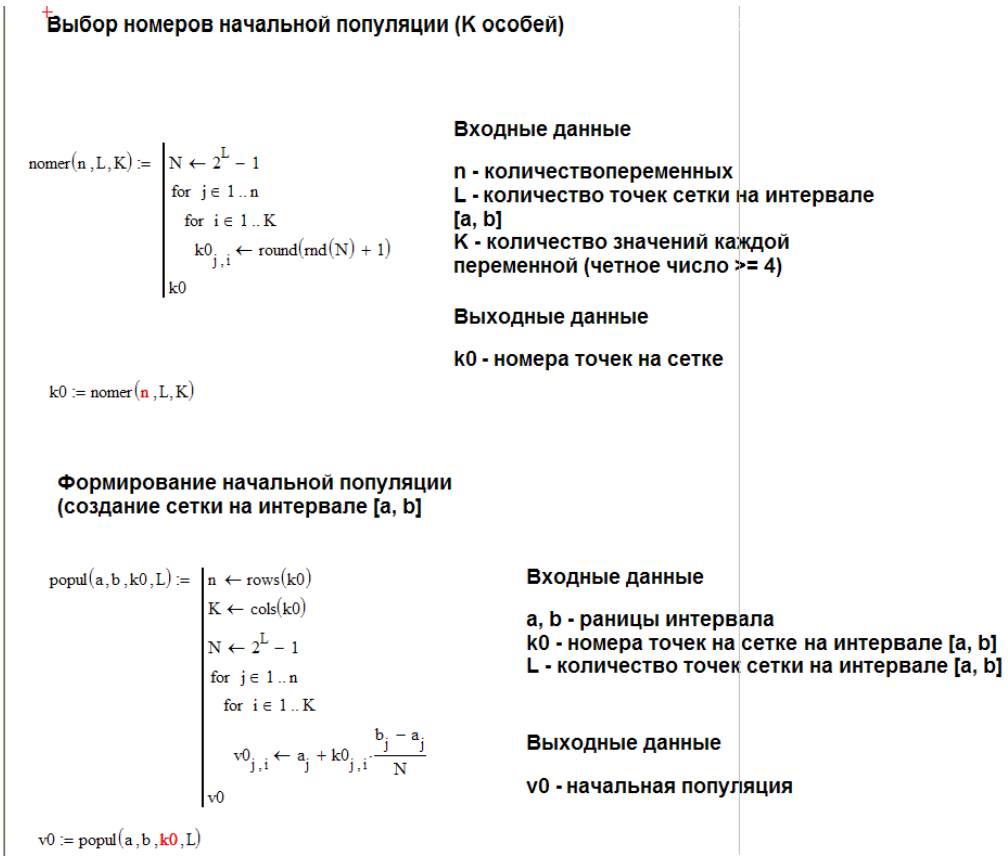

Рис. 5.1 Пример программы формирования начальной популяции

Шаг 2.

Оценить степень приспособленности каждой особи (рассчитать значения целевой функции $f_k(x) = f(x^k)$.

$$f(v0) = \begin{pmatrix} 48,837 \\ 19,165 \\ 15,779 \\ 12,809 \\ 56,236 \\ 30,087 \end{pmatrix}.$$ Среднее значение $\overline{f} = 30,486.$

Шаг 3. Выбрать K родителей из популяции при помощи метода селекции (вероятность выбора родителя должна зависеть от степени его приспособленности). Веса рассчитываются по формуле

$$w_i = \frac{\left| f(x^i) \right|}{\sum_{k=1}^{K} \left| f(x^k) \right|}.$$

При поиске минимума функции вероятности можно вычислять по формулам

$$p_i = \frac{1 - w_i}{\sum_{j=1}^{K} (1 - w_j)}, \; i = 1, ..., K. \quad w = \begin{pmatrix} 0,147 \\ 0,179 \\ 0,183 \\ 0,186 \\ 0,139 \\ 0,167 \end{pmatrix}. \qquad kr = \begin{pmatrix} 1 \\ 1 \\ 3 \\ 2 \\ 5 \\ 1 \end{pmatrix}$$

Примеры программ показаны на рис. 5.2 и 5.3.

Результаты расчета (при каждом запуске программы результаты меняются)

Выбор родителей с помощью рулетки

```
ruletka(w) := | n ← rows(w)
              | w1 ← sort(w)
              | for j ∈ 1..n
              |   kr_j ← j
              | for j ∈ 1..n
              |   r_j ← rnd(1)
              | for j ∈ 1..n
              |   for i ∈ 1..n
              |     if r_j ≥ Σ_{m=1}^{i} w1_m
              |       | kr_j ← i
              |       | continue
              | kr

kr := ruletka(w)
```

Входной параметр

w - массив вероятностей

Выходной параметр

k - номера выбранных особей-родителей

Рис. 5.2 Пример программы Рулетка

Результаты работы программы выбора особей.

