ЛАБОРАТОРНАЯ РАБОТА № 2

«Инструментальные средства ГА»

Цель работы: изучение основ работы с Genetic Algorithm в MatLAB, исследование экстремумов функций с помощью генетических алгоритмов.

Теоретическая часть
Общие сведения о генетических алгоритмах

Генетические алгоритмы – это метод решения оптимизационных задач, основанный на биологических принципах естественного отбора и эволюции. Генетический алгоритм повторяет определенное количество раз процедуру модификации популяции (набора отдельных решений), добиваясь тем самым получения новых наборов решений (новых популяций). При этом на каждом шаге из популяции выбираются «родительские особи», то есть решения, совместная модификация которых (скрещивание) и приводит к формированию новой особи в следующем поколении. Генетический алгоритм использует три вида правил, на основе которых формируется новое поколение: правила отбора, скрещивания и мутации. Мутация позволяет путем внесения изменений в новое поколение избежать попадания в локальные минимумы оптимизируемой функции.

Данные алгоритмы основаны на принципах естественного отбора Ч. Дарвина и предложены относительно недавно – в 1975 году Джоном Холландом. В них используются как аналог механизма генетического наследования, так и аналог естественного отбора. При этом сохраняется биологическая терминология в упрощенном виде и основные понятия линейной алгебры.

Генетические алгоритмы оптимизации являются алгоритмами случайно-направленного поиска и применяются в основном там, где сложно или невозможно сформулировать задачу в виде, пригодном для более быстрых алгоритмов локальной оптимизации, либо если стоит задача оптимизации недифференцируемой функции или задача многоэкстремальной глобальной оптимизации.

Типичными применениями ГА являются следующие:

- оптимизация функций;
- оптимизация запросов в базах данных;
- задачи на графах (задача коммивояжера, раскраска, нахождение паросочетаний);
- настройка и обучение искусственной нейронной сети;
- задачи компоновки;
- составление расписаний;
- игровые стратегии;
- теория приближений и др.

Основной идеей ГА является организация «борьбы за существование» и «естественный отбор» среди пробных (примерных)

решений задачи. Поскольку ГА используют биологические аналогии, то применяющаяся терминология напоминает биологическую. Как известно, эволюционная теория утверждает, что жизнь на нашей планете возникла вначале лишь в простейших ее формах – в виде одноклеточных организмов. Эти формы постепенно усложнялись, приспосабливаясь к окружающей среде и порождая новые виды, и только через многие миллионы лет появились первые животные и люди. Можно сказать, что каждый биологический вид с течением времени улучшает свои качества так, чтобы наиболее эффективно справляться с важнейшими задачами выживания, самозащиты, размножения и т. д. Таким путем возникла защитная окраска у многих рыб и насекомых, панцирь у черепахи, яд у скорпиона и многие другие полезные приспособления.

Такие алгоритмы чаще всего используются в случае, когда искомая целевая функция является разрывной, существенно нелинейной, стохастической и не имеет производных или эти производные являются недостаточно определенными. Собственно генетические алгоритмы относятся к разделу Genetic Algorithm и вызываются из командной строки с помощью gatool.

Основными параметрами ГА являются:

- вероятность мутации;
- точность получения результата;
- количество итераций алгоритма или количество поколений;
- размер популяции.

Генетический алгоритм работает согласно следующей схеме:

1) Прежде всего, в данном алгоритме для организации начала счета создается произвольное исходное семейство.

2) Далее алгоритм производит некую последовательность новых семейств или поколений. На каждом отдельном шаге алгоритм использует определенные индивидуумы из текущего поколения, для того, что бы создать последующее поколение. При формировании нового поколения в алгоритме проводятся следующие действия:

- Отмечается каждый член текущего семейства посредством вычисления соответствующего значения пригодности;
- Проводится масштабирование полученного ряда значений функции пригодности, что позволяет построить диапазон значений более удобный для последующего использования;
- Выбираются родительские значения на основе значений их пригодности;
- Часть индивидуумов из родительского поколения имеет более меньшие значения функции пригодности и которые в далее выбираются как элитные значения. Эти элитные значения передаются далее уже в последующее поколение;

- Дочерние значения образуются или путем неких случайных изменений отдельного одного родителя - мутация - или путем комбинации векторных компонентов некой пары родителей – кроссовер;
- Замена текущего семейства на дочернее с целью формирования последующего поколения.

3) Останов алгоритма производится тогда, когда выполняется какой-нибудь критерий останова.

