МИНОБРНАУКИ РОССИИ
САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ
ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ
«ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)
Кафедра МО ЭВМ

ОТЧЕТ

по лабораторной работе №5

по дисциплине «Основы машинного обучения»

Тема: Генетические алгоритмы

Вариант 3

Студентка гр. 1304 Чернякова А.Д.

Преподаватель Жангиров Т.Р.

Санкт-Петербург
2024

Цель работы.

Решить задачу с помощью генетического алгоритма.

Задание

Задача о марсианском броске мяча. Марсианин Сёма пытается попасть мячом в точку находящуюся на краю обрыва в M (задается пользователем) метрах от него на том уровне, что и Сёма . Мяч при броске летит по закону движения тела брошенного под углом к горизонту (известно, что ускорение свободного падения на Марсе равно 3.71 м/с^2). Необходимо определить, под каким углом (от 0 до $\pi/2$) и с какой скоростью (не менее 0) нужно бросить мяч, чтобы он попал в желаемую точку. Нежелательно, чтобы мяч перелетел точку, так как он упадет с обрыва и его придется искать. Также известно, что на расстоянии m (задается пользователем) метров от Сёмы находится стена высотой h (задается пользователем) метров, которая останавливает полет мяча.

Все данные создаются студентом. Необходимо найти такие параметры, при которых решение находится почти для любых данных.

Реализация решения рекомендуется проводить с помощью библиотеки DEAP, но можно использовать другие библиотеки или собственную реализацию (в таком случае в отчете нужно дать подробное описание кода)

Выполнение работы.

1. **Постановка задачи**

1.1. Опишем задачу в математическом виде.

Задача о марсианском броске мяча сводится к нахождению оптимального угла θ и скорости v0, при которых мяч попадет в точку на краю обрыва на расстоянии M метров, не перелетев её и преодолев стену на расстоянии m метров с высотой h метров. Ускорение свободного падения на Марсе g=3.71 м/с².

Для описания данной задачи воспользуемся формулами движение тела, брошенного под углом к горизонту (выводить не будем, взяты из интернета).

Расчет дальности полета мяча, брошенного под углом к горизонту, представлен в формуле 1.1:

$$L = \frac{v_0^2 \cdot \sin(2\theta)}{g}$$

Формула 1.1 - Дальность полета мяча,

где L — дальность полета,
v0— начальная скорость,
θ — угол броска,
g — ускорение свободного падения (3.71).

Также необходимо учесть, что мяч должен преодолеть стену на расстоянии m (задается пользователем) метров с высотой h (задается пользователем) метров. Условие преодоления стены описано в формуле 1.2.

$$m \cdot \tan(\theta) - \frac{1}{2} \cdot g \cdot \frac{m^2}{v_0^2 \cdot \cos^2(\theta)} \geq h$$

Формула 1.2 - Условие преодоления стены

где m — расстояния от Семы до стены,
h — высота стены.

Итак, изначально известны ускорение свободного падения на Марсе g, расстояние от Семы до точки на краю обрыва M, расстояние от Семы до стены m и высота стены h. Необходимо найти начальную скорость v0 и угол броска θ. Данная задача решается генетическим алгоритмом.

1.2. В генетических алгоритма хромосома — это структура данных, представляющая потенциальное решение задачи. Хромосома состоит из набора генов, каждый из которых кодирует определенный параметр или характеристику решения.

В данной задаче задаче о марсианском броске мяча хромосома представляет собой пару значений (v0,θ), где:

- v0— начальная скорость броска мяча (м/с).
- θ — угол броска мяча относительно горизонтали (рад).

Каждое из этих значений является геном хромосомы.

Ограничения на хромосому:

1) Начальная скорость v0 должна быть положительной.
2) Угол броска θ должен находиться в пределах от 0 до π/2.

1.3. Перед тем как алгоритм начнет работу, нужно определить функцию годности (фитнес-функцию). Для этого нужно определить мягкие и жесткие ограничения.

Мягкое ограничение заключается в минимизации разницы между расчетным и фактическим расстоянием от точки броска до точки падения мяча.

Жесткое ограничение заключается в необходимости преодоления мячом стены, расположенной на расстоянии *m* метров от точки броска, высотой h метров.

Получим, что метрика будет представлять три вещественных числа – модуль расстояния между рассчитанной и фактической точкой падения мяча, ошибка мягкого ограничения и ошибка жесткого ограничения.

На основе этих аспектов напишем фитнес-функцию, которая отвечает за оценку качества решения. Реализация функции представлена в листинге 1.3.

Листинг 1.3 - Реализация функции годности

```
g = 3.71  # Ускорение свободного падения на Марсе
M = 100   # Расстояние до обрыва
m = 50    # Расстояние до стены
h = 15    # Высота стены

LIGHT_PENALTY_PARAM = 10
HARD_PENALTY_PARAM = 80

def evaluate(individual):
    v0, theta = individual

    if v0 < 0:
         v0 = -v0
    if theta < 0:
         theta = -theta
    if theta > np.pi/2:
         theta = np.pi - theta

    # Расчет дальности полета
    L = (v0 ** 2) * np.sin(2 * theta) / g
    dist_to_target = M - L
    light_penalty = abs(dist_to_target) * LIGHT_PENALTY_PARAM if
dist_to_target < 0 else 0

    # Проверка пересечения стены
    height = m * np.tan(theta) - (1 / 2) * g * ((m / (v0 *
np.cos(theta))) ** 2)
    hard_penalty = (h-height) * HARD_PENALTY_PARAM if height < h
else 0


    return abs(dist_to_target) + light_penalty + hard_penalty,
```

Здесь заданы начальные значения для величин M, m, h. В функции годности коэффициент для мягкого ограничения равен 10, для жесткого - 80. Если мяч перелетает цель, то добавляется штраф, пропорциональный модулю расстояния между рассчитанной и фактической точкой падения мяча и коэффициенту для мягкого ограничения. Вычисляется высота траектории мяча в точке, где находится стена. Если высота меньше заданной высоты стены h, то добавляется штраф, пропорциональный разнице этих высот и коэффициенту для жесткого ограничения. Далее для скорости и угла напишем свойство симметрии для соблюдения ограничение на хромосому, так как значения могут не войти в допустимый диапазон в ходе скрещивания и мутации. Функция возвращает итоговую оценку, которая учитывает: абсолютное значение разницы между целевым и фактическим расстоянием, штраф за перелет цели и штраф за непреодоление стены. Запятая в конце возвращаемого значения в функции evaluate нужна для совместимости с библиотекой DEAP.

2. Выполнение ГА

Сначала напишем функции генерации случайного угла броска и случайной скорости броска, учитывая ограничения. Реализацию смотреть в листинге 2.0.

Листинг 2.0 - Генерация случайного угла броска и случайной скорости броска

```
def angle_generating():
    return random.uniform(0, np.pi / 2)

def speed_generating():
    return random.uniform(0, 100)
```

2.1. Функция для запуска ГА представлена в листинге 2.1.

