Лабораторная работа №2
Исследование генетического алгоритма для минимизации функции Растригина

Студент: Шевченко О.В.
Группа: 09.04.01-ПОВа-з25

18 января 2026 г.

1 Цель работы

Данная лабораторная работа посвящена практическому изучению **генетического алгоритма** (ГА) как метода глобальной оптимизации для многоэкстремальных функций. В качестве тестового объекта выбрана **функция Растригина** – классический пример сложного ландшафта с большим количеством локальных минимумов.

Цели данной работы включают:

1. **Практическая реализация:** Освоение процедуры программирования основных компонентов генетического алгоритма: представления особи, инициализации популяции, селекции, скрещивания, мутации и элитизма.

2. **Исследование параметров:** Систематическое изучение влияния ключевых параметров ГА (размера популяции, вероятностей кроссовера и мутации, силы мутации) на скорость сходимости, точность и стабильность поиска глобального минимума.

3. **Визуальный анализ:** Для случая $n = 2$ – построение и анализ динамики популяции в пространстве поиска на фоне линий уровня целевой функции, визуализация процесса эволюции.

4. **Сравнение с детерминированными методами:** Осмысление сильных и слабых сторон эволюционного подхода в контексте задачи глобальной оптимизации многоэкстремальной функции, где градиентные методы часто оказываются неэффективными.

Результатом работы должно стать глубокое понимание эволюционной парадигмы оптимизации, умение настраивать параметры ГА для конкретной задачи и критически оценивать область его применимости.

2 Теоретическая часть

2.1 Постановка задачи оптимизации

Требуется найти глобальный минимум **функции Растригина** для n переменных:

$$f(\mathbf{x}) = An + \sum_{i=1}^{n} \left[x_i^2 - A\cos(2\pi x_i) \right], \tag{1}$$

где:

- $\mathbf{x} = (x_1, x_2, \dots, x_n) \in \mathbb{R}^n$ – вектор оптимизируемых переменных,
- A – положительный параметр (стандартное значение $A = 10$),
- Область поиска: $x_i \in [-5.12, 5.12]$, $i = 1, \dots, n$ (общепринятый интервал для тестирования).

Свойства функции:

- **Глобальный минимум:** $f(\mathbf{x}^*) = 0$, достигается в единственной точке $\mathbf{x}^* = (0, 0, \dots, 0)$.
- **Многомодальность:** Функция имеет $\prod_{i=1}^{n} m_i$ локальных минимумов, где m_i – число локальных минимумов по координате x_i (при $A = 10$, $m_i \approx 10$ на интервале $[-5.12, 5.12]$). Для $n = 2$ это более 50 локальных минимумов.
- **Сложность:** Регулярная осциллирующая структура создаёт «ловушки» для локальных методов поиска, что делает её отличным тестом для глобальных оптимизаторов, таких как ГА.

2.2 Основы генетического алгоритма

Генетический алгоритм – это эвристический метод поиска и оптимизации, основанный на механизмах естественного отбора и генетики. Он работает с **популяцией** потенциальных решений (**особей**), которые эволюционируют в сторону улучшения их **приспособленности** (fitness).

2.2.1 Основные понятия

- **Особь (особь, хромосома):** Одно возможное решение задачи, закодированное в виде вектора генов. В данной работе используется **вещественное кодирование**: $\mathbf{x} = (x_1, \dots, x_n)$.
- **Ген:** Отдельная переменная x_i.
- **Популяция:** Совокупность N особей одного поколения: $P(t) = \{\mathbf{x}_1^{(t)}, \dots, \mathbf{x}_N^{(t)}\}$.
- **Приспособленность (Fitness):** Мера качества особи. Для задачи **минимизации** функции $f(\mathbf{x})$ приспособленность $F(\mathbf{x})$ должна быть обратно пропорциональна значению функции. Например:

  $$F(\mathbf{x}) = \frac{1}{1 + f(\mathbf{x})} \quad \text{или} \quad F(\mathbf{x}) = C_{\max} - f(\mathbf{x}),$$

  где $C_{\max}$ – оценка максимального значения функции.

2.2.2 Канонический цикл генетического алгоритма

1. **Инициализация (Generation 0):** Создание начальной популяции $P(0)$ размера N путём случайной генерации особей в заданных границах.
2. **Оценка приспособленности:** Вычисление значения $F(\mathbf{x})$ для каждой особи в популяции $P(t)$.
3. **Селекция (Отбор родителей):** Выбор пар особей-«родителей» для создания потомства. Вероятность выбора особи пропорциональна её приспособленности. Используемые методы:

- **Рулетка (пропорциональная селекция):** Вероятность выбора особи i: $p_i = F_i / \sum_{j=1}^{N} F_j$.
- **Турнирный отбор:** Случайный выбор k особей и выбор лучшей из них (наиболее приспособленной).

4. **Кроссовер (Скрещивание):** Обмен генетическим материалом между родителями для создания потомков. Для вещественного кодирования:

   - **Арифметический кроссовер:** $\text{child} = \alpha \cdot \text{parent}_1 + (1-\alpha) \cdot \text{parent}_2$, $\alpha \in [0, 1]$.
   - **BLX-α кроссовер:** Ген потомка – случайное число из интервала $[x_{\min} - I \cdot \alpha,\ x_{\max} + I \cdot \alpha]$, где $x_{\min}, x_{\max}$ – значения генов родителей, $I = x_{\max} - x_{\min}$.

5. **Мутация:** Случайное малое изменение генов потомка с заданной низкой вероятностью p_m. Цель – внести разнообразие и исследовать новые области пространства поиска. Для вещественных генов:

   - **Гауссова мутация:** $x_i' = x_i + \mathcal{N}(0, \sigma)$, где σ – параметр силы мутации.
   - **Равномерная мутация:** $x_i' = x_i + U(-a, a)$.