Также основной принцип работы ГА можно описать по схеме, представленной на рис. 1.

1. Генерируем начальную популяцию из n хромосом;
2. Вычисляем для каждой хромосомы ее пригодность;
3. Выбираем пару хромосом-родителей с помощью одного из способов отбора;
4. Проводим кроссинговер двух родителей с вероятностью , производя двух потомков;
5. Проводим мутацию потомков с вероятностью ;
6. Повторяем шаги 3–5, пока не будет сгенерировано новое поколение популяции, содержащее n хромосом;
7. Повторяем шаги 2–6, пока не будет достигнут критерий окончания процесса.

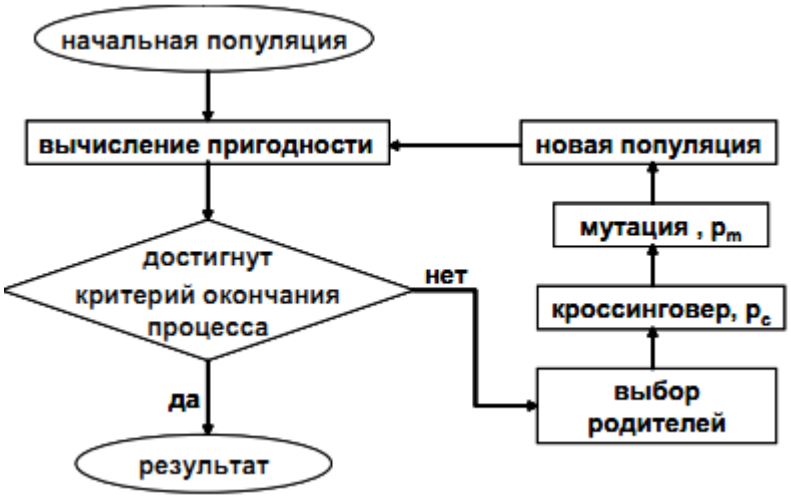

Рис. 1. Схема работы генетического алгоритма

Условия останова алгоритма:

Для определения условий останова в генетическом алгоритме используются следующие пять условий:

Generations – алгоритм останавливается тогда, когда число поколений достигает некоего заданного значения **Generations**.

Time limit – алгоритм останавливается по истечению некоего заданного времени в секундах **Time limit**.

Fitness limit – алгоритм останавливается тогда, когда значение функции пригодности для наилучшей точки для текущего семейства будет меньше или равно **Fitness limit**.

Stall generations – алгоритм останавливается в случае, если нет улучшения для целевой функции в последовательности следующих друг за другом поколений длиной **Stall generations**.

Stall time limit – алгоритм останавливается в случае, если нет улучшения для целевой функции в течение интервала времени в секундах равного **Stall time limit**.

Мутация

После процесса воспроизводства происходят мутации (mutation). Данный оператор необходим для «выбивания» популяции из локального экстремума и препятствует преждевременной сходимости. Это достигается за счет того, что изменяется случайно выбранный ген в хромосоме (рис. 2).

$$x_1 x_2 \dots x_{n-1} x_n x_{n+1} \dots x_m \longrightarrow x_1 x_2 \dots x_{n-1} \bar{x}_n x_{n+1} \dots x_m$$

Рис. 2. Мутация в точке x_n

Используемая по умолчанию функция Гауссовской мутации добавляет случайно выбранное с помощью распределения Гаусса число, **mutation**, к каждому элементу родительского вектора. Как правило, это число мутаций, которое пропорционально стандартному среднеквадратичному отклонению от распределения, уменьшается с каждой последующей итерацией. В алгоритме путем установки опций **Scale** и **Shrink** имеется возможность управлять усредненным числом мутаций для каждой итерации:

- Опция **Scale** управляет стандартным среднеквадратичным отклонением на первой итерации, которое равно параметру **Scale**, умноженному на ранг исходного семейства, задаваемого с помощью опции **Initial range**.
- Опция **Shrink** управляет скоростью уменьшения среднего числа мутаций. Стандартное среднеквадратичное отклонение линейно уменьшается до тех пор, пока его конечное значение будет равно 1 – **Shrink** от его начального значения для первого поколения. Например, если величина **Shrink** по умолчанию принимается равной 1, то тогда число мутаций уменьшается к конечному шагу до нуля.