Листинг 2.1 - Запуск ГА

```
POPULATION = 100
HALL_OF_FAME = 5
P_CROSS = 0.9 # вероятность скрещивания
P_MUTATION = 0.2 # вероятность мутации
MAX_GENERATIONS = 100

def genetic_algorythm():
  # Определение типа фитнеса и хромосомы
  creator.create("FitnessMin", base.Fitness, weights=(-1.0,))
  creator.create("Individual", list, fitness=creator.FitnessMin)

  # Инициализация объектов для работы с хромосомами
  toolbox = base.Toolbox()

  # Определение функций для генерации и оценки индивидов
  toolbox.register("angle", angle_generating)
  toolbox.register("speed", speed_generating)
  toolbox.register("individual", tools.initCycle,
creator.Individual, (toolbox.speed, toolbox.angle))
  toolbox.register("population", tools.initRepeat, list,
toolbox.individual)

  # Регистрация операций ГА
  toolbox.register("evaluate", evaluate)
  toolbox.register("mate", tools.cxBlend, alpha=0.5)
  toolbox.register("mutate", tools.mutGaussian, mu=0, sigma=[5,
0.1], indpb=0.2)
  toolbox.register("select", tools.selTournament, tournsize=3)

  pop = toolbox.population(n=POPULATION)
  hof = tools.HallOfFame(HALL_OF_FAME)
  stats = tools.Statistics(lambda ind: ind.fitness.values)
  stats.register("avg", np.mean)
  stats.register("std", np.std)
  stats.register("min", np.min)
  stats.register("max", np.max)
  pop, logbook = algorithms.eaSimple(pop, toolbox, cxpb=P_CROSS,
mutpb=P_MUTATION, ngen=MAX_GENERATIONS, stats=stats,
halloffame=hof, verbose=True)

  best = hof[0]
  v0, theta = best
  print(f"Best solution: speed = {v0}, angle = {theta}")

  # Расчет дальности полета
  L = (v0 ** 2) * np.sin(2 * theta) / g
  dist_to_target = M - L
  light_penalty = abs(dist_to_target) * LIGHT_PENALTY_PARAM if
dist_to_target < 0 else 0

  # Проверка пересечения стены
  height = m * np.tan(theta) - (1 / 2) * g * ((m / (v0 *
```

```
np.cos(theta))) ** 2)
  hard_penalty = (h-height) * HARD_PENALTY_PARAM if height < h
else 0

  print("Metric result: ", abs(dist_to_target))
  print("Light penalty: ", light_penalty)
  print("Hard penalty: ", hard_penalty)

  minFitnessValues, meanFitnessValues = logbook.select("min",
"avg")
  sns.set_style("whitegrid")

  plt.plot(minFitnessValues, color='red')
  plt.xlabel('Generation')
  plt.ylabel('Min')
  plt.show()

  plt.plot(meanFitnessValues, color='blue')
  plt.xlabel('Generation')
  plt.ylabel('Average Fitness')
  plt.show()

  return v0, theta
```

Здесь:

- POPULATION - это количество индивидуумов (особей) в каждом поколении популяции. Чем больше популяция, тем больше разнообразие мы имеем, но тем больше времени и ресурсов требуется для вычисления каждого поколения.
- HALL_OF_FAME - это количество лучших индивидуумов, которые будут сохранены в Зале Славы (Hall of Fame). Эти индивидуумы обычно являются лучшими решениями, найденными генетическим алгоритмом на протяжении всех поколений.
- P_CROSS - это вероятность скрещивания между двумя индивидуумами в паре. Скрещивание позволяет создавать новых потомков, комбинируя характеристики родителей.
- P_MUTATION - это вероятность мутации для каждого гена в индивидууме. Мутация вносит случайные изменения в гены, что помогает

алгоритму исследовать пространство решений и избегать застревания в локальных оптимумах.

- MAX_GENERATIONS - это максимальное количество поколений, которое будет создано генетическим алгоритмом. После достижения этого количества поколений алгоритм завершает работу. Увеличение этого параметра может привести к лучшему решению, но может также потребовать больше времени для вычислений.

Далее создаются необходимые классы для определения типа фитнеса и хромосомы:

- creator.create("FitnessMin", base.Fitness, weights=(-1.0,)): Эта строка создает новый класс FitnessMin, который наследуется от базового класса Fitness из библиотеки DEAP. Мы указываем weights=(-1.0,), чтобы определить, что мы решаем задачу минимизации: более низкие значения фитнес-функции будут считаться лучше.
- creator.create("Individual", list, fitness=creator.FitnessMin): Здесь создается класс Individual, который является списком и имеет атрибут fitness типа FitnessMin. Этот класс представляет собой хромосому или индивидуум в генетическом алгоритме.

Создается объект toolbox, который позволяет регистрировать различные операции для работы с хромосомами.

Регистрируются функции для генерации начальных значений индивидов и популяции:

- toolbox.register("angle", angle_generating): регистрируется функция angle_generating в toolbox под именем "angle". Эта функция будет использоваться для генерации случайных значений углов.

- toolbox.register("speed", speed_generating): регистрируется функция speed_generating в toolbox под именем "speed". Эта функция будет использоваться для генерации случайных значений скоростей.
- toolbox.register("individual", tools.initCycle, creator.Individual, (toolbox.speed, toolbox.angle)): регистрируется функция tools.initCycle в toolbox под именем "individual". tools.initCycle используется для создания индивидуумов (хромосом). Первый аргумент (creator.Individual) указывает на тип создаваемых объектов (в данном случае, объекты класса Individual, который мы определили ранее). Второй аргумент - это набор функций (в данном случае, toolbox.speed и toolbox.angle), которые будут вызываться для генерации значений для каждого гена хромосомы. Для каждого индивидуума функции будут вызываться поочередно, начиная с первой.
- toolbox.register("population", tools.initRepeat, list, toolbox.individual): регистрируется функция tools.initRepeat в toolbox под именем "population". Эта функция используется для создания начальной популяции. Первый аргумент (list) указывает на тип, в который будет помещена каждая особь. Второй аргумент - это функция для создания каждой особи (в данном случае, toolbox.individual).

Далее регистрируются операции ГА, такие как функция оценки, операторы скрещивания, мутации и отбора:

- toolbox.register("evaluate", evaluate): регистрируем функцию evaluate в toolbox под именем "evaluate". Эта функция будет использоваться для оценки фитнеса (приспособленности) каждого индивидуума в популяции.
- toolbox.register("mate", tools.cxBlend, alpha=0.5): регистрируем функцию tools.cxBlend в toolbox под именем "mate". Это один из методов скрещивания, предоставляемых библиотекой DEAP. Он использует смешивание (blending) для создания потомства путем комбинирования хромосом родителей с использованием параметра смешивания alpha. Параметр alpha контролирует, какую долю информации о каждом гене

ребенка будет унаследована от каждого из родителей. Если alpha равно 0.0, каждый ген потомка будет идентичен соответствующему гену одного из его родителей, а если alpha равно 1.0, каждый ген будет равновероятно выбран из двух родителей. В данном случае, alpha=0.5 означает, что каждый ген потомка будет в среднем на половину наследоваться от одного родителя и на половину от другого, что делает этот метод скрещивания подходящим выбором для обеспечения некоторого уровня разнообразия в потомстве.