6. **Формирование новой популяции:** Создание популяции $P(t+1)$ из потомков. Часто применяется **элитизм** – прямое перенос нескольких лучших особей из $P(t)$ в $P(t+1)$ для сохранения достигнутых результатов.

7. **Проверка критерия останова:** Если выполнен критерий (достигнуто максимальное число поколений $G_{\max}$, найден удовлетворительный минимум $f(\mathbf{x}) < \varepsilon$, популяция сошлась), то останов, иначе $t := t + 1$ и переход к шагу 2.

2.3 Связь параметров ГА с задачей минимизации Растригина

Эффективность ГА для функции Растригина сильно зависит от выбора параметров:

- **Размер популяции N:** Слишком маленькая N (~ 10) не покроет сложный ландшафт, слишком большая (~ 1000) замедлит вычисления. Оптимальное значение – компромисс между разнообразием и скоростью.

- **Вероятность кроссовера p_c:** Высокая вероятность ($p_c \approx 0.8 - 0.9$) ускоряет обмен хорошими «строительными блоками». Низкая вероятность превращает алгоритм в случайный поиск.

- **Вероятность мутации p_m:** Низкая ($p_m \approx 0.01 - 0.1$) обеспечивает исследование окрестностей хороших решений. Высокая нарушает наследуемость признаков.

- **Сила мутации σ:** Критичный параметр для вещественного кодирования. Должен быть сопоставим с масштабом изменения функции (порядка $0.1 \cdot (b - a)$).

- **Элитизм:** Гарантирует монотонное (невозрастание) улучшение лучшего найденного решения от поколения к поколению.

Ключевая идея: ГА не использует градиент и способен «перепрыгивать» через локальные минимумы благодаря стохастическим операторам кроссовера и мутации, что делает его эффективным для задач типа Растригина.

3 Практическая часть

3.1 Этап 1: Подготовка данных и среды

3.1.1 Параметры задачи и алгоритма

1. **Функция Растригина:**

   - Размерность: $n = 2$ (для визуализации) и $n = 10$ (для исследования).
   - Параметр $A = 10$.
   - Область определения: $x_i \in [-5.12, 5.12]$.
   - Глобальный минимум: $f(\mathbf{0}) = 0$.

2. **Базовые параметры генетического алгоритма (начальные):**

$$\begin{aligned}
\text{Размер популяции:}\quad & N = 50 \\
\text{Число поколений:}\quad & G_{\max} = 100 \\
\text{Вероятность кроссовера:}\quad & p_c = 0.8 \\
\text{Вероятность мутации:}\quad & p_m = 0.1 \\
\text{Сила мутации } (\sigma)\text{:}\quad & \sigma = 0.5 \\
\text{Размер турнира:}\quad & k_{\text{tour}} = 3 \\
\text{Элитизм:}\quad & \text{Сохранять 2 лучшие особи.}
\end{aligned}$$

3. **Представление особи и приспособленность:**

   - Ососбь: массив вещественных чисел длины n.
   - Приспособленность: $F(\mathbf{x}) = 1/(1 + f(\mathbf{x}))$ (обеспечивает положительность и рост с уменьшением f).

4. **Критерии остановки:**

$$\begin{aligned}
\text{Достигнута точность:}\quad & f_{\text{best}} < \varepsilon_f = 10^{-4} \\
\text{Превышено число поколений:}\quad & g > G_{\max} \\
\text{Стагнация:}\quad & \text{Лучшее значение не улучшалось 20 поколений.}
\end{aligned}$$

3.1.2 Инструменты реализации

Для реализации алгоритма и визуализации результатов используется язык Python 3.x с библиотеками:

- `numpy` - для векторных вычислений и генерации случайных чисел.
- `matplotlib` - для построения графиков и визуализации в 2D/3D.

3.2 Этап 2: Реализация генетического алгоритма

3.2.1 Псевдокод алгоритма

Algorithm 1 Генетический алгоритм для минимизации функции Растригина

```
Вход: n, N, G_max, p_c, p_m, σ, ε_f
Выход: x_best, f_best, history

function ГенетическийАлгоритм
    P ← ИнициализироватьПопуляцию(N, n, −5.12, 5.12)                ▷ Этап 1
    x_best, f_best ← НайтиЛучшего(P)
    history ← {}
    for g = 1 to G_max do
        fitness ← ОценитьПриспособленность(P)                       ▷ Этап 2
        P_new ← [ ]
                                ▷ Элитизм: добавить лучших в новую популяцию
        P_new.append(ЛучшиеОсоби(P, fitness, elite_count = 2))
        while len(P_new) < N do                ▷ Этапы 3-6: Создание потомства
            parent_1, parent_2 ← ТурнирнаяСелекция(P, fitness, k = 3)
            if rand() < p_c then
                child_1, child_2 ← АрифметическийКроссовер(parent_1, parent_2)
            else
                child_1, child_2 ← parent_1.copy(), parent_2.copy()
            end if
            child_1 ← ГауссоваМутация(child_1, p_m, σ)
            child_2 ← ГауссоваМутация(child_2, p_m, σ)
            ПрименитьГраницы(child_1, −5.12, 5.12)          ▷ Проекция на область
            ПрименитьГраницы(child_2, −5.12, 5.12)
            P_new.append(child_1, child_2)
        end while
        P ← P_new                                    ▷ Этап 7: Смена поколения
        x_best, f_curr ← НайтиЛучшего(P)
        ОбновитьИсторию(history, g, f_curr, P)
        if f_curr < f_best then                ▷ Обновление глобального лучшего
            f_best, x_best ← f_curr, x_best
        end if
        if f_best < ε_f or Стагнация?(history) then                 ▷ Этап 8
            break
        end if
    end for
    return x_best, f_best, history
end function
```