Кроссинговер (кроссовер)

Кроссовер (скрещивание) – это операция, при которой из двух хромосом порождается одна или несколько новых хромосом. В простейшем случае кроссовер в генетическом алгоритме реализуется так же, как и в биологии. При этом хромосомы разрезаются в случайной точке и

обмениваются частями между собой. Например, если хромосомы (1, 2, 3, 4, 5) и (0, 0, 0, 0, 0) разрезать между третьим и четвертым генами и обменять их части, то получатся потомки (1, 2, 3, 0, 0) и (0, 0, 0, 4, 5).

Установка без мутаций/ установка без кроссовера

Для того, что проконтролировать работу Генетического алгоритма без включения операции мутации, следует установить параметр **Crossover fraction** из вкладки **Reproduction** в 1.0.

Для того, что проконтролировать работу Генетического алгоритма без включения операции кроссовера, следует установить параметр **Crossover fraction** в 0.

Реализация генетических алгоритмов в MatLAB

Для запуска пакета Genetic Algorithm Tool следует в командной строке MATLAB выполнить команду gatool. После этого запустится пакет генетических алгоритмов и на экране появится основное окно утилиты (рис. 3).

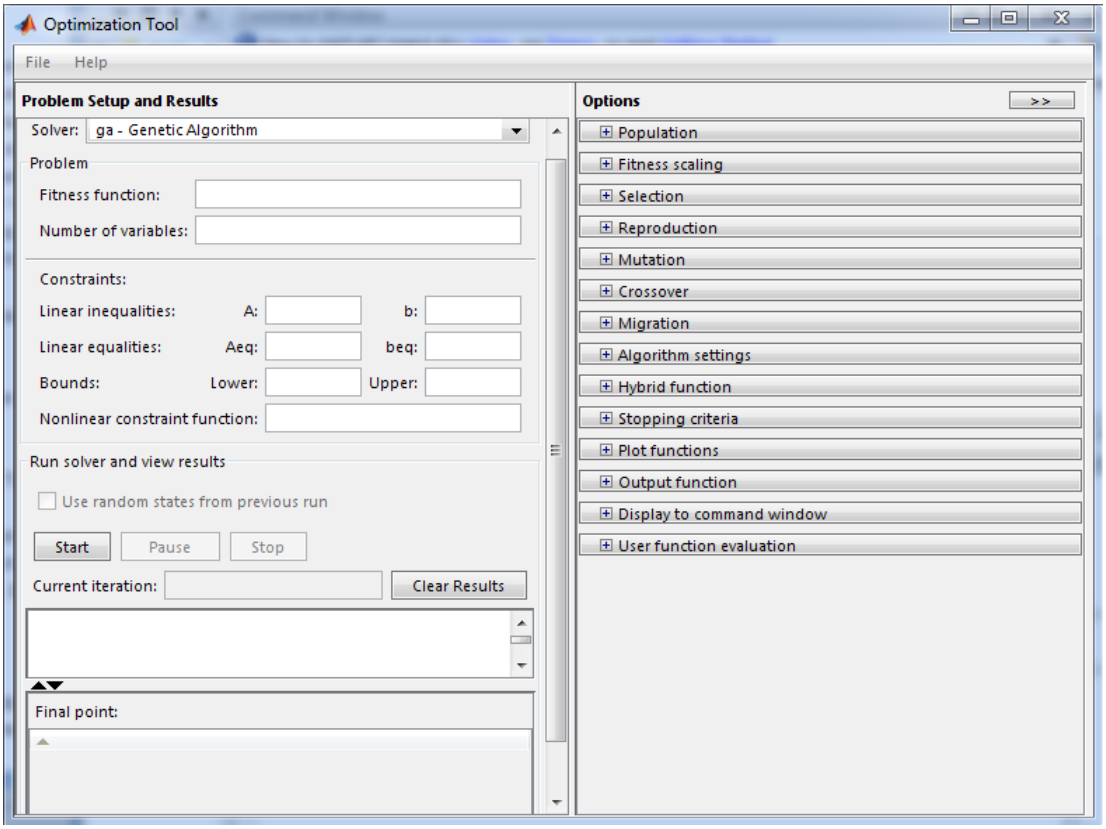

Рис. 3. Окно утилиты Genetic Algorithm Tool

Для того чтобы ввести необходимую для решения задачу следует заполнить два поля:

- Fitness function – подлежащая минимизации функция. Следует ввести указатель на М-файл функции.

- Number of variables – число независимых переменных для функции пригодности.

Пример заполнения выше описанных полей приведен на рис. 4. Функция минимизации в данном случае двумерна и имеет имя **rozenbr_fun**.