- toolbox.register("mutate", tools.mutGaussian, mu=0, sigma=[5, 0.1], indpb=0.2): регистрируем функцию tools.mutGaussian в toolbox под именем "mutate". Эта функция будет использоваться для мутации генов в индивидуумах. tools.mutGaussian применяет гауссовское распределение к каждому гену индивидуума, чтобы внести случайные изменения. Параметр mu указывает среднее значение, sigma - стандартное отклонение, indpb - вероятность мутации для каждого гена.
- toolbox.register("select", tools.selTournament, tournsize=3): регистрируем функцию tools.selTournament в toolbox под именем "select". Эта функция будет использоваться для выбора индивидуумов для следующего поколения на основе турнирного отбора. tournsize=3 указывает количество индивидуумов, которые участвуют в каждом турнире. Победитель турнира (лучший индивидуум среди участников) выбирается в новое поколение.

Далее происходит инициализация начальной популяции, зала славы (hall of fame) и статистики для мониторинга процесса эволюции:

- pop = toolbox.population(n=POPULATION): Создается начальная популяция размером POPULATION (количество индивидуумов в популяции), используя функцию toolbox.population, которая была зарегистрирована ранее. Каждый индивидуум будет создан с помощью

функции toolbox.individual, которая в свою очередь использует зарегистрированные функции для генерации скорости и угла.

- hof = tools.HallOfFame(HALL_OF_FAME): Создается зал славы размером HALL_OF_FAME, который будет хранить лучших индивидуумов, когда они будут обнаружены во время выполнения генетического алгоритма. Это помогает отслеживать лучшие решения на протяжении всего процесса оптимизации.
- stats = tools.Statistics(lambda ind: ind.fitness.values): Создается объект Statistics, который будет использоваться для сбора статистических данных в процессе эволюции. Аргумент lambda ind: ind.fitness.values указывает, как извлекать значения фитнеса из каждого индивидуума. В данном случае, каждый индивидуум имеет только одно значение фитнеса, поэтому используется ind.fitness.values.
- stats.register("avg", np.mean), stats.register("std", np.std), stats.register("min", np.min), stats.register("max", np.max): Здесь регистрируются различные статистические функции (np.mean, np.std, np.min, np.max) для сбора средних, стандартных отклонений, минимальных и максимальных значений фитнеса в каждом поколении. Это позволит отслеживать изменения фитнеса в процессе оптимизации и оценить прогресс алгоритма.

Далее запускается стандартный генетический алгоритм (EA - Evolutionary Algorithm) на основе простой эволюции с использованием следующих параметров:

- pop: начальная популяция индивидуумов.
- toolbox: набор инструментов, содержащий функции для оценки, скрещивания, мутации и выбора индивидуумов.
- cxpb: вероятность скрещивания (вероятность того, что два выбранных индивидуума будут скрещены).

- mutpb: вероятность мутации (вероятность того, что каждый ген каждого индивидуума будет подвержен мутации).
- ngen: количество поколений (количество итераций эволюционного процесса).
- stats: объект, содержащий информацию о статистике процесса эволюции (средний фитнес, минимальный фитнес, максимальный фитнес и т. д.).
- halloffame: объект, содержащий лучшие индивидуумы, обнаруженные в процессе эволюции.
- verbose=True: параметр, указывающий на вывод информации о прогрессе эволюции во время выполнения алгоритма.

  Эта функция возвращает два значения:
- pop: конечная популяция индивидуумов после завершения эволюционного процесса.
- logbook: объект, содержащий журнал эволюции, в котором сохраняются статистические данные о каждом поколении (средний фитнес, минимальный фитнес, максимальный фитнес и т. д.).

Выбирается лучший индивидуум из зала славы (hof), который содержит наилучшие индивидуумы, обнаруженные в процессе эволюции. Распаковываются значения скорости (v0) и угла (theta) из лучшего индивидуума. Рассчитываются метрики и выводятся результаты метрик: модуль расстояния между рассчитанной и фактической точкой падения мяча, штрафы за мягкое и жесткое нарушения.

Извлекаются значения минимального и среднего фитнеса из журнала эволюции. Графики показывают изменение минимального (красный цвет) и среднего (синий цвет) фитнеса в зависимости от поколения.

Обоснуем, почему именно эти методы скрещивания, мутации и отбора были выбраны для решения задачи с марсианским броском мяча.

Для скрещивания был выбран метод смешения потому, что он используется для вещественных чисел и ген каждого потомка генерируется из диапазона, задаваемого родителем. Это позволяет улучшать следующие поколения. alpha = 0.5.

Метод mutGaussian (гауссовая мутация) был выбран потому, что он хорошо подходит для непрерывных переменных, таких как скорость и угол в нашей задаче. Этот метод работает следующим образом: к каждому гену хромосомы добавляется случайное значение, взятое из нормального распределения с заданным средним (mu) и стандартным отклонением (sigma). Обоснование выбора: Гауссовая мутация позволяет вносить небольшие изменения в значения генов, что способствует исследованию близлежащих областей пространства решений и помогает избежать преждевременной сходимости к локальному оптимуму. Значения параметров sigma=[5, 0.1] были выбраны для обеспечения разумного уровня вариации в скорости (v0) и угле (theta), а вероятность мутации indpb=0.2 (20%) обеспечивает баланс между стабильностью и разнообразием популяции. Данный метод мутации подходит лучше всего, так как используется для вещественных чисел по нормальному закону.

Метод selTournament (турнирный отбор) был выбран потому, что он эффективен и прост в реализации. Этот метод работает следующим образом: из популяции случайным образом выбираются (в данном случае 3) индивиды и из этих индивидов выбирается лучший (с наименьшей оценкой фитнес-функции) и добавляется в новый набор родителей.
Обоснование выбора: Турнирный отбор является достаточно устойчивым методом отбора, который позволяет сохранить высокоприспособленных индивидов в популяции, обеспечивая при этом шанс на размножение и менее приспособленных индивидов. Размер турнира tournsize=3 обеспечивает баланс между селективностью и разнообразием, что важно для предотвращения

преждевременной сходимости и обеспечения эффективного поиска глобального оптимума.

2.2. Подбор параметров для генетического алгоритма — это важный шаг, который напрямую влияет на его эффективность и производительность. Основные параметры, которые обычно подбираются, включают размер популяции, вероятность скрещивания, вероятность мутации, количество поколений и размер зала славы. Рассмотрим процесс подбора каждого из этих параметров.

