

НУЛП, САПР, СПК		Тема	Оцінка:	Підпис:
КНМ-14	2	Методи еволюційного пошуку		
Ласка Г. Л.				
№11341784				
Методи нечіткої логіки та еволюційні алгоритми			Викладач: Кривий Р. З.	

Мета: Ознайомитися з принципом роботи методів еволюційного пошуку, та їх реалізації в MatLab.

Теоретичні відомості

Генетичні алгоритми - один із методів розв'язання оптимізаційних задач, що базується на природному відборі (процес що нагадує біологічну еволюцію). Генетичний алгоритм повторює знову і знову модифікацію популяції окремих розв'язків (особин). На кожному кроці алгоритм випадковим чином вибирає із популяції особин, що стануть батьками і будуть використовуватися при створенні нащадків для формування наступного її покоління. Через послідовність поколінь, популяція розвивається в напрямку оптимального розв'язку. Генетичні алгоритми використовують на кожній ітерації три основні типи дій для створення наступного покоління з поточної популяції:

Селекція (Selection) - вибір особин, що називаються батьками і які беруть участь у створенні наступного покоління.

Кросовер (Crossover) - комбінація двох батьків для формування нащадків.

Мутація (Mutation) - внесення випадкових змін до батьківських особин, щоб сформувати нових нащадків.

Також важливими є поняття:

- Функція пристосованості (Fitness function) - функція, яку необхідно оптимізувати (у Matlab здійснюється пошук її мінімуму).
- Особина (Individual) - значення, для яких можна обчислити функцію пристосованості. Значення цієї функції для певної особини і буде величиною її пристосованості.
- Популяція (Population) - набір особин. Наприклад, якщо розмір популяції 100 і у функції пристосованості є три параметри, то популяцію можна зобразити у вигляді матриці 100 x 3. Популяція може містити однакові особини.
- Покоління (Generation) - популяція, яка формується на кожному черговому кроці роботи генетичного алгоритму.
- Пристосованість (Fitness value) - значення функції пристосованості для конкретної особини. Чим менше це значення, тим краще.

Словесний опис генетичного алгоритму:

1) Створення початкової популяції (Initial population)

2) Використання особин поточної популяції для створення нового покоління. Для цього виконуються такі дії:

- Обчислюється пристосованість кожної особини поточної популяції
- Особини сортуються по значеннях своєї пристосованості
- Окремі особини, пристосованість яких має суттєво менше значення ніж у інших, називаються елітою і вони переходять у наступне покоління
- На основі своїх пристосованостей вибираються особини, які називаються батьками
- За допомогою батьківських особин створюються нащадки. Нашадки також можуть появитися з одної батьківської особини завдяки мутації
- Поточна популяція замінюється на нову, сформовану з нащадків

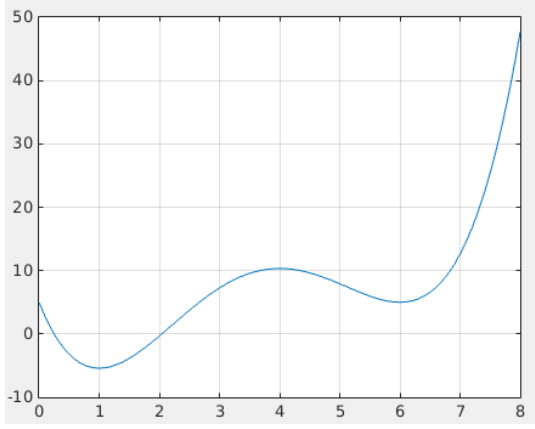
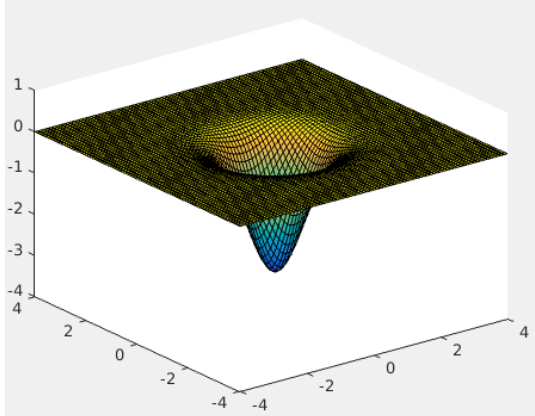
3) Якщо виконується одна із умов зупинки алгоритму (описано далі), то робота завершується. Інакше повторюється крок 2.