Выбор особей для скрещивания

```
viborvs(kr,v0) := | n ← rows(v0)
                  | K ← cols(v0)
                  | for j ∈ 1..K
                  |   | v1^⟨j⟩ ← v0^⟨kr_j⟩
                  |   | ff_j ← f(v1^⟨j⟩)
                  | a ← sort(ff)
                  | for j ∈ 1..K
                  |   for i ∈ 1..K
                  |     k1_i ← j  if ff_j = a_i
                  | r ← 0
                  | for j ∈ 1..K/2
                  |   | vr^⟨j+r⟩ ← v1^⟨k1_j⟩
                  |   | vr^⟨j+1+r⟩ ← v1^⟨k1_{j+1}⟩
                  |   | r ← r + 1
                  | vr

vr := viborvs(kr,v0)
```

Входные параметры

kr - массив номеров особей для скрещивания
v0 - массив особей для скрещивания (начальная популяция)

Выходной параметр

vr - массив выбранных особей-родителей

Рис. 5.3 Пример программы выбора особей для скрещивания

$$vr = \begin{pmatrix} 1{,}129 & -2{,}742 & -2{,}742 & -4{,}677 & -4{,}677 & -4{,}677 \\ -3{,}71 & -1{,}174 & -1{,}174 & 2{,}419 & 2{,}419 & 2{,}419 \end{pmatrix};$$

$$f(vr) = \begin{pmatrix} 15{,}779 \\ 19{,}165 \\ 19{,}165 \\ 48{,}837 \\ 48{,}837 \\ 48{,}837 \end{pmatrix}.$$

Шаг 4. Выбрать из родительского пула пару родителей для репродукции. При помощи оператора кроссинговера получить потомка.

Скрещивание

```
cros(vr, α) := | n ← rows(vr)
               | for j ∈ 1..n/2
               |     rs ← rnd(1)
               |     m_j ← (1 + 2·α)·rs − α
               | r ← 0
               | for j ∈ 1..n/2
               |     vs_{j+r} ← (1 − m_j)·min(vr_{j+r}, vr_{j+1+r}) + m_j·max(vr_{j+r}, vr_{j+1+r})
               |     vs_{j+1+r} ← (1 − m_j)·max(vr_{j+r}, vr_{j+1+r}) + m_j·min(vr_{j+r}, vr_{j+1+r})
               |     r ← r + 1
               | vs

vp := cros(vr, α)
```

Входной параметр

vr - массив особей для скрещивания
α - параметр (0<= α <=1)

Выходной параметр

cs - массив особей потомков

Рис. 5.4 Пример программы генетического оператора скрещивания.

Результаты расчета (при каждом запуске программы результаты меняются)

$$vp = \begin{pmatrix} -8{,}53 & -3{,}706 & -3{,}706 & -11{,}747 & -11{,}747 & -11{,}747 \\ 5{,}949 & -0{,}810 & -0{,}810 & 9{,}489 & 9{,}489 & 9{,}489 \end{pmatrix};$$

$$f(vp) = \begin{pmatrix} 180{,}383 \\ 29{,}106 \\ 29{,}106 \\ 365{,}263 \\ 365{,}263 \\ 365{,}263 \end{pmatrix}.$$ Среднее значение $\overline{f} = 222{,}398.$

Шаг 5. Подвергнуть потомков мутации с вероятностью $p_m = \frac{1}{L}$.

Мутация

Мутации подвергаем последние наихудшие хромосомы

```
mutvm(vs, a, b, β) := | n ← rows(vs)
                      | K ← cols(vs)
                      | for j ∈ 1..K
                      |   fm_j ← f(vs^⟨j⟩)
                      | aa ← sort(fm)
                      | for j ∈ 1..K
                      |   for i ∈ 1..K
                      |     k1_i ← j  if fm_j = aa_i
                      | for i ∈ 1..K
                      |   for j ∈ 1..n
                      |     vm_{j,i} ← vs_{j,k1_i}
                      | n1 ← K/2
                      | for i ∈ 1..n
                      |   for j ∈ 1..n1
                      |     r ← rnd(1)
                      |     g ← (vm_{i,j+n1} − a_i)/(b_i − a_i)
                      |     g1 ← g  if r = g
                      |     g1 ← g − g·((g − r)/g)^β  if g > r
                      |     g1 ← g + (1 − g)·((r − g)/(1 − g))^β  if g < r
                      |     vm_{i,j+n1} ← (1 − g1)·a_i + g1·b_i
                      | vm

vm := mutvm(vs, a, b, β)
```

Входной параметр

vs- массив особей для мутации

a, b - границы интервала
β - парметр (0 <= β <= 1)

Выходной параметр

vm- массив мутированных особе

Рис. 5.5 Пример программы генетического оператора Мутация

Результаты расчета (при каждом запуске программы результаты меняются)

$$vm = \begin{pmatrix} -3,706 & -3,706 & -8,53 & 3,844 & 1,055 & -0,420 \\ -0,810 & -0,810 & 5,949 & -4,244 & 1,322 & -4,392 \end{pmatrix}. \quad f(vm) = \begin{pmatrix} 29,106 \\ 29,106 \\ 180,383 \\ 41,595 \\ 4,667 \\ 20,636,144 \end{pmatrix}.$$

Среднее значение $\overline{f} = 50,915$.