Размер популяции (POPULATION = 100)

- **Подбор:**
  - **Малые размеры популяции (например, 20-50):** Могут приводить к быстрому сходимости, но с риском попадания в локальный минимум.
  - **Большие размеры популяции (например, 100-500):** Способствуют более тщательному исследованию пространства решений, но увеличивают вычислительную сложность.
  - **Оптимизация:** Обычно начинают с умеренных значений (например, 50-100) и регулируют их в зависимости от результатов. Если наблюдается преждевременная сходимость, размер популяции может быть увеличен.

В итоге было решено, что оптимальным размером популяции является 100 индивидов.

Размер зала славы (HALL_OF_FAME = 5)

- **Описание:** Зал славы сохраняет лучшие решения, найденные за весь процесс выполнения алгоритма.

- **Подбор:**
    - **Малый размер зала славы (например, 1-2):** Может не дать достаточного разнообразия для сохранения всех хороших решений.
    - **Большой размер зала славы (например, 10 и более):** Может приводить к избыточному сохранению, не сильно влияя на конечный результат.
    - **Оптимизация:** Обычно выбирают небольшой размер зала славы (например, 5), чтобы сохранить наиболее приспособленные индивиды.

Для нашей задачи подбирались значения от 2 до 10 и решено было выбрать 5.

Вероятность скрещивания (P_CROSS = 0.9)

- **Подбор:**
    - **Низкие значения (например, 0.5-0.7):** Могут замедлить процесс поиска, так как меньшее количество генов комбинируется для создания новых решений.
    - **Высокие значения (например, 0.8-1.0):** Способствуют быстрому исследованию новых областей пространства решений.
    - **Оптимизация:** Часто выбирается высокая вероятность скрещивания (например, 0.8-0.9) для максимизации генерации новых решений.

Вероятность скрещивания была взята по умолчанию как 0.9.

Вероятность мутации (P_MUTATION = 0.2)

- **Подбор:**

- **Низкие значения (например, 0.01-0.1):** Могут привести к недостаточной вариативности, повышая риск преждевременной сходимости.
- **Высокие значения (например, 0.2-0.5):** Могут вводить слишком много случайности, что усложняет сходимость.
- **Оптимизация:** Вероятность мутации обычно выбирается в умеренных значениях (например, 0.1-0.3), чтобы сохранить баланс между исследованием новых решений и эксплуатацией найденных.

Вероятность мутации была взята по умолчанию как 0.2.

Максимальное количество поколений (MAX_GENERATIONS = 100)

- **Описание:** Максимальное количество поколений определяет, сколько раз алгоритм будет обновлять популяцию.
- **Подбор:**
  - **Низкие значения (например, 20-50):** Могут быть недостаточными для достижения оптимального решения, особенно в сложных пространствах решений.
  - **Высокие значения (например, 100-200):** Позволяют более тщательно исследовать пространство решений, но увеличивают вычислительные затраты.
  - **Оптимизация:** Обычно начинают с умеренных значений (например, 100) и регулируют в зависимости от сходимости. Если алгоритм не сходит в пределах заданного числа поколений, можно увеличить их количество.

Количество поколения подбиралось с 1 до 100, и в итоге было решено, что большинство параметров сходилось за 100 поколений.

- **Величина штрафов (10, 80):** Выбирались 10-30 и 70-100. В итоге наилучшими оказались 10 для мягкого штрафа и 80 для жесткого штрафа.

В ходе подбора было замечено, что сильное влияние оказало правильно выбрать метод скрещивания. В данном случае получаем, что мяч не перелетает обрыв (ошибка 10^-11, что приблизительно 0). При выборе других методов перелет мяча достигал до нескольких сантиметров.

Подбором были выбраны необходимые методы и параметры.

2.3. Проведем оптимизацию для значений выбранных ранее: M=100, m=50, h=15. Код визуализации представлен в листинге 2.3.1. Графическое решение смотреть на рисунке 2.3.1.

Листинг 2.3.1 - Графическое решение задачи (траектория мяча)

```
v0, theta = genetic_algorythm()

# Визуализация траектории
import matplotlib.pyplot as plt

t = np.linspace(0, 2 * v0 * np.sin(theta) / g, num=500)
x = v0 * t * np.cos(theta)
y = v0 * t * np.sin(theta) - 0.5 * g * t**2

plt.plot(x, y)
plt.axvline(m, color='red', linestyle='--', label='Wall')
plt.axhline(h, color='red', linestyle='--')
plt.axvline(M, color='blue', linestyle='--', label='Target')
plt.title("Trajectory of the Ball")
plt.xlabel("Distance (m)")
plt.ylabel("Height (m)")
plt.legend()
plt.grid()
plt.show()
```

Этот код использует результаты генетического алгоритма (значения скорости и угла) для расчета траектории полета мяча и ее визуализации. Создается массив t, который представляет собой равномерно распределенные

значения времени от 0 до времени полета, рассчитанного на основе скорости v0 и угла theta. Вычисляются горизонтальные координаты полета мяча в зависимости от времени с использованием формулы движения равномерно ускоренного тела. Вычисляются вертикальные координаты полета мяча в зависимости от времени с использованием формулы движения равномерно ускоренного тела.

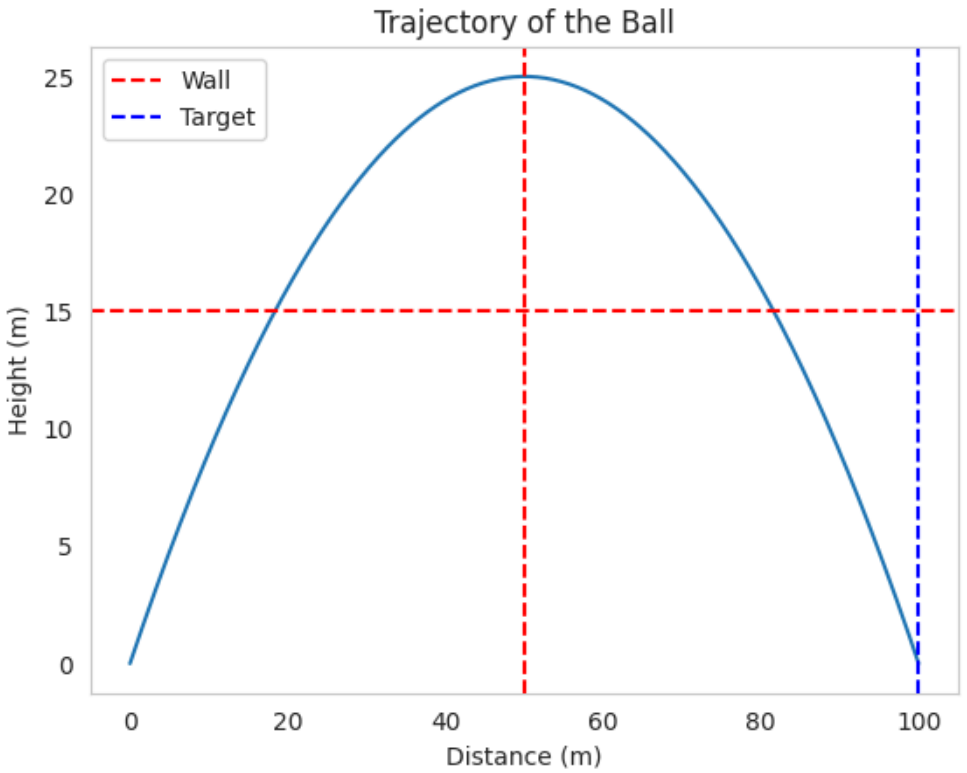

Рисунок 2.3.1 - Графическое решение задачи (траектория мяча)