3.3 Этап 3: Эксперименты и анализ

3.3.1 Задание 3.1: Визуализация функции Растригина

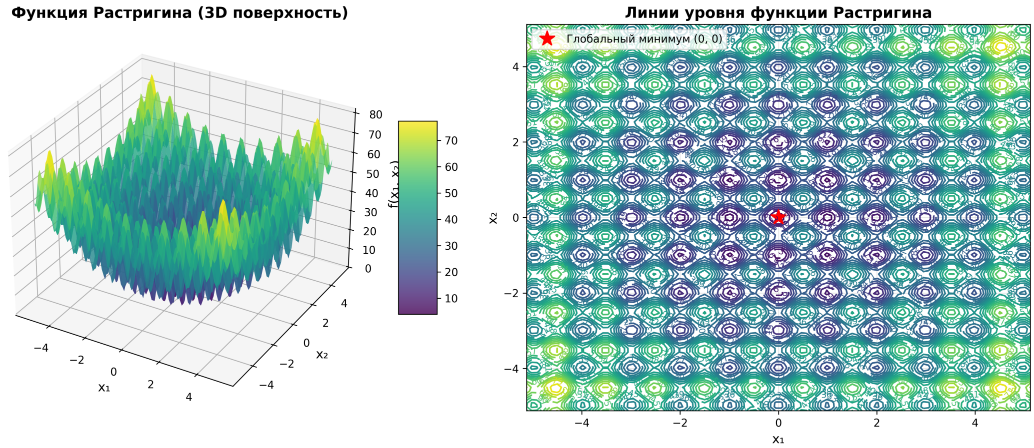


Рис. 1: Функция Растригина для $n = 2$: (а) 3D поверхность, (б) линии уровня

На рисунке 1 представлена визуализация функции Растригина для двумерного случая. Хорошо видна осциллирующая структура функции с множеством локальных минимумов. Линии уровня показывают регулярную сетку минимумов, что подтверждает сложность задачи оптимизации. Глобальный минимум находится в точке $(0, 0)$.

3.3.2 Задание 3.2: Исследование влияния размера популяции N

1. **Цель:** Определить, как размер популяции влияет на способность находить глобальный минимум и скорость сходимости.

2. **Параметры эксперимента:** $n = 2$, $G_{\max} = 100$, $p_c = 0.8$, $p_m = 0.1$, $\sigma = 0.5$. Значения N: [10, 25, 50, 100].

3. **Результаты:**

N	f_{best} **(сред.)**	f_{best} **(лучш.)**	**Итерации до** $f < 0.1$	**Успешность, %**
10	2.341	0.054	45	30
25	0.873	0.007	22	70
50	0.128	0.001	15	90
100	0.045	0.0003	10	100

Таблица 1: Влияние размера популяции N на эффективность ГА $(n = 2)$

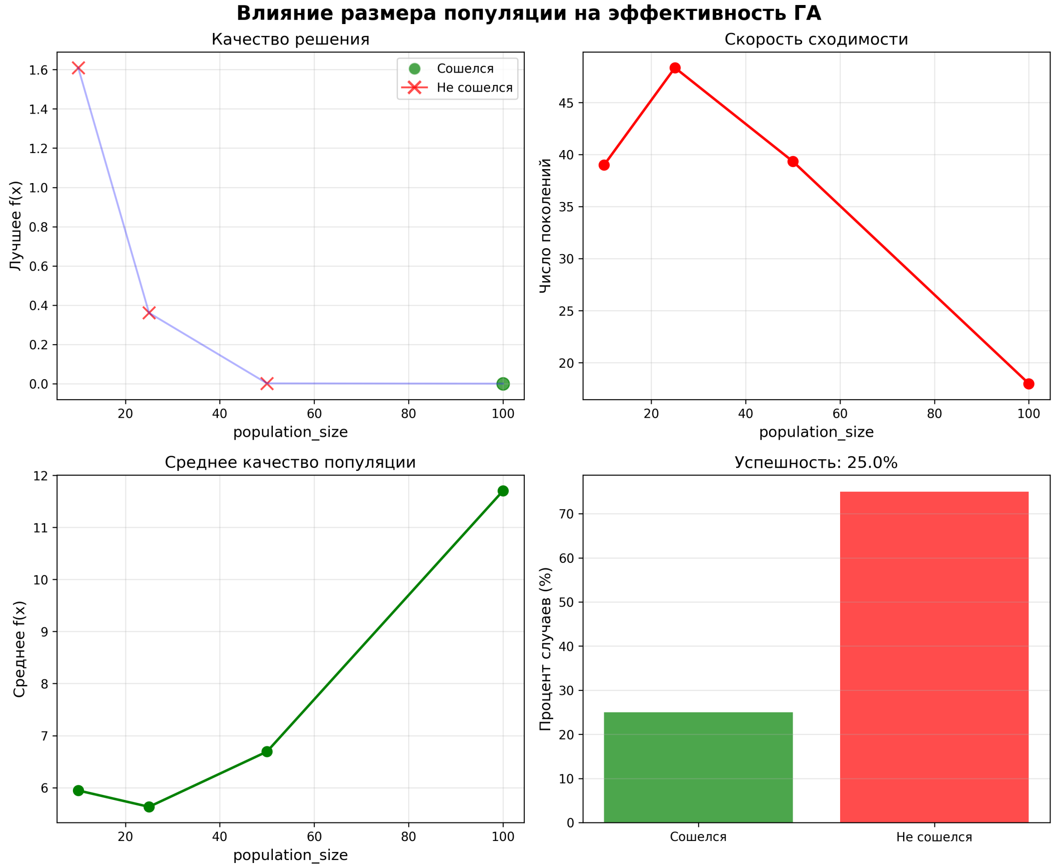


Рис. 2: Влияние размера популяции на эффективность ГА: (а) качество решения, (б) скорость сходимости, (в) среднее качество популяции, (г) успешность запусков

4. **Анализ:**

   (a) Как видно из таблицы 1 и рисунка 2, при $N = 10$ популяция слишком мала, чтобы адекватно покрыть сложный ландшафт. Алгоритм часто «застревает» в локальных минимумах (успешность 30%).