Шаг 6. Выполнить элитарный отбор особей.

На рис. 5.6 приведен пример программы элитарного отбора особей в новую популяцию.

Элитарный отбор особей в новую популяцию

```
elitarvibor(vr, vm, a, b) :=
    n ← rows(vr)
    K ← cols(vr)
    for j ∈ 1..K
        ffr_j ← f(vr^⟨j⟩)
    ar ← sort(ffr)
    for j ∈ 1..K
        for i ∈ 1..K
            kr_i ← j  if  ffr_j = ar_i
    for j ∈ 1..K
        ffm_j ← f(vm^⟨j⟩)
    am ← sort(ffm)
    for j ∈ 1..K
        for i ∈ 1..K
            km_i ← j  if  ffm_j = am_i
    for j ∈ 1..K/2
        for i ∈ 1..n
            vn_{i,j} ← vr_{i,kr_j}
            vn_{i,j+K/2} ← vm_{i,km_j}
    vn

vn := elitarvibor(vr, vm, a, b)
```

Входные параметры

vr - массив выбранных особей-родителей

vm - массив особей после мутации (особи потомки)

a,b - границы интервала

Выходной параметр

vn - массив выбранных особей, составленный из лучших особей родителей и лучших особей потомков

Рис. 5.6 Элитарный отбор особей

Результаты расчета (при каждом запуске программы результаты меняются)

$$vr = \begin{pmatrix} 1{,}129 & -2{,}742 & -2{,}742 & -4{,}677 & -4{,}677 & -4{,}677 \\ -3{,}71 & -1{,}174 & -1{,}174 & 2{,}419 & 2{,}419 & 2{,}419 \end{pmatrix};$$

$$f(vr) = \begin{pmatrix} 15{,}779 \\ 19{,}165 \\ 19{,}165 \\ 48{,}837 \\ 48{,}837 \\ 48{,}837 \end{pmatrix}.$$

$$vm = \begin{pmatrix} -3{,}706 & -3{,}706 & -8{,}53 & 3{,}844 & 1{,}055 & -0{,}420 \\ -0{,}810 & -0{,}810 & 5{,}949 & -4{,}244 & 1{,}322 & -4{,}392 \end{pmatrix}.$$

$$f(vm) = \begin{pmatrix} 29{,}106 \\ 29{,}106 \\ 180{,}383 \\ 41{,}595 \\ 4{,}667 \\ 20{,}636{,}144 \end{pmatrix}.$$

$$vn = \begin{pmatrix} 1{,}129 & -2{,}742 & -2{,}742 & 1{,}055 & -0{,}420 & -3{,}706 \\ -3{,}710 & -1{,}174 & -1{,}174 & 1{,}332 & -4{,}392 & -0{,}810 \end{pmatrix};$$

$$f(vn) = \begin{pmatrix} 15{,}779 \\ 19{,}165 \\ 19{,}165 \\ 4{,}667 \\ 20{,}636 \\ 29{,}106 \end{pmatrix}.$$ Среднее значение $\overline{f} = 18{,}086$.

Шаг 7. Повторить шаги 5–6, пока не будет сгенерировано новое поколение популяции, содержащее K хромосом

Шаг 8. Оценить степень приспособленности каждой особи в новой популяции.

Шаг 9. Выход, если выполняется критерий останова, иначе на шаг 3.

Шаг 10. Перейти к шагу 3, если количество поколений не превышает допустимого.

5.2 Практическое задание

Написать программу поиска минимума функции многих переменных на множестве вещественных чисел с вещественным кодированием. Программа может быть написана на любом алгоритмическом языке.

Входные данные:

- целевая функция;
- границы интервала поиска минимума функции $[a_i, b_i],\ i = 1,...,n$, где a_i и b_i – вещественные числа;
- количество особей в популяции K;
- количество поколений N (количество итераций);
- параметры алгоритма $\varepsilon = 1,\ \beta = 0{,}5$.
- параметр окончания поиска ε (точность решения).