Выведем лучший индивид и оценку метрики – абсолютное расстояние, ошибка мягкого ограничения и ошибка жесткого ограничения (см. листинг 2.3.2).

Листинг 2.3.2 - Оценка метрики

```
Best solution: speed = 19.26136029666157, angle =
0.7853727872305624
Metric result:  0.0
Light penalty:  0
```

```
Hard penalty:  0
```

Metric result равно 0, что указывает на то, что найденное решение идеально.

Light penalty равно 0, что указывает на отсутствие перелета целевой точки. Это хорошо, потому что показывает, что объект приземлился в пределах заданного расстояния.

Hard penalty также равно 0, что указывает на то, что объект пересек стену, что было одним из условий задачи.

2.4. Построим графики зависимости лучше и средней приспособленности в зависимости от поколения (графики отображаем отдельно для большей наглядности). Смотреть рисунки 2.4.1 и 2.4.2.

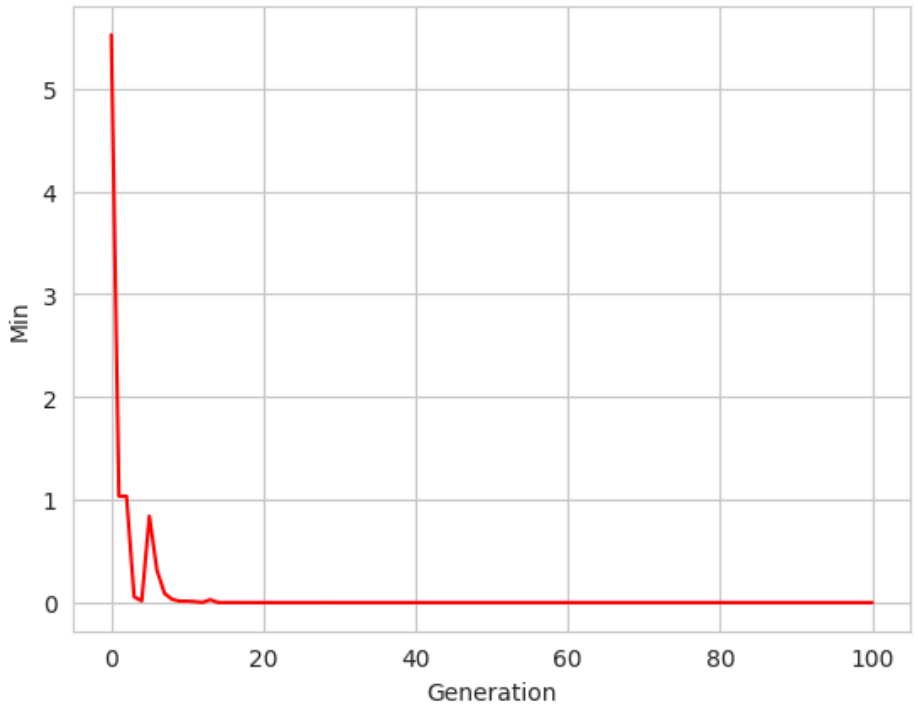

Рисунок 2.4.1 - Графики зависимости лучше приспособленности от поколения

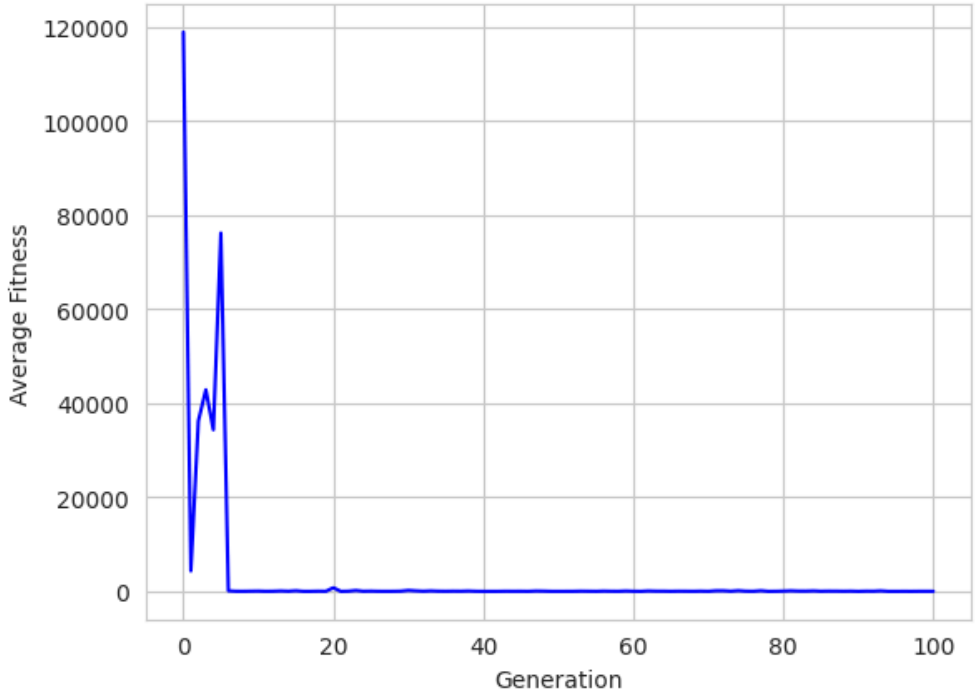

Рисунок 2.4.2 - Графики зависимости средней приспособленности от поколения

Так как задача минимизации, то чем ниже линия на графике, тем лучше приспособлено поколение.

График минимальной приспособленности показывает, как изменяется наименьшее значение приспособленности в каждом поколении. Если кривая убывает, это означает, что ГА продвигается в направлении улучшения лучших решений. В нуле значение огромное, так данные рандомно сгенерированы, далее кривая убывает. ГА успешно находит более приспособленные решения.

График средней приспособленности показывает среднее значение приспособленности в каждом поколении. По графику 2.4.2 видно, что он похож на график 2.4.1, так как они несомненно связаны, пики у них также наблюдаются в одинаковых диапахонвх.