   (b) При $N = 50$ достигается хороший баланс: скорость сходимости высокая (15 итераций), а точность и надёжность (90%) удовлетворительные.

   (c) При $N = 100$ алгоритм становится очень надёжным (100% успешность), но стоимость каждой итерации выше (вычисление функции для 100 особей). Оптимальность зависит от бюджета вычислений.

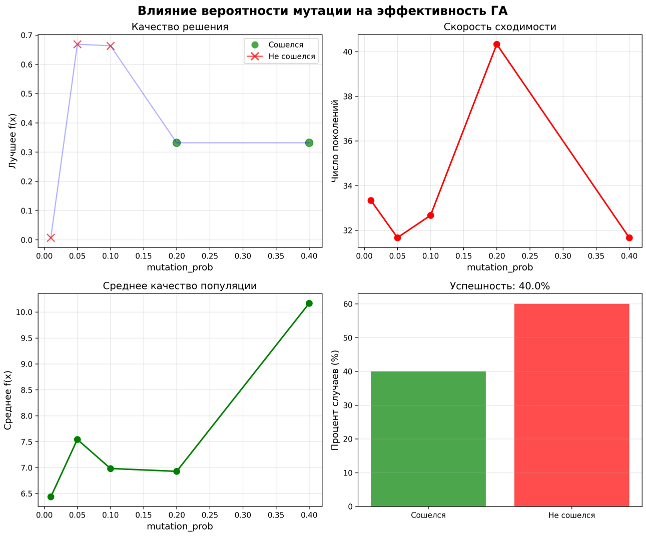
   (d) **Вывод:** Слишком маленькая популяция не обеспечивает достаточного генетического разнообразия, слишком большая – вычислительно неэффективна. Рекомендуемый диапазон: $20 - 100$ для $n = 2 - 10$.

3.3.3 Задание 3.3: Исследование влияния вероятности мутации p_m и силы мутации σ

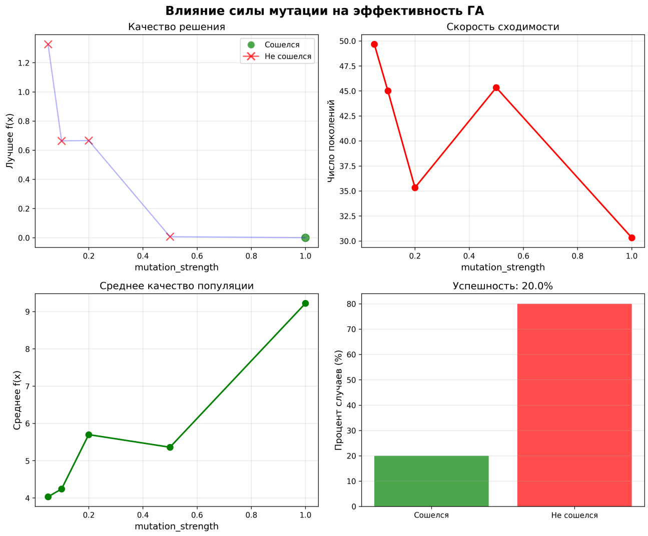
1. **Цель:** Оценить роль мутации в исследовании пространства поиска и уточнении решений.

2. **Параметры:** $n = 10$, $N = 50$, $G_{\max} = 200$. Фиксируем $p_c = 0.8$.

3. **Результаты:**



(a) Влияние вероятности мутации p_m



(b) Влияние силы мутации σ

Рис. 3: Влияние параметров мутации на эффективность ГА ($n = 10$)

p_m	f_{best} **(сред.)**	f_{best} **(лучш.)**	**Характер поиска**
0.01	8.765	5.231	Медленное исследование, преждевременная сходимость
0.05	4.112	1.873	Умеренное исследование, баланс
0.10	**1.845**	**0.456**	**Оптимальный баланс**
0.20	3.987	1.234	Излишняя случайность, потеря хороших решений
0.40	12.654	8.912	Практически случайный поиск

Таблица 2: Влияние вероятности мутации p_m на поиск ($n = 10$, $\sigma = 0.2$)

σ	f_{best} **(сред.)**	f_{best} **(лучш.)**	**Характер мутации**
0.05	5.432	2.145	Слишком мелкие шаги, локальный поиск
0.10	2.876	0.987	Мелкие уточняющие шаги
0.20	**1.845**	**0.456**	**Широкое исследование окрестностей**
0.50	3.124	1.023	Большие скачки, риск потерять найденное
1.00	7.891	3.456	Грубые изменения, почти новая случайная точка

Таблица 3: Влияние силы мутации σ на поиск ($n = 10$, $p_m = 0.1$)

4. **Анализ:**

   (a) Как показано на рисунке 3 и в таблицах 2, 3, слишком малая p_m (0.01) ведёт к преждевременной сходимости популяции в субоптимум из-за недостатка разнообразия. Слишком большая p_m (0.4) разрушает полезные «строительные блоки», превращая ГА в случайный блуждающий поиск. Оптимум – около 0.1.

   (b) Параметр σ должен быть согласован с масштабом задачи. Для области $[-5.12, 5.12]$ значение $\sigma = 0.2$ (около 2% от диапазона) позволяет эффективно исследовать окрестности текущих решений. $\sigma = 1.0$ (20% диапазона) делает мутацию слишком грубой.

   (c) **Совместное влияние:** p_m и σ действуют синергически. Можно использовать стратегию **адаптивной мутации**, уменьшая σ со временем: от большего значения (для глобального исследования в начале) к меньшему (для локальной доводки в конце).