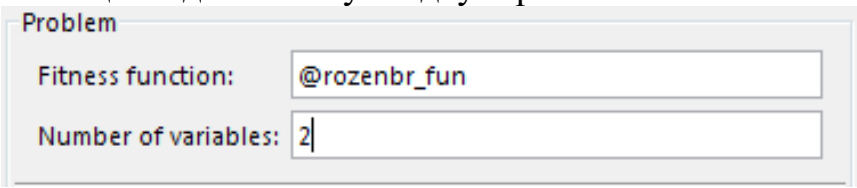

Рис. 4. Пример заполнения полей

Примечание:

М-файл с целевой функцией должен находиться в одной из папок доступа MatLAB, либо пользователь сам может внести в каталог поиска MatLAB с помощью следующей последовательности действий: Основное окно MatLAB -> File -> Set Path -> Add Folder -> Save.

Для начала работы с Генетическим алгоритмом следует кликнуть на кнопку Start, которая расположена на панели Run solver and view results.

В поле Current iteration выводится текущее число поколений. После окончания работы алгоритма на поле ниже выводится значение функции в конечной найденной точке, а на поле Final point соответственно координаты этой точки.

В правой части окна ГА расположена панель Options. Она позволяет устанавливать различные настройки для работы генетических алгоритмов. Основные вкладки, которые необходимы для работы с ГА:

- популяция (вкладка Population);
- оператор отбора (вкладка Selection);
- оператор репродукции (вкладка Reproduction);
- оператор мутации (вкладка Mutation);
- оператор скрещивание (вкладка Crossover);
- задание критерия остановки алгоритма (вкладка Stopping criteria);
- вывод различной дополнительной информации по ходу работы генетического алгоритма (вкладка Plot Functions).

Опишем возможности этих вкладок более подробно:

Во вкладке настройки популяций пользователь имеет возможность выбрать тип математических объектов, к которому будут относиться особи всех популяций (двойной вектор, битовая строка или пользовательский тип). Также вкладка популяции позволяет настраивать размер популяции (из скольких особей будет состоять каждое поколение) – поле Population size. Кроме того, в рассматриваемой вкладке имеется возможность задать вручную начальное поколение (используя пункт Initial population) или его часть, начальный рейтинг особей (пункт Initial scores), а также ввести ограничительный числовой диапазон, которому должны принадлежать особи начальной популяции (Initial range).

Вкладка Selection позволяет выбрать оператор отбора родительских особей на основе данных из функции масштабирования. В качестве доступных для выбора вариантов оператора отбора предлагаются следующие:

- Tournament – случайно выбирается указанное число особей, среди них на конкурсной основе выбираются лучшие;
- Roulette – имитируется рулетка, в которой размер каждого сегмента устанавливается в соответствии с его вероятностью;
- Uniform – родители выбираются случайным образом согласно заданному распределению и с учетом количества родительских особей и их вероятностей;
- Stochastic uniform – строится линия, в которой каждому родителю ставится в соответствие её часть определенного размера (в зависимости от вероятности родителя), затем алгоритм пробегает пот линии шагами одинаковой длины и выбирает родителей в зависимости от того, на какую часть линии попал шаг.

Вкладка Reproduction уточняет каким образом происходит создание новых особей. Пункт Elite count позволяет указать число особей, которые гарантировано перейдут в следующее поколение. Пункт Crossover fraction указывает долю особей, которые создаются путем скрещивания. Остальная доля создается путем мутации.

Во вкладке оператора мутации выбирается тип оператора мутации. Доступны следующие варианты:

- Gaussian – добавляет небольшое случайное число (согласно распределению Гаусса) ко всем компонентам каждого вектора-особи;
- Uniform – выбираются случайным образом компоненты векторов и вместо них записываются случайные числа из допустимого диапазона;
- Adaptive feasible – генерирует набор направлений в зависимости от последних наиболее удачных и неудачных поколений и с учетом налагаемых ограничений продвигается вдоль всех направлений на разную длину;
- Custom – позволяет задать собственную функцию.

Вкладка Crossover позволяет выбрать тип оператора скрещивания (одноточечное, двухточечное, эвристическое, арифметическое или рассеянное (Scattered), при котором генерируется случайный двоичный вектор соответствия родителей). Также имеется возможность задания произвольной (custom) функции скрещивания.