Пики приспособленности объясняются тем, что поколение подвержено мутациям, и в некоторых из поколений мутант является наиболее приспособленным.

В последнем поколении пик не наблюдается и значение лучшей приспособленности находится близко к значению средней приспособленности и близко к 0, что означает верность полученного решения.

В приложении A, B и C представлены работа алгоритма и оценки метрики для разных начальных параметров. Во всех случаях мягкое и жесткое ограничение равно 0, модуль расстояния между рассчитанной и фактической точкой падения мяча находится в пределах 10^-12 - 10^-9, что можно округлить до 0. Задача решена верно.

Вывод.

В ходе выполнения работы был изучен генетический алгоритм, с помощью которого решена поставленная задача.

ПРИЛОЖЕНИЕ А

Листинг А.

M = 1000 # Расстояние до обрыва

m = 200 # Расстояние до стены

h = 50 # Высота стены

gen	nevals	avg	std	min	max
0	100	1.31102e+07	1.17563e+08	1.13114	1.17995e+09
1	84	9294.14	44952.9	1.13114	445046
2	85	5743.62	37198.3	1.82956	373924
3	87	5559.18	40561.8	1.82956	401993
4	93	463.579	1066.13	5.0116	6337.05
5	95	376.315	1182.93	0.894602	10111.4
6	92	154.35	204.005	1.33734	1047.86
7	87	117.211	251.567	1.33734	2026.39
8	97	136.487	223.09	1.13325	1419.47
9	97	185.697	410.229	1.07819	2575.53
10	94	43.6129	65.9107	1.2426	435.638
11	99	31.3574	106.442	0.444172	1003.78
12	87	21.5251	45.8947	0.115613	261.809
13	89	82.5213	453.195	0.444172	4034.32
14	90	78.8216	341.02	0.0449393	2207.6
15	92	32.5591	174.279	0.0449393	1697.17
16	97	21.0685	158.771	0.0453831	1594.27
17	96	43.6412	372.152	0.0453831	3738.82
18	91	52.8981	375.274	0.0199545	3706.53
19	95	17.6448	84.9704	0.0132026	777.565
20	87	43.4491	273.205	0.0143906	2093.3
21	93	2.54702	16.5179	0.0231795	166
22	90	30.4753	205.997	0.00460804	1593.71
23	90	1.90405	10.9652	0.0063437	107.413
24	91	56.9831	353.232	0.000537676	2538.75
25	90	15.2001	135.341	0.00167104	1360.17
26	98	6.70417	23.6651	0.00224979	172.797
27	95	34.4127	242.99	0.0020276	2062.39
28	84	3.70812	20.2597	0.0020961	148.012
29	90	27.9608	188.384	0.00204584	1556.67
30	85	22.7853	116.342	0.0020961	889.133
31	90	14.7776	93.4327	0.0020961	889.842
32	88	12.9056	93.8538	0.0020961	935.622
33	97	9.3143	46.1749	0.000156996	297.373
34	94	66.6982	383.866	0.0020961	2994.95
35	87	45.3385	311.855	0.00100479	2707.92
36	94	22.131	141.969	0.000818375	1232.16
37	93	7.33785	62.1433	0.000253398	622.468
38	92	14.0899	138.706	6.41584e-05	1394.18
39	92	8.02775	39.1937	0.000102038	280.54
40	92	47.2944	279.036	0.000345095	2363.96
41	89	25.6309	229.705	0.000345095	2304.91
42	94	9.69359	56.6861	5.0139e-05	468.078
43	95	0.712568	4.79923	5.0139e-05	44.9583
44	87	5.55264	24.0789	0.000113743	197.373
45	89	53.8137	410.89	4.24369e-05	3987.59
46	93	69.941	350.862	6.69524e-05	2954.78
47	96	19.0023	145.372	0.000187943	1439.59
48	90	76.431	519.009	2.0908e-05	4298.73
49	97	25.4864	238.453	2.1673e-05	2396.69
50	94	73.779	602.886	2.02276e-05	5983.33

```
51    91    23.6113     162.97      2.1741e-06  1224.62
52    91    15.3026     137.898     6.59512e-06 1384.31
53    90    35.3048     299.725     6.59512e-06 3007.51
54    89    14.305      120.748     3.87713e-06 1205.29
55    87    19.7986     136.316     4.71416e-07 1330.25
56    88    37.1108     253.355     4.71416e-07 1957.93
57    93    53.008      322.147     3.2423e-05  2643
58    95    21.7042     208.83      1.50862e-05 2099.04
59    95    107.363     610.511     2.24353e-06 4450.63
60    94    67.2071     478.209     2.11015e-06 4342.32
61    90    18.0869     166.466     6.57379e-06 1671.71
62    92    4.54283     25.5322     4.89311e-06 218.073
63    97    95.8961     453.412     1.81096e-06 2738.77
64    96    16.9713     113.532     1.81096e-06 1073.83
65    94    80.4591     565.022     3.52355e-06 5492.65
66    90    32.9413     321.647     1.25232e-06 3232.85
67    98    66.5693     385.173     3.90833e-07 3371.77
68    89    28.4047     140.179     2.02764e-07 1004.83
69    96    32.8366     252.644     9.23851e-08 2518.79
70    90    18.4916     141.22      9.23851e-08 1381.93
71    89    91.5168     523.153     1.36573e-07 3620.29
72    91    60.3311     323.324     1.74478e-07 2726.2
73    92    66.4407     378.763     4.06915e-08 3141.2
74    88    3.69196     27.7782     1.0042e-09  275.627
75    92    15.1938     86.7623     4.43049e-09 810.687
76    92    3.19267     28.8818     2.66537e-08 289.959
77    92    49.814      481.26      5.48005e-09 4837.69
78    97    24.0539     165.646     1.34207e-09 1376.33
79    84    26.979      253.732     2.96041e-10 2547.23
80    95    105.09      495.166     7.84041e-09 3378.26
81    92    2.68809     14.8602     1.4104e-09  127.465
82    92    30.7779     167.075     1.23091e-08 1215.21
83    95    39.6751     249.571     1.03717e-09 2171.86
84    98    5.49204     26.4588     6.72685e-10 173.58
85    92    17.0903     137.479     1.55717e-09 1363.28
86    90    16.2976     131.447     5.81508e-10 1309.71
87    94    50.4459     449.482     6.29484e-10 4500.69
88    92    61.4071     405.252     6.29484e-10 3298.98
89    85    6.48833     36.152      2.09525e-10 291.063
90    88    2.09651     14.9375     3.46404e-10 124.519
91    91    3.75766     22.4224     3.46404e-10 179.696
92    90    3.8051      21.428      3.75849e-10 183.965
93    92    72.9711     334.43      3.08432e-10 2353.22
94    92    56.0741     384.125     3.08432e-10 2966.69
95    92    5.58034     32.0718     3.1946e-11  255.127
96    89    57.4517     325.125     1.02773e-10 2861.93
97    88    36.8125     232.837     8.41283e-12 2229.53
98    92    1.16838     9.7801      8.41283e-12 97.2854
99    89    18.6259     117.663     8.41283e-12 959.877
100   84    7.74708     52.7193     2.38742e-12 510.173
Best solution: speed = 60.912054837473065, angle = 0.7915236163704374
Metric result:  2.3874235921539366e-12
Light penalty:  0
Hard penalty:  0
```

Листинг В.