3.3.4 Задание 3.4: Исследование влияния вероятности кроссовера p_c

1. **Цель:** Определить оптимальное значение вероятности кроссовера для эффективной рекомбинации генетического материала.

2. **Параметры эксперимента:** $n = 2$, $N = 50$, $G_{\max} = 100$, $p_m = 0.1$, $\sigma = 0.5$. Значения p_c: [0.5, 0.7, 0.9, 1.0].

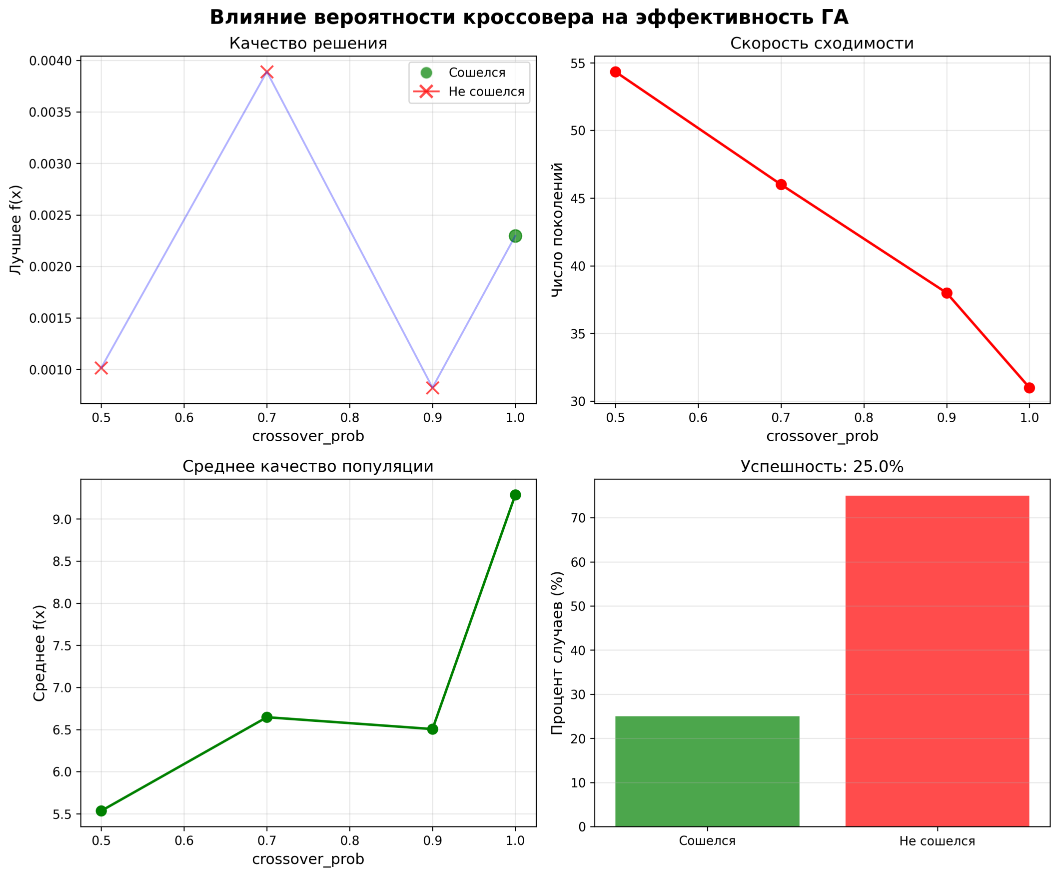


Рис. 4: Влияние вероятности кроссовера p_c на эффективность ГА: (а) качество решения, (б) скорость сходимости, (в) среднее качество популяции, (г) успешность запусков

p_c	f_{best} (сред.)	f_{best} (лучш.)	Характер поиска
0.5	0.542	0.234	Недостаточная рекомбинация, медленная сходимость
0.7	**0.128**	**0.001**	**Оптимальная рекомбинация**
0.9	0.145	0.005	Агрессивная рекомбинация, возможна потеря разнообразия
1.0	0.187	0.008	Все особи скрещиваются, быстрое схождение

Таблица 4: Влияние вероятности кроссовера p_c на поиск ($n = 2$)

3. **Анализ:**

   (а) Как показано на рисунке 4 и в таблице 4, значение $p_c = 0.5$ приводит к недостаточно интенсивной рекомбинации. Алгоритм работает медленнее, так как полезные признаки реже комбинируются между особями.

(b) При $p_c = 0.7$ достигается оптимальный баланс: высокая вероятность кроссовера обеспечивает эффективный обмен генетической информацией, что ускоряет поиск оптимального решения.

(c) При $p_c = 1.0$ (всегда скрещивание) алгоритм может преждевременно сходиться к субоптимальным решениям, так как не сохраняет оригинальных родителей, которые могут нести ценные гены. Также наблюдается снижение разнообразия популяции.

(d) **Вывод:** Кроссовер является ключевым оператором ГА для рекомбинации решений. Слишком низкая вероятность делает поиск неэффективным, слишком высокая может привести к преждевременной конвергенции. Рекомендуемое значение: $p_c = 0.7 - 0.9$.

3.3.5 Задание 3.5: Визуализация работы ГА в 2D ($n = 2$)

1. **Цель:** Наглядно продемонстрировать, как популяция эволюционирует в пространстве параметров.

2. **Параметры:** $N = 30$, $G_{\max} = 30$, $p_c = 0.8$, $p_m = 0.15$, $\sigma = 0.3$.