Во вкладке критерия остановки (Stopping criteria) указываются ситуации, при которых алгоритм совершает остановку (см. Условия останова алгоритма).

Вкладка Plot Functions, которая позволяет выбирать различную информацию, которая выводится по ходу работы алгоритма и показывает как корректность его работы, так и конкретные достигаемые алгоритмом

результаты. Наиболее важными и используемыми для отображения параметрами являются:

- Best fitness – вывод наилучшего значения оптимизируемой функции для каждого поколения;
- Best individual – вывод наилучшего представителя поколения при наилучшем оптимизационном результате в каждом из поколений;
- Distance – вывод интервала между значениями особей в поколении.

Пример поиска глобального минимума

Рассмотрим принцип работы генетических алгоритмов на примере сложной функции. Определим минимум функции Розенброка, определяемой как:

$f(x_1, x_2) = 100 * (x_2 - x_1^2)^2 + (1 - x_1)^2$.

График функции Розенброка представлен ниже на рис. 5.

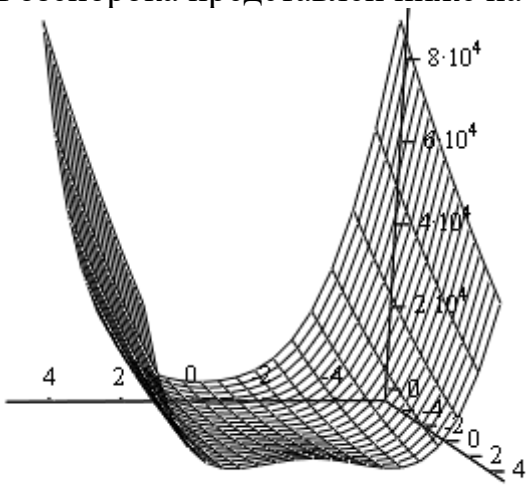

Рис. 5. График функции Розенброка

Как видно из графика, функция Розенброка содержит несколько локальных минимумов. Однако функция имеет только один глобальный минимум, который находится в точке с координатами (1;1). Значение функции в этой точке $f(1,1) = 0$.

Определим глобальный минимум с помощью одного из методов поиска многомерной оптимизации – метод Хука-Дживса. На рис. 6 приведен алгоритм поиска по методу Хука-Дживса, реализованный с помощью MathLAB.