M = 400 # Расстояние до обрыва

m = 100 # Расстояние до стены

h = 100 # Высота стены

gen	nevals	avg	std	min	max
0	100	9.85814e+06	9.75586e+07	5.84182	9.80552e+08
1	94	1.62748e+06	1.61086e+07	23.7931	1.61905e+08
2	98	4555.41	16007.2	39.0064	159305
3	97	1749.88	4129.38	1.33595	29029.4
4	95	13729.9	79351.9	0.387465	701838
5	98	2377.74	22102.7	0.315427	222277
6	92	91.1724	188.692	0.315427	1539.5
7	86	71.3987	154.842	0.113466	850.035
8	85	74.208	201.764	0.315427	1677.24
9	94	89.1679	290.297	0.321045	1879.49
10	91	91.6526	347.765	0.049827	2670.69
11	89	50.3282	202.132	0.101845	1363.29
12	96	67.3594	334.397	0.270908	2618.65
13	93	19.6696	104.334	0.0625493	1018.57
14	98	71.7005	271.591	0.0634417	1849.99
15	94	11.1533	27.3604	0.0858095	196.435
16	81	84.399	313.347	0.0508852	2152.28
17	91	26.39	174.483	0.00878926	1724.92
18	95	120.131	437.804	0.0199823	2561.08
19	91	102.085	415.165	0.0197938	2692.32
20	88	20.9682	145.802	0.0148315	1431.86
21	88	97.127	321.836	0.00890684	1516.83
22	93	93.9356	472.284	0.00931694	2795.88
23	84	60.2099	254.603	0.0159087	1499.95
24	94	93.3588	433.379	0.00494674	3126.92
25	95	38.5285	185.93	0.00494674	1184.95
26	89	55.0393	331.154	0.00494674	3012.98
27	94	55.0545	295.127	0.00504267	2396.62
28	91	49.4894	339.895	0.00156824	2887.45
29	88	60.6198	347.244	0.000240214	3195.98
30	96	2.73435	14.5324	0.00450709	108.02
31	91	27.1218	134.218	0.00934706	996.867
32	89	53.4661	332.707	0.00181483	2623.17
33	92	22.1272	111.396	0.00181483	770.045
34	95	55.4461	230.407	0.00102012	1634.22
35	88	40.8064	262.009	0.000232186	2449.13
36	95	45.9166	229.113	2.94493e-07	1858.48
37	94	15.0358	102.897	0.000281326	868.808
38	98	2.49209	14.938	9.18014e-05	127.547
39	94	26.2415	172.809	0.0005249	1644.41
40	92	87.4817	417.406	0.000191007	3592.62
41	98	65.2287	296.967	2.01138e-05	1931.07
42	90	93.2546	419.548	2.01138e-05	2909.32
43	94	68.6396	402.934	2.01138e-05	3343.5
44	90	48.4113	215.451	2.01138e-05	1421.63
45	91	44.3908	190.341	2.01138e-05	1088.16
46	92	53.6836	275.888	3.28335e-05	2101.09
47	97	45.5201	317.022	4.15642e-05	2529.57
48	93	25.7132	156.269	8.54076e-06	1257.43
49	94	6.14551	51.0706	1.17686e-06	510.029
50	92	10.4368	69.7244	1.17686e-06	611.903
51	91	5.97723	36.905	2.56579e-07	292.5
52	88	46.7833	244.376	1.15817e-06	1946.92

```
53    93    42.0874     196.015     4.31243e-06 1424.25
54    94    43.127      230.278     3.37132e-06 1733.95
55    93    120.255     549.187     5.33496e-08 3967.13
56    91    0.811298    5.8229      2.05785e-07 49.6671
57    92    67.1004     358.795     2.96716e-07 2426.73
58    90    1.15572     8.14972     2.43114e-07 79.3945
59    86    61.6674     294.955     1.94104e-07 2110.94
60    93    74.8739     343.702     9.98232e-08 2397.12
61    88    89.5044     395.128     1.82453e-08 2765.52
62    95    47.1803     276.083     4.58526e-08 2285.35
63    90    126.919     490.49      8.19787e-08 2830.4
64    94    34.6488     223.012     8.16954e-09 2127.34
65    93    16.1657     123.162     2.72927e-08 1159.77
66    94    70.4357     313.28      2.0327e-08  2334.27
67    88    48.1432     280.545     2.0327e-08  2263.84
68    91    24.3224     173.203     2.48315e-09 1667.89
69    95    54.3871     266.634     1.42345e-08 2005.46
70    86    41.9695     305.041     1.42345e-08 2926.42
71    91    29.4636     135.468     1.42345e-08 1005.63
72    99    100.57      501.153     1.34434e-08 3336.38
73    91    26.794      124.555     9.71761e-09 1017.82
74    86    48.6712     312.103     1.19693e-08 2725.63
75    93    85.5482     322.893     2.98962e-09 1910.6
76    96    81.1721     358.874     5.30179e-10 2024.1
77    89    89.2203     399.237     5.68002e-09 2440.19
78    88    53.8857     302.769     4.72789e-09 1829.77
79    93    94.7099     411.972     4.77485e-10 3016.95
80    86    42.3184     211.973     4.77485e-10 1843.23
81    96    64.7729     331.635     2.47542e-09 2899.07
82    100   19.065      110.82      3.01213e-10 1035.15
83    93    41.6322     279.922     3.01213e-10 2301.75
84    90    119.652     620.723     3.01213e-10 5155.63
85    93    11.8682     106.977     8.46967e-12 1073.73
86    96    47.2399     212.227     1.00044e-11 1304.61
87    97    22.1767     113.449     6.753e-11   822.521
88    93    33.3845     258.344     6.753e-11   2549.28
89    86    14.8157     142.807     2.5409e-11  1435.18
90    86    46.2523     338.259     2.25668e-11 3272
91    95    42.6348     257.998     3.79146e-11 2405.02
92    94    52.8934     304.084     1.90994e-11 2330.26
93    94    72.0033     317.015     7.95808e-12 1989.23
94    93    87.7136     402.993     1.28466e-11 2784.86
95    94    28.126      164.498     9.32232e-12 1375.31
96    91    99.451      386.357     1.87583e-12 2409.64
97    100   48.9207     268.226     1.13687e-12 1642.3
98    89    58.2085     370.889     1.08002e-12 2906.64
99    94    84.5372     383.938     1.08002e-12 2694.96
100   92    18.8976     107.199     9.54969e-12 750.91
Best solution: speed = 39.70356134505419, angle = 0.9574151975045301
Metric result:  1.0800249583553523e-12
Light penalty:  0
Hard penalty:  0
```

Листинг С.