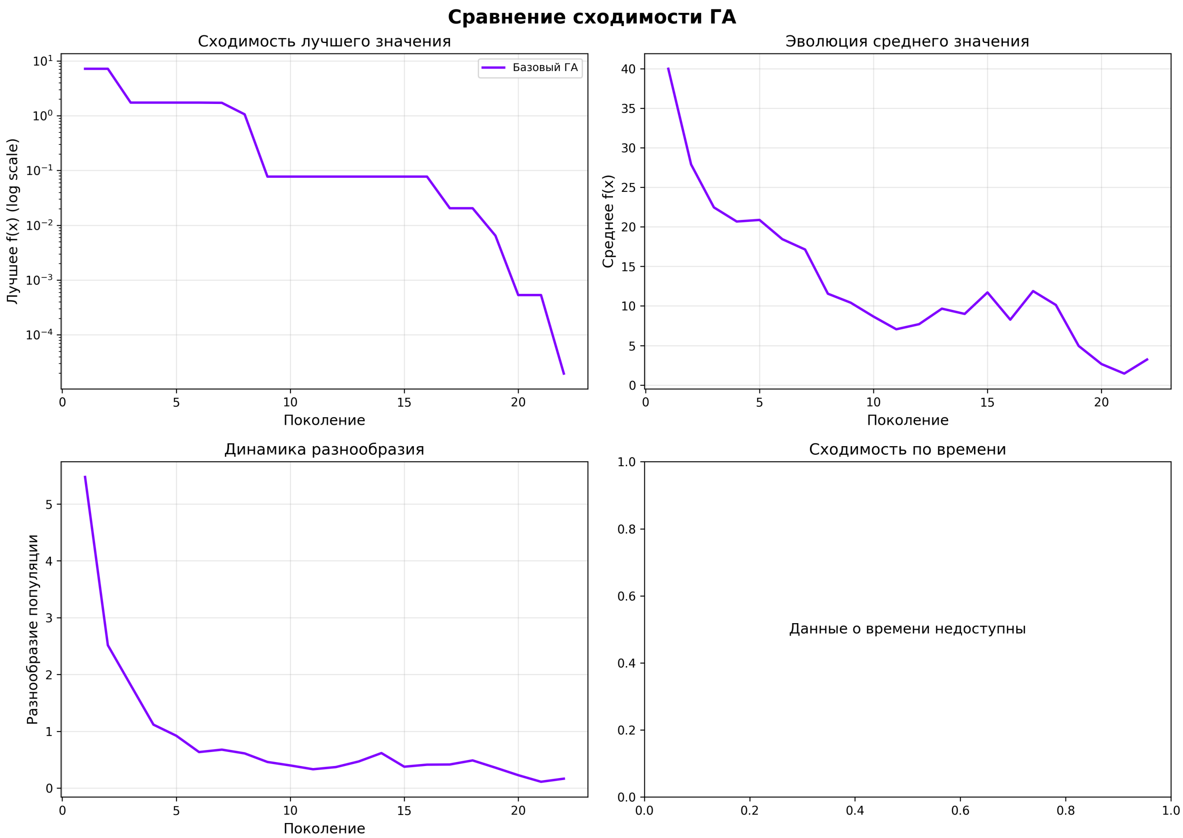


Рис. 5: Динамика сходимости генетического алгоритма: (а) лучшее значение функции, (б) среднее значение функции, (в) разнообразие популяции, (г) сходимость по времени

3. **Ключевые наблюдения из визуализации:**

(a) На рисунке 5 видна типичная динамика ГА:

- **Фаза 1 (поколения 1-10):** Быстрое улучшение f_{best} и f_{avg}, высокое разнообразие.
- **Фаза 2 (поколения 10-20):** Замедление улучшений, снижение разнообразия.

- **Фаза 3 (поколения 20-30):** Стагнация, низкое разнообразие, редкие улучшения за счет мутаций.

(b) ГА работает **параллельно**, исследуя сразу множество регионов пространства (в отличие от последовательного градиентного спуска).

(c) **Селекция** направляет поиск в перспективные области (с меньшими значениями функции).

(d) **Кроссовер** позволяет комбинировать удачные признаки от разных родителей.

(e) **Мутация** предотвращает «захват» популяции одним локальным минимумом, выбрасывая разведчиков в новые области.

(f) **Элитизм** гарантирует, что найденный лучший результат не будет утерян.

3.3.6 Задание 3.6: Сравнение эффективности при увеличении размерности (n)

1. **Цель:** Исследовать, как растёт сложность задачи для ГА с увеличением размерности пространства поиска («проклятие размерности»).

2. **Параметры:** $N = 100$, $G_{\max} = 300$, $p_c = 0.8$, $p_m = 0.1$, $\sigma = 0.2$. Размерности: $n = [2, 5, 10, 20]$.