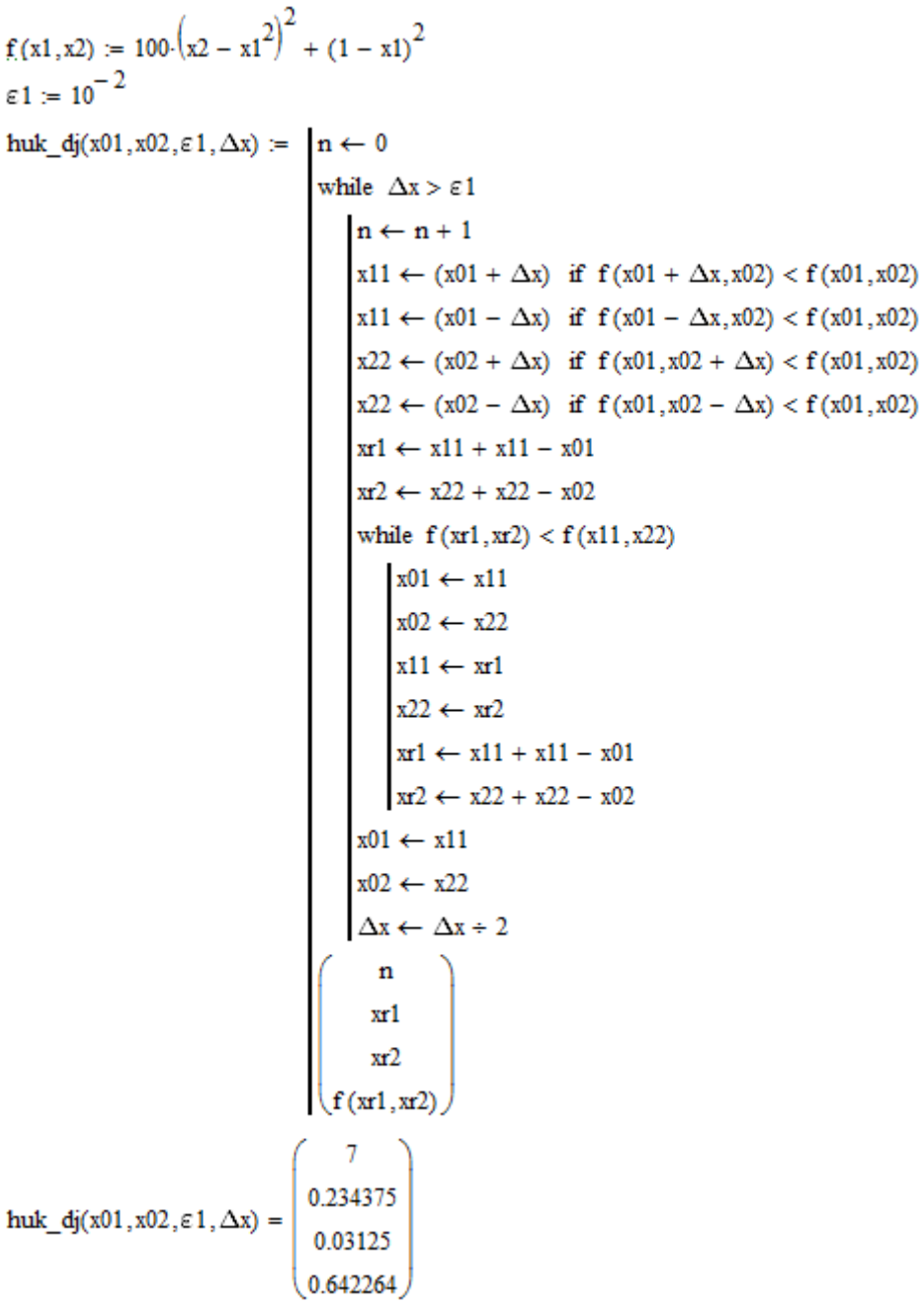

Рис. 6. Поиск минимума методом Хука-Дживса

Из результатов работы алгоритма по поиску минимума функции видно, что истинный глобальный минимум не найден, а определен лишь локальный минимум. Существуют множество других схожих методов, к примеру, градиентный метод. Все они работают очень быстро, но не гарантируют оптимальности найденного решения. Они идеальны для применения в так называемых унимодальных задачах, где целевая функция имеет единственный локальный минимум (максимум), он же является глобальным.

Таким образом, можно сделать вывод, что задача оптимизации по нахождению минимума сложной функции не всегда может быть решена с помощью стандартных методов оптимизации.

Проведем исследование этой же функции с помощью генетических алгоритмов в MatLAB.

Напишем М-файл для данной функции и сохраним его под именем **rozenbr_fun.m** (рис. 7).

```
1    function y=rozenbr_fun(x)
2 -      y=100*(x(2)-(x(1))^2)^2+(1-x(1))^2;
```

Рис. 7. М-файл с целевой функцией

В поле **fitness function** введем имя целевой функции **@rozenbr_fun**.

Установим количество особей в популяции = 10, в разделе plots установим флажки для **best fitness, best individual, distance**. Щелкнем по кнопке **start.**

Для данной задачи результаты получились следующие - минимум функции достигается в точке с координатами $x_1 = 0{,}901;\ x_2 = 0{,}806.$ Значение функции Розенброка в этой точке $f(x_1, x_2) = 0{,}014071620305311536$.

Графическое решение приведено на рис. 8.