M = 700 # Расстояние до обрыва

m = 200 # Расстояние до стены

h = 400 # Высота стены

gen	nevals	avg	std	min	max
0	100	2.03882e+06	1.16684e+07	111.891	05759e+08
1	92	56587.8	238558	33.2163	2.37401e+06
2	94	157156	1.13871e+06	7.641	1.12369e+07
3	96	38227.6	314848	2.11637	3.16699e+06
4	87	100061	979253	9.44478	9.84344e+06
5	88	1936.34	8398.3	0.743281	59982.7
6	92	809.65	1908.65	2.01446	11779.3
7	89	1434.37	9987.43	0.949034	100366
8	92	8058.73	75478.7	4.29324	759004
9	95	506.117	957.591	4.29324	5812.57
10	95	50913.2	499795	4.29324	5.02379e+06
11	94	581.365	1264.46	3.61139	7627.77
12	93	405.401	1006	1.03467	7632.3
13	93	296.218	513.642	1.07459	2251.92
14	96	3535.09	22480.4	1.22	184679
15	86	260.861	609.395	0.254642	4129.25
16	90	116.947	251.421	0.495312	1722.34
17	90	230.316	1033.94	1.40616	10251.5
18	93	3989.37	38543.3	0.0225265	387481
19	96	389.29	1304.14	0.0225265	9963.68
20	97	107.441	250.734	0.215231	1347.99
21	86	79.5901	196.249	0.359888	1321.33
22	88	187.969	692.687	0.135974	6416.6
23	95	230.533	1272.03	0.135974	11833.4
24	85	156.753	950.799	0.14303	9507.05
25	90	151.384	1118.54	0.026776	11221.7
26	97	362.029	2223.64	0.280709	20814.3
27	92	162.989	933.805	0.186488	9039.03
28	95	52.7389	140.997	0.0215809	1130.22
29	85	34.162	101.993	0.118868	839.945
30	90	156.688	1204.45	0.11613	12013.6
31	99	17.3395	72.6293	0.0597463	645.584
32	92	275.788	1685.16	0.00935453	15253.1
33	96	43.6532	340.366	0.00165667	3407.25
34	100	235.372	1150.07	0.0098448	7921.68
35	95	247.305	1670.09	0.0232667	15956.4
36	97	63.3988	610.078	0.00229033	6132.93
37	98	51.6035	490.198	0.0145139	4927.78
38	87	40.2821	242.568	0.000809088	1742.68
39	86	134.447	971.183	0.00156978	9429.6
40	94	107.213	996.719	0.00363993	10015.2
41	93	3.41971	14.0598	0.000926366	85.9696
42	98	107.994	926.671	0.00168734	9309.68
43	94	412.698	2014.65	0.00168734	14974.2
44	94	81.363	392.094	0.00053883	3198.69
45	96	81.0034	370.162	0.000493326	2471.9
46	96	257.799	1205.61	0.00112369	10506.9
47	86	33.5131	271.465	0.000468977	2693.87
48	97	116.695	691.608	7.57448e-05	5436.34
49	98	150.687	868.129	0.000247938	6575.19
50	95	197.149	1367.46	0.000217123	13532.6
51	93	8.84589	44.5148	0.000259652	316.248
52	94	171.708	785.643	0.000545844	5748.75

```
53    84    34.1624     226.154     0.000548535 2192.61
54    94    56.3097     435.176     5.72107e-05 4320.18
55    95    0.985094    6.85925     0.000109089 54.2989
56    91    43.1135     215.722     0.000108336 1372.33
57    90    75.9708     566.23      0.000113761 5409.83
58    85    62.0207     560.595     8.10686e-05 5627.01
59    90    4.36239     26.8549     8.10686e-05 195.812
60    89    391.057     1994.17     3.81845e-05 13301.4
61    96    63.6631     349.093     1.91039e-05 2661.81
62    88    112.933     965.557     1.91039e-05 9602.9
63    95    263.489     1608.11     1.56346e-05 14743.6
64    94    7.31266     40.0253     6.8796e-06  315.286
65    94    108.614     949.698     6.8796e-06  9506.1
66    97    43.3285     206.09      2.24713e-05 1449.9
67    91    16.8728     135.551     4.27285e-06 1347.06
68    94    127.579     661.702     2.41358e-05 5188.06
69    91    100.275     650.186     1.0328e-05  6153.04
70    97    17273.3     171051      6.86757e-06 1.7192e+06
71    95    163.661     1042.6      1.84451e-06 7946.96
72    89    127.062     702.672     9.16304e-06 4830.48
73    94    79.4639     747.672     1.29429e-06 7512.49
74    86    24.6685     171.673     1.29429e-06 1360.93
75    93    57.1157     271.434     1.8629e-05  1951.85
76    95    115.266     643.332     1.07878e-05 5309.07
77    92    50.3019     236.726     6.30724e-06 1690.64
78    92    864.891     6874.4      5.88769e-06 68172.7
79    94    109.323     961.596     4.96081e-06 9654.76
80    94    4891.37     43690.7     4.96081e-06 436928
81    92    1678.96     16196.9     1.76518e-07 162801
82    94    231.106     1643.15     1.76518e-07 15975.8
83    93    78.2173     473.856     8.73135e-07 4468.74
84    91    217.78      1021.63     6.78367e-07 7228.07
85    94    114.524     493.209     5.0994e-07  4074.48
86    96    199.003     1417.96     1.44332e-06 12798.3
87    90    67.1128     284.272     1.71541e-06 1932.95
88    94    109.385     998.558     1.22675e-06 10032.3
89    93    54.6014     253.991     1.22072e-08 1768.27
90    87    30.2146     175.577     2.34944e-07 1471.54
91    94    112.896     1089.72     6.75562e-08 10953.7
92    94    112589      1.11049e+06 4.63373e-07 1.11615e+07
93    90    65.6574     351.118     1.38234e-07 2544.88
94    97    102.951     739.299     3.85696e-08 7273.06
95    99    19.1675     121.093     6.85782e-09 1117.48
96    96    130.168     1059.8      1.89368e-09 10563.2
97    89    21.5574     183.734     2.23064e-08 1834.84
98    88    138.578     1025.33     6.81325e-09 10138
99    94    73.1733     457.601     2.81647e-08 4124.25
100   92    74.662      574.172     7.26686e-09 5516.47
Best solution: speed = 68.28257769436635, angle = 1.2754141007424902
Metric result:  1.8936816559289582e-09
Light penalty:  0
Hard penalty:  0
```