Варианты заданий

1	$f(x) = 2x_1^2 + x_2^2 + \sin(x_1 + x_2)$; $x_1 \in [-5;5],\ x_2 \in [-5;5],\ \varepsilon = 0{,}01;\ xT = (-0{,}204; -0{,}409)$
2	$f(x) = c_1\left(x_2 - x_1^2\right)^2 + \left(x_2 - 1\right)^2$

	$x_1 \in [-2;2],\ x_2 \in [-2;2],\ \varepsilon = 0{,}01;\ xT = (-0{,}229;0{,}037)$
3	$f(x) = c_1 x_1^2 + c_2 x_1 x_2 + c_3 x_2^2 + c_4 x_1 + c_5 x_2 - c_6$; $c_1 = 1,\ c_2 = 1,\ c_3 = 1,\ c_4 = 1,\ c_5 = 1,\ c_6 = 1;$ $x_1 \in [-2;2],\ x_2 \in [-2;2],\ \varepsilon = 0{,}01;\ xT = (-0{,}333;-0{,}333)$
4	$f(x) = 3x_1^2 + 4x_2^2 + 23\cos(x_1 - 05)$; $x_1 \in [-5;5],\ x_2 \in [-5;5],\ \varepsilon = 0{,}01;\ xT = (-2{,}071;0)$
5	$f(x) = 2x_1^2 + x_1 x_2 + x_2^2 - 3x_1$; $x_1 \in [-2;2],\ x_2 \in [-2;2],\ \varepsilon = 0{,}01.\ xT = (0{,}857;-0{,}429)$
6	$f(x_1, x_2) = 4(x_1 - 5)^2 + (x_2 - 6)^2,\ \varepsilon = 0{,}01$ $x_1 \in [0;0],\ x_2 \in [10;10],\ \varepsilon = 0{,}01;\ xT = (5;10)$
7	$f(x) = x_1^2 + x_1 x_2 + x_2^2 - 3x_1 - 6x_2\ ,\ \varepsilon = 0{,}01;\ xT = (0;3)$ $x_1 = [-2;2];\ x_2 = [0;\ 4]$
8	$f(x) = -4 + \left(x_1^2 + x_2^2\right)^{2/3},\ \varepsilon = 0{,}01;\ xT = (0;0)$ $x_1 = [-3;3];\ x_2 = [-3;3].$
9	$f(x) = -\left(x_1^2 + x_2^2\right) \cdot \left(\exp\left(-(x_1^2 + x_2^2)\right) - 1\right),\ \varepsilon = 0{,}01$ $x_1 = [-3;3];\ x_2 = [-3;3];\ xT = (0;0)$
10	$f(x) = 100 \cdot (x_2 - x_1^2)^2 + (1 - x_1)^2,\ \varepsilon = 0{,}01;\ xT = (1;1)$ $x_1 = [-1;1];\ x_2 = [-1;1]$
11	$f(x) = (x_1 - 3)^2 - (x_2 - x_1) + \exp\ [20 \cdot (x_2 - x_1)];\ \varepsilon = 0{,}01$ $x_1 = [0;6];\ x_2 = [0;6];\ xT = (3;2{,}85)$
12	$f(x) = 4x_1 + 2x_2 - x_1^2 - x_2^2 + 5,\ \varepsilon = 0{,}01$ $x_1 = [-2;2];\ x_2 = [-3;3];\ xT = (0;-1)$
13	$f(x) = 2 \cdot x_1^2 + 4x_1 x_2 + 3x_2^2,\ \varepsilon = 0{,}001;\ xT = (0;0)$ $x_1 = [-3;3];\ x_2 = [-3;3].$
14	$f(x) = x_1^2 + 2x_2^2 + x_1 x_2 - 7x_1 - 7x_2,\ \varepsilon = 0{,}01$ $x_1 = [-3;6];\ x_2 = [-3;6];\ xT = (3;1)$
15	$f(x) = 2x_1^2 + x_2^2 + x_1 x_2 + x_1 + x_2,\ \varepsilon = 0{,}01$ $x_1 = [-3;3];\ x_2 = [-3;3];\ xT = (-0{,}143;-0{,}429)$
16	$f(x) = \left(2x^2 - 5 - 2^y + y^2\right),\ \varepsilon = 0{,}01$ $x_1 = [-3;3];\ x_2 = [-3;3];\ xT = (0;0{,}485)$

Литература

10 Holland J.H. Adaptation in Natural and Artificial Systems. – The University of Michigan Press, University of Michigan, Ann Arbor, 1975.

11 Вороновский Г.К. Генетические алгоритмы, искусственные нейронные сети и проблемы виртуальной реальности / Г. К. Вороновский, К. В. Махотило, С. Н. Петрашев, С. А. Сергеев.– Х.: ОСНОВА, 1997.– 112 с.

12 Herrera F., Lozano M., J.L. Verdegay. Tackling Real-Coded Genetic Algorithms: Operators and Tools for Behavioural Analysis // Artificial Intelligence Review, vol. 12 Issue 4, Aug. 1, 1998. – P. 265- 319.

ПРИЛОЖЕНИЕ

Пример отчёта по лабораторной работе

Министерство науки и высшего образования Российской Федерации

Федеральное государственное бюджетное образовательное учреждение высшего образования

ТОМСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ СИСТЕМ УПРАВЛЕНИЯ И РАДИОЭЛЕКТРОНИКИ (ТУСУР)

Кафедра автоматизированных систем управления (АСУ)

Отчет по лабораторной работе №1

По дисциплине

«Эвристические методы оптимизации»

Студент гр. 433-М1

_____ В.С. Шелихов

_____ Д.Д. Пимонов

«___»_____ 2023 г.