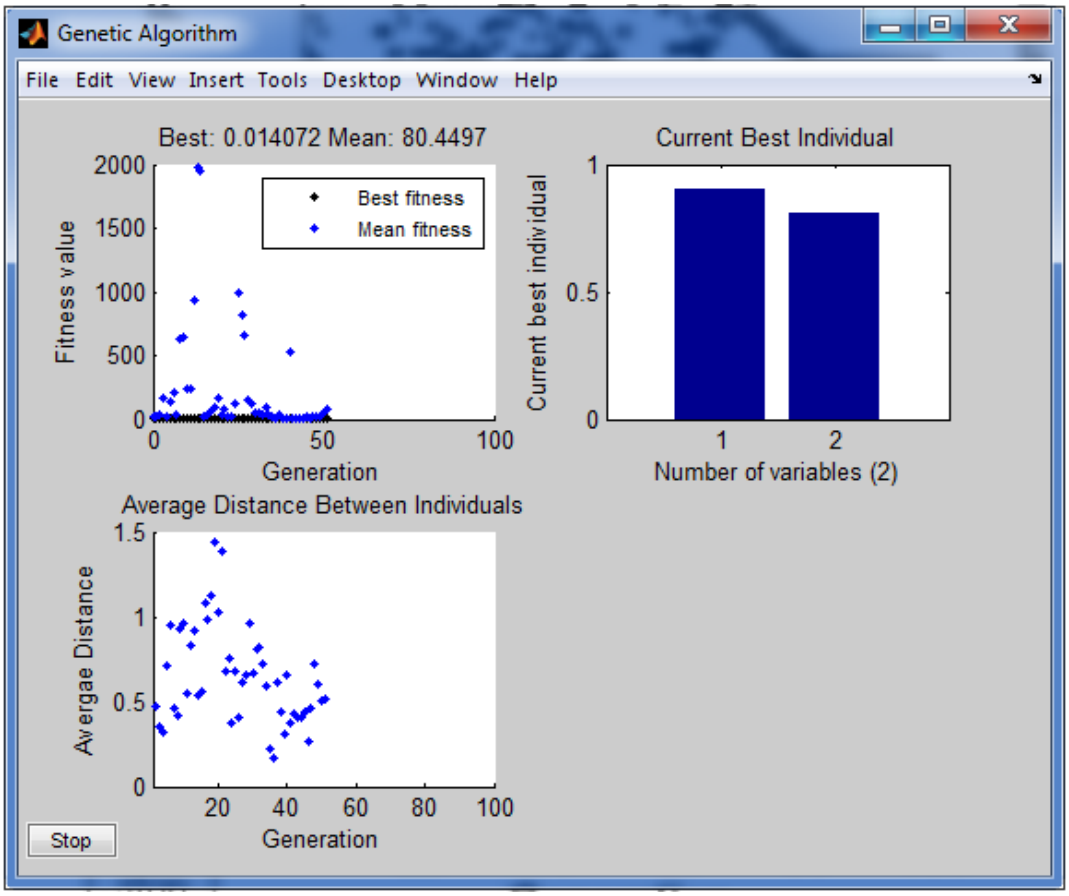

Рис. 8. Графическое решение задачи

Первый график (Fitness Value) отображает изменение значение целевой функции. Видно, что, начиная с 40 популяции, алгоритм сошелся к решению. На втором графике (Current best individual) изображена наилучшая особь. Третий график (Average Distance) соответствует изменению расстояния между особями в поколениях.

Конечная точка расчетов близка к точке истинного минимума (1, 1). Исследуем функцию при различных исходных данных, результаты приведены в табл. 1. Количество итераций алгоритма или количество поколений является остановом алгоритма, его значение оставим по умолчанию равным 100 поколениям. Тип оператора мутации выберем Гауссов с вероятностью 1%.

Таблица 1. Результаты эксперимента по нахождению минимума

		Размер начальной популяции				
		10	50	100	500	1000
Вероятность кроссовера	1	$x_1 = 0{,}847$ $x_2 = 0{,}657$	$x_1 = 0{,}838$ $x_2 = 0{,}7$	$x_1 = 0{,}896$ $x_2 = 0{,}807$	$x_1 = 0{,}963$ $x_2 = 0{,}925$	$x_1 = 0{,}977$ $x_2 = 0{,}956$
	0,5	$x_1 = 0{,}894$ $x_2 = 0{,}798$	$x_1 = 1{,}069$ $x_2 = 1{,}151$	$x_1 = 1{,}111$ $x_2 = 1{,}234$	$x_1 = 0{,}991$ $x_2 = 0{,}981$	$x_1 = 1{,}012$ $x_2 = 1{,}023$
	0	$x_1 = 1{,}048$ $x_2 = 1{,}117$	$x_1 = 0{,}9$ $x_2 = 0{,}803$	$x_1 = 0{,}949$ $x_2 = 0{,}894$	$x_1 = 0{,}968$ $x_2 = 0{,}936$	$x_1 = 1{,}018$ $x_2 = 1{,}035$

По результатам проведенных экспериментов видно, что однозначную зависимость полученных координат от вероятности влияния скрещивания и мутации от величины популяции выявить сложно, так как исходная популяция создается случайным образом. Но в любом случае видно, что при отсутствии мутации значения координат более искажены и с ростом популяции значения координат становятся более близкими к истинным значениям.

Программа работы и методические указания

1) Найти минимум функции одной переменной любым известным способом. Исследовать функцию с помощью генетических алгоритмов. Сравнить полученные результаты. Определить глобальный минимум и значение функции в этой точке.