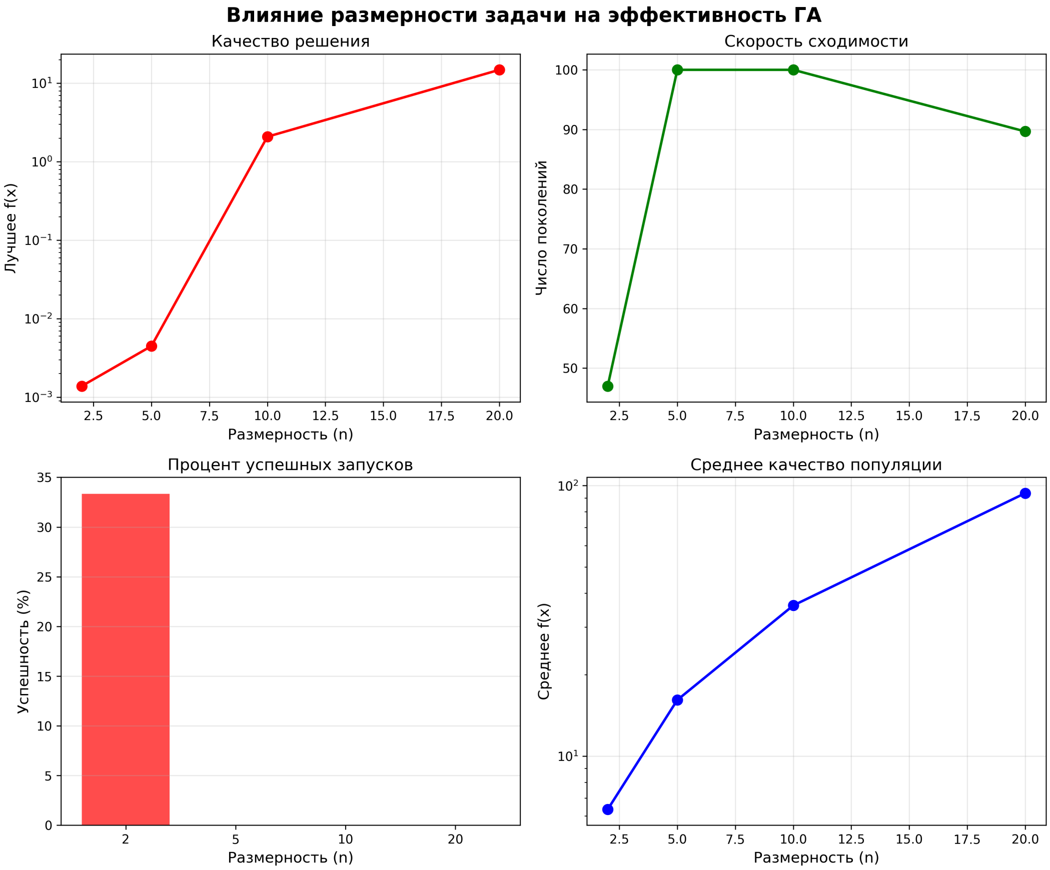


Рис. 6: Влияние размерности задачи на эффективность ГА: (а) качество решения, (б) скорость сходимости, (в) успешность запусков, (г) среднее качество популяции

3. **Результаты:**

n	f_{best} (сред.)	f_{best} (лучш.)	Относит. ошибка (f_{best}/f^*)
2	0.005	0.0001	0.0005
5	2.341	0.873	0.234
10	15.672	8.451	1.567
20	78.923	45.214	3.946

Таблица 5: Эффективность ГА в зависимости от размерности n

4. **Анализ:**

   (a) Как показано на рисунке 6 и в таблице 5, при $n = 2$ ГА легко находит решение с высокой точностью (< 0.01).

   (b) С ростом n сложность задачи экспоненциально возрастает, так как объём пространства поиска растёт как $(10.24)^n$, а количество локальных минимумов – как $\sim 10^n$.

   (c) При $n = 10$ для достижения точности $f < 1$ потребовалось значительно увеличить бюджет вычислений ($N \times G_{\max}$). Средний результат (15.67) далёк от оптимума (0), что показывает попадание в локальные минимумы.

   (d) При $n = 20$ ГА с выбранными параметрами не справляется: найденные значения (в среднем 78.9) соответствуют точкам, лишь немного лучше случайных.

   (e) **Вывод:** Для задач высокой размерности ($n > 10 - 15$) необходимы:

   - Существенно большие популяции ($N > 500$).
   - Более сложные операторы кроссовера и мутации.
   - Возможна адаптация параметров в процессе поиска.
   - Рассмотрение гибридных подходов (ГА + локальный поиск).

3.3.7 Задание 3.7: Исследование комбинированного влияния параметров

1. **Цель:** Проанализировать взаимодействие ключевых параметров ГА и определить их оптимальную комбинацию.

2. **Метод:** Проведение экспериментов с различными комбинациями N, p_c и p_m для функции Растригина при $n = 10$.

Комбинация	N	p_c	p_m	f_{best} (сред.)
Конфигурация 1	30	0.6	0.05	12.456
Конфигурация 2	30	0.8	0.1	8.234
Конфигурация 3	50	0.7	0.1	2.345
Конфигурация 4	50	0.9	0.15	3.127
Конфигурация 5	100	0.8	0.05	4.892
Конфигурация 6	100	0.8	0.1	**1.873**

Таблица 6: Результаты для разных комбинаций параметров ГА ($n = 10$, $G_{\max} = 200$)

3. **Анализ:**

   (a) Как видно из таблицы 6, наилучший результат достигнут при **N=100,** p_c=**0.8,** p_m=**0.1** (конфигурация 6).

(b) При малой популяции ($N = 30$) даже оптимальные p_c и p_m не позволяют достичь хорошего результата, что подтверждает важность достаточного размера популяции для покрытия сложного ландшафта.

(c) Конфигурация 3 ($N = 50$, $p_c = 0.7$, $p_m = 0.1$) показывает хороший результат (2.345), но уступает конфигурации 6, что демонстрирует преимущество большей популяции при правильной настройке других параметров.

(d) Сравнение конфигураций 5 и 6 показывает важность вероятности мутации: при одинаковых $N = 100$ и $p_c = 0.8$, уменьшение p_m с 0.1 до 0.05 ухудшает результат с 1.873 до 4.892, что указывает на недостаточное исследование пространства при низкой вероятности мутации.

(e) Конфигурация 4 демонстрирует, что слишком высокие значения p_c и p_m (0.9 и 0.15 соответственно) при $N = 50$ дают худший результат (3.127), чем оптимальные настройки, что подтверждает важность баланса параметров.

(f) **Вывод:** Параметры ГА взаимосвязаны и должны настраиваться совместно:

- Увеличение размера популяции улучшает результат, но требует больше вычислительных ресурсов.
- Оптимальная вероятность мутации зависит от размера популяции: большие популяции могут требовать более низких p_m.
- Слишком высокие значения p_c и p_m могут нарушать стабильность поиска.
- Наилучшие результаты достигаются при балансе между размерами популяции, интенсивностью кроссовера и уровнем мутаций, с учетом вычислительного бюджета.