Профессор каф. АСУ, д.т.н

_____ А.А. Мицель

«___»_____ 2023 г.

Томск 2023

Цель работы

Научиться кодировать и декодировать целые и вещественные числа.

Задание на лабораторную работу

1) Написать программы в пакете Mathcad (исходные данные – количество битов L):
   a) бинарного кодирования целых чисел с помощью двоично-десятичного кода; декодирования двоичных чисел в целые десятичные числа;
   b) бинарного кодирования целых чисел с помощью кода Грея;
   c) декодирования двоичных чисел в целые десятичные числа;
   d) декодирования чисел Грея в целые десятичные числа.
2) Написать программы кодирования вещественных чисел в пакете Mathcad (исходные данные – границы интервала вещественной переменной [a, b], количество битов L):
   a) бинарного кодирования вещественных чисел с помощью двоично-десятичного кода;
   b) бинарного кодирования вещественных чисел с помощью кода Грея;
   c) декодирования двоичных чисел в вещественные десятичные числа;
   d) декодирования чисел Грея в вещественные десятичные числа.
3) Написать программу вещественного кодирования многомерных переменных в пакете Mathcad (исходные данные – границы интервала вещественных переменных [a_i, b_i], i=1,..,n)

Краткая теория

Ген – атомарный элемент хромосомы (может быть битом, числом или неким другим объектом.
Хромосома (цепочка) – упорядоченная последовательность генов (строка из каких-либо чисел). Если эта строка представлена бинарной строкой из нулей и единиц, например, 101010, то она получена либо с использованием двоичного кодирования, либо кода Грея. Каждая позиция хромосомы называется геном.

Код Гре́я — двоичный код в котором две «соседние» кодовые комбинации различаются только цифрой в одном двоичном разряде.

Бинарное кодирование - представление информации с помощью двоичного алфавита.

Вещественное кодирование – кодирование, при котором хромосомы представляются в виде вектора вещественных чисел. Связь целого числа в двоичном коде с вещественной координатой производится по формуле a+i(b-a)/($2^L - 1$).

Ход работы

На рисунке 4.1 представлен пример программы, которая преобразует десятичное целое число в двоичное целое число.

1а) Бинарное кодирование целых чисел с помощью двоично-десятичного кода;

ORIGIN := 1

Cod10_2(v) :=
L ← trunc(log(v + 1,2)) + 1
k ← 1
while k ≤ L
  $c_k \leftarrow \text{trunc}\left(\text{mod}\left(\frac{v}{2^{L-k}}, 2\right)\right)$
  k ← k + 1
c^T

На входе:
v - целое десятичное число

На выходе:
Двоичное число

c1 := Cod10_2(25) = (1 1 0 0 1)

cc1 := Cod10_2(363) = (1 0 1 1 0 1 0 1 1)

Рисунок 4.1 - Результат работы программы

На рисунке 4.2 представлен пример программы, которая преобразует бинарное целое число в код грея.

1б) Программа бинарного кодирования целых чисел с помощью кода Грея;

На входе:
c- число в двоичном коде

На выходе:
Двоичное число в коде Грея

$$\text{Codg}(c) := \left| \begin{array}{l} L \leftarrow \text{cols}(c) \\ g_{1,1} \leftarrow c_{1,1} \\ \text{for } k \in 2..L \\ \quad \left| \begin{array}{l} g_{1,k} \leftarrow 0 \text{ if } \left(c_{1,k-1} + c_{1,k}\right) = 0 \vee \left(c_{1,k-1} + c_{1,k}\right) = 2 \\ g_{1,k} \leftarrow 1 \text{ otherwise} \end{array} \right. \\ g \end{array} \right.$$

$$cc2 := (0 \;\; 1 \;\; 0 \;\; 0)$$

$$gg2 := \text{Codg}(cc2) = (0 \;\; 1 \;\; 1 \;\; 0)$$

Рисунок 4.2 - Результат работы программы

На рисунке 4.3 представлен пример программы, которая преобразует десятичное целое число в двоичный код грея.

Программа бинарного кодирования целых десятичных чисел с помощью кода Грея;

$$\text{Code10\_Gr}(v) := \left| \begin{array}{l} v \leftarrow \text{Cod10\_2}(v) \\ v \leftarrow \text{Codg}(v) \\ v \end{array} \right.$$

$$\text{Code10\_Gr}(5) = (1 \;\; 1 \;\; 1)$$

На входе:
v- число в десятичной

На выходе:
Двоичное число в коде Грея

Рисунок 4.3 - Результат работы программы

На рисунке 4.4 представлен пример программы, которая преобразует матрицу, строки которой являются двоичными целыми числами, в вектор десятичных целых чисел.