Провести эксперимент при различном размере начальной популяции: 10; 50; 300; 800. Для каждого из этих значений принять следующие операторы отбора родительских особей:

- Stochastic uniform;
- Uniform;
- Roulette.

Сделать вывод о том, как влияет размер исходной популяции на результаты, а также какое влияние вносят различные операторы отбора родительских особей. Привести графическое решение, найденное с

помощью генетических алгоритмов, которое соответствует максимально точному найденному решению. Объяснить каким образом влияют операторы отбора родительских особей на результаты.

Целевые функции одной переменной приведены в табл. 2.

Таблица 2. Функции одной переменной

Вариант	Целевая функция
1	$f(x) = \sin(\mathbf{x}) + \sin(\mathbf{x^2})$
2	$f(x) = \frac{6 * \sqrt[3]{6(x-3)^2}}{(x-1)^2 + 8}$
3	$f(x) = -2x - 8 + 3 * \sqrt[3]{6(x+4)^2}$
4	$f(x) = \sqrt{3 + \mathbf{x^2}} + 3 * \cos(x)$

2) Найти минимум функции двух переменных любым известным способом. Провести исследование этой функции с помощью генетических алгоритмов и заполнить табл. 3, изменяя вероятность скрещивания и размер начальной популяции. Выбрать оператор мутации по Гауссу, вероятность мутации принять равным по умолчанию 1,0.

Таблица 3. Результаты экспериментов

		Размер начальной популяции				
		10	50	100	500	1000
Вероятность кроссовера	1	$x_1 =$ $x_2 =$ $f(x_1, x_2) =$				
	0,8					
	0,6					
	0,3					
	0					

По результатам проведенных экспериментов сделать вывод о том, при каком соотношении вероятности скрещивания к мутации результаты максимально точны. Привести графическое решение, найденное с помощью генетических алгоритмов, которое соответствует максимально точному найденному решению.

Определить относительную погрешность полученных значений функций при каждом значении размера начальной популяции (для наилучшего соотношения вероятности скрещивания к мутации) и построить график зависимости погрешностей от размера начальной популяции. Сделать вывод о влиянии размера начальной популяции на результаты.

Целевые функции двух переменных приведены в табл. 4.

Таблица 4. Функции двух переменных

№ вар.	Целевая функция	Значение функции в точке глобального минимума
1	$f(x_1, x_2) = \left(x_2 - x_1^2\frac{5{,}1}{4\pi^2} + x_1\frac{5}{\pi} - 6\right)^2 + 10\left(1 - \frac{1}{8\pi}\right)\cos(x_1) + 10$	$f(x_{1min}, x_{2min}) = 0{,}397887$
2	$f(x_1, x_2) = 20 + e - 20 * \exp\left(-0{,}2\sqrt{\frac{1}{2}(x_1^2 + x_2^2)}\right) - \exp\left(\frac{1}{2(\cos(2\pi x_1) + \cos(2\pi x_2))}\right)$	$f(x_{1min}, x_{2min}) = 0$
3	$f(x) = \sum_{i=1}^{n} \lvert x_i \rvert^{i+1}, \quad i = 1{:}n, \quad n = 2$	$f(x_{\min}) = 0$
4	$f(x) = 10n + \sum_{i=1}^{n}(x_i^2 - 10 * \cos(2\pi x_i)), \quad i = 1{:}n, \quad n = 2$	$f(x_{\min}) = 0$

Содержание отчета

- цель работы;
- задание;
- краткое описание действий по пунктам;
- графики;
- вывод по каждой части задания.

Литература

1. Гладков Л.А., Курейчик В.В., Курейчик В.М. Генетические алгоритмы. 2-е изд., – М.: ФИЗМАТЛИТ, 2006. – 320 с.
2. Борисов В.В., Круглов В.В., Федулов А.С. Нечеткие модели и сети. – М.: Горячая линия - Телеком, 2007. – 284 с.
3. Конышева Л.К., Назаров Д.М. Основы теории нечетких множеств. Учебное пособие. Стандарт третьего поколения. – СПб. Питер, 2011. – 192 с.

Контрольные вопросы

1. Каковы механизм передачи наследственной информации?
2. Дайте определение генетического алгоритма (ГА).
3. Перечислите основные отличительные особенности ГА.
4. Перечислите генетические операторы.
5. Какие критерии останова используются для ГА?
6. Опишите схему классического ГА.
7. В чем заключаются особенности совместного использования генетических операторов?
8. Сформулируйте фундаментальную теорему ГА.