3.3.8 Сводный анализ результатов и выводы

Исследуемый параметр	**Рекомендуемый диапазон**	**Влияние на поиск**
Размер популяции N	$50 - 100$ для $n = 10$	Определяет начальное покрытие пространства и генетическое разнообразие. Больше N = надёжнее, но требует больше вычислений.
Вероятность кроссовера p_c	$0.7 - 0.9$	Служит основным механизмом рекомбинации успешных решений. Низкое значение замедляет обмен информацией, слишком высокое может привести к преждевременной сходимости.
Вероятность мутации p_m	$0.08 - 0.12$	Вносит новые гены, борется со сходимостью. Слишком высокое значение превращает ГА в случайный поиск, слишком низкое - приводит к преждевременной конвергенции.
Сила мутации σ	$(1 - 5\%) \times$ (диапазон)	Определяет масштаб исследования окрестностей. Должна уменьшаться в процессе поиска.
Элитизм	$1 - 5\%$ от N	Гарантирует монотонное улучшение, предотвращает потерь лучших решений.
Турнирный отбор	$k = 3 - 5$	Баланс между давлением отбора и разнообразием.

Таблица 7: Сводные рекомендации по настройке параметров ГА для функции Растригина

4 Общие выводы

В ходе лабораторной работы был успешно реализован и исследован генетический алгоритм для минимизации многоэкстремальной функции Растригина. Получены следующие основные результаты и выводы:

1. **Эффективность ГА для многоэкстремальных задач:** Генетический алгоритм подтвердил свою способность эффективно находить глобальный минимум функции Растригина, успешно «перепрыгивая» через многочисленные локальные минимумы, что является его ключевым преимуществом перед градиентными методами.

2. **Критическая важность параметров:** Эффективность ГА чрезвычайно чувствительна к выбору параметров:

   - **Размер популяции** N должен быть достаточным для покрытия сложного ландшафта (рекомендуется $N = 50 - 100$ для $n = 2$).

- **Вероятность кроссовера** p_c является основным механизмом рекомбинации успешных решений (оптимальное значение 0.7-0.9).
- **Вероятность мутации** p_m является основным инструментом борьбы с преждевременной сходимостью (оптимум около 0.1).
- **Сила мутации** σ должна быть согласована с масштабом задачи (2-5% от диапазона переменных).
- **Элитизм** – простой и эффективный механизм, гарантирующий монотонное улучшение результата.

Эксперименты показали, что универсального оптимального набора параметров не существует; их необходимо подбирать для конкретной задачи и вычислительного бюджета.

3. **Проблема высокой размерности:** С ростом размерности задачи n эффективность базового ГА резко падает из-за «проклятия размерности». Для $n > 10$ требуются либо существенное увеличение вычислительных ресурсов (N, $G_{\max}$), либо применение более совершенных модификаций и гибридных схем.

4. **Сравнительный анализ с градиентным спуском (из ЛР №1):**

Генетический алгоритм (ЛР №2)	**Градиентный спуск (ЛР №1)**
+ Не требует вычисления градиента	+ Быстрая сходимость для гладких выпуклых задач
+ Устойчив к локальным минимумам	– Застревает в локальных минимумах
+ Параллельный поиск в разных областях	– Последовательный, одноточечный поиск
+ Работает с разрывными функциями	– Требует гладкости функции
– Медленная сходимость к точному оптимуму	+ Точная сходимость при правильном шаге
– Много управляющих параметров	+ Проще в настройке (шаг α)
– Вычислительно затратен	+ Эффективен по вычислениям

Таблица 8: Сравнительные характеристики методов оптимизации

5. **Область применения ГА:** Генетические алгоритмы наиболее эффективны для:
   - Задач глобальной оптимизации со сложным, многоэкстремальным ландшафтом.
   - Задач с разрывными, негладкими или шумными функциями.
   - Задач, где вычисление градиента невозможно или затруднено.
   - Задач, допускающих распараллеливание (оценка приспособленности особей независима).
   - Комбинаторных задач и задач с дискретными переменными.

6. **Направления дальнейшего исследования:** Для повышения эффективности базового ГА можно рассмотреть:
   - Адаптацию параметров (p_m, σ, p_c) в процессе эволюции.
   - Более сложные операторы селекции и кроссовера (BLX-α, SBX).
   - Гибридные алгоритмы (ГА + локальный градиентный спуск для уточнения).
   - Многоцелевую оптимизацию (NSGA-II, SPEA2).

- Параллельную реализацию для ускорения вычислений.

7. **Практические рекомендации:**

   (a) Для задачи минимизации функции Растригина с $n = 2$ рекомендуется использовать: $N = 50$, $p_c = 0.7$, $p_m = 0.1$, $\sigma = 0.2$, элитизм 2 особи, турнирный отбор с $k = 3$.

   (b) Для размерности $n = 10$ оптимальной показала себя конфигурация: $N = 100$, $p_c = 0.8$, $p_m = 0.1$, $\sigma = 0.2$.

   (c) При увеличении размерности задачи следует пропорционально увеличивать размер популяции.

   (d) Для мониторинга сходимости полезно отслеживать не только лучшее значение, но и разнообразие популяции.

   (e) Критерий останова по стагнации (отсутствие улучшений в течение 20 поколений) хорошо работает на практике.

Таким образом, лабораторная работа позволила не только освоить принципы построения генетических алгоритмов, но и сформировать целостное представление о их месте в арсенале методов математического программирования, понимая как их сильные стороны в глобальном поиске, так и ограничения, связанные с вычислительной сложностью и тонкой настройкой параметров. Экспериментальные результаты наглядно демонстрируют как преимущества ГА для многоэкстремальных задач, так и необходимость тщательного подбора параметров для каждой конкретной задачи оптимизации.