1c) Программа перевода двоичных чисел в целые десятичные.

На входе:
c - Матрица чисел в двоичной системе счисления. Строки матрицы - двоичные числа.

На выходе:
Матрица целых чисел

$$\text{cod2\_10}(c) := \left| \begin{array}{l} N \leftarrow \text{rows}(c) \\ L \leftarrow \text{cols}(c) \\ \text{for } j \in 1..N \\ \quad v_j \leftarrow \sum_{k=1}^{L} \left(c_{j,k} \cdot 2^{L-k} \right) \\ v \end{array} \right.$$

$$cc := \begin{pmatrix} 1 & 0 & 1 & 1 & 0 & 1 & 0 & 1 & 1 \\ 0 & 1 & 1 & 0 & 0 & 1 & 1 & 0 & 1 \\ 1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 & 1 \end{pmatrix}$$

$$\text{cod2\_10}(cc) = \begin{pmatrix} 363 \\ 205 \\ 455 \end{pmatrix}$$

Рисунок 4.4 - Результат работы программы

На рисунке 4.5 представлен пример программы, которая преобразует матрицу чисел, записанных в коде грея в вектор десятичных чисел.

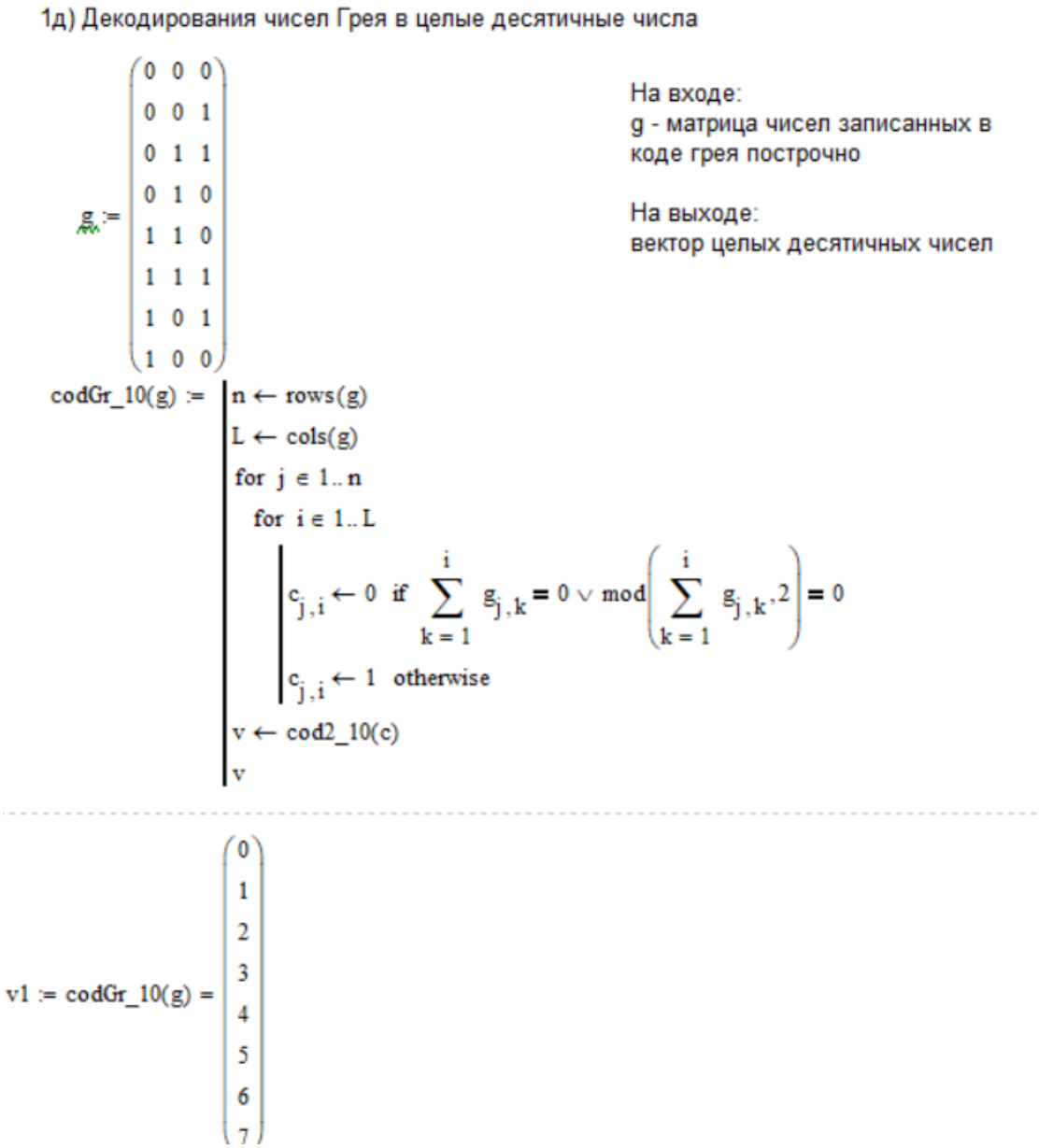

Рисунок 4.5 - Результат работы программы

На рисунке 4.6 представлен пример программы, которая преобразует вещественное число в десятичном коде в число, записанное в двоичном коде.

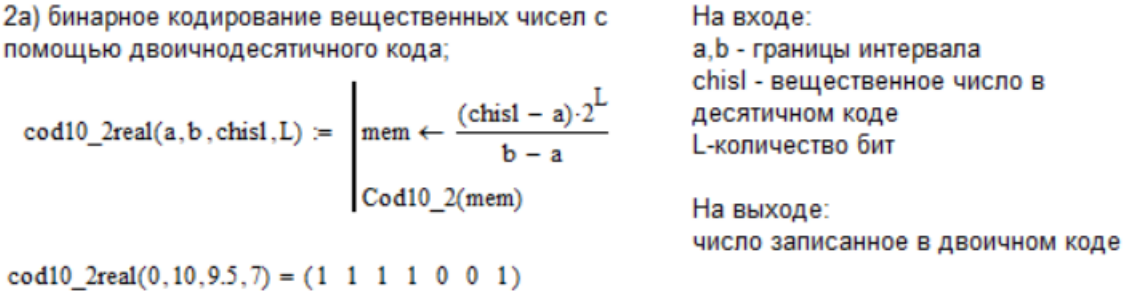

Рисунок 4.6 - Результат работы программы

На рисунке 4.7 представлен пример программы, которая преобразует вещественное

число в десятичном коде в число, записанное в двоичном коде грея.

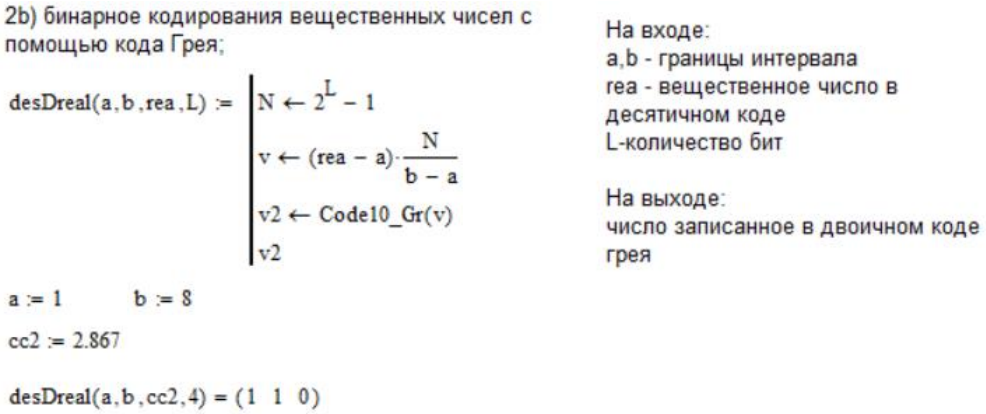

Рисунок 4.7 - Результат работы программы

На рисунке 4.8 представлен пример программы, которая преобразует вещественное число в двоичном коде в вещественное число, записанное в десятичном коде.

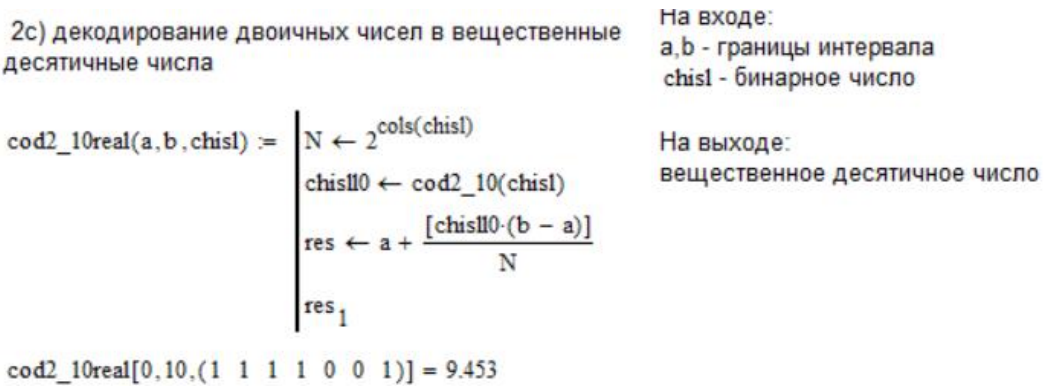

Рисунок 4.8 - Результат работы программы

На рисунке 4.9 представлен пример программы, которая преобразует вещественное число в двоичном коде в вещественное число, записанное в десятичном коде грея.

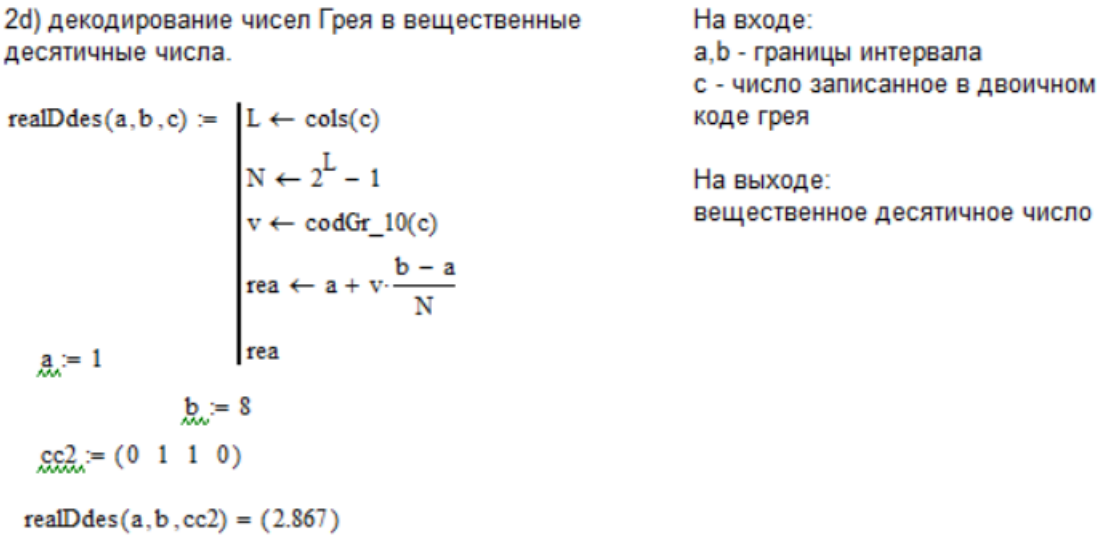

Рисунок 4.9 - Результат работы программы

На рисунке 4.10 представлен пример программы, которая генерирует случайные вещественные числа в заданных диапазонах.

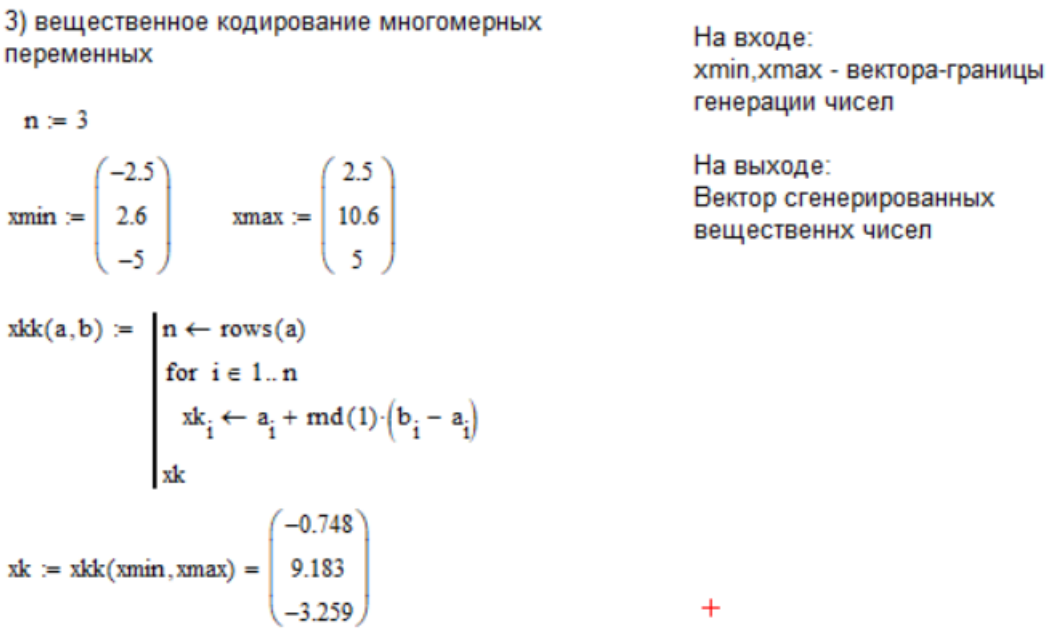

Рисунок 4.10 - Результат работы программы

Вывод

Во время выполнения лабораторной работы я приобрел опыт использования программы Mathcad 15.

Средствами Mathcad были написаны программы для кодирования и декодирования целых чисел с помощью двоично-десятичного кода.

Была написаны программы, которая преобразует десятичное целое число в двоичный код грея и которая преобразует матрицу двоичных чисел, записанных в коде грея в вектор десятичных чисел.

Были написаны программы для кодирования и декодирования вещественных чисел с помощью двоично-десятичного кода.

Были написаны программы бинарного кодирования и декодирования вещественных чисел с помощью кода Грея.