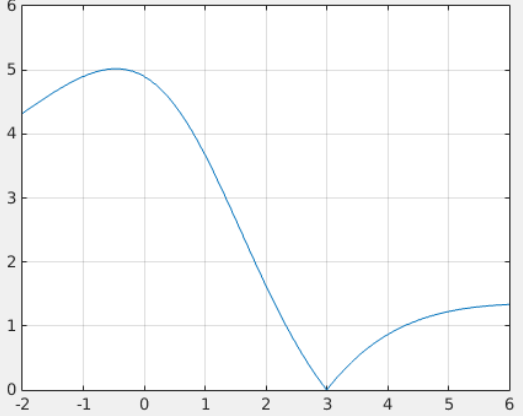
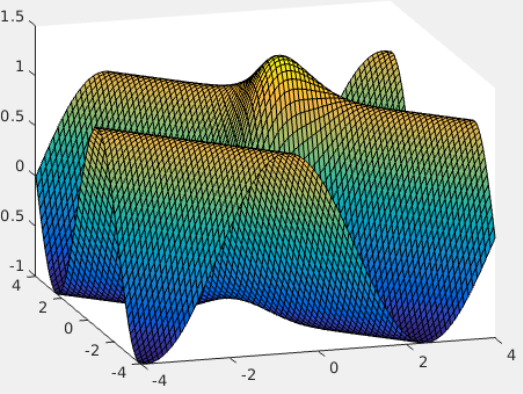
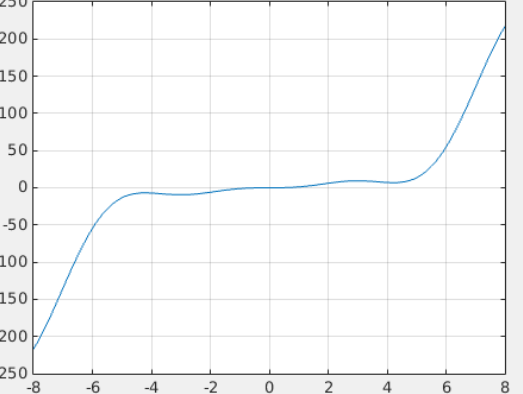
Завдання

Варіант 3

№ задачі	Еволюційні оператори		
	Відбір	Схрещування	Мутація
1	турнірний	одноточкове	гаусівська
2	ранжування	діагональне	нерівномірна

Обрані функції:

№	Функція	Графік	Ціль
1	$f(x) = 5 - 24x + 17x^2 - \frac{11}{3}x^3 + \frac{1}{4}x^4$		min
2	$f(x, y) = (y - 3) \exp(-x^2 - y^2)$		min

3	$f(x) = \frac{6\sqrt[3]{6(x-3)^2}}{(x-1)^2+8}$		max
4	$f(x, y) = \exp(-x^2 - y^2) + \sin(x + y)$		max
5	$f(x) = 0.3x^3 + x^2 \cdot \sin(x)$		min

Порівняння генетичних алгоритмів здійснюється при сталих значеннях діапазону вибору початкової популяції, розміру популяції та максимальної кількості ітерацій. Результати порівнюються за часом виконання алгоритму та точністю отриманих значень. Оскільки генетичні алгоритми мають випадковий характер, то за еталонне значення приймається мінімальне з отриманих після декількох застосувань алгоритму, а за результат — середнє арифметичне. Чим більшою буде похибка, тим ймовірніше, що повторне застосування обраного алгоритму не дасть бажаного результату.

Функція з налаштуваннями ga згідно Залачі 1

```
function [ time, fval ] = ga_var1( func, nvars, range, population_size,
generations )
% Задача 1 %
options = gaoptimset();
options = gaoptimset(options, 'SelectionFcn', @selectiontournament);
options = gaoptimset(options, 'CrossoverFcn', @crossoversinglepoint);
options = gaoptimset(options, 'MutationFcn', @mutationgaussian);
%options = gaoptimset(options, 'PopInitRange', range);
```

```

options = gaoptimset(options, 'Generations', generations);
options = gaoptimset(options, 'PopulationSize', population_size);

tic
[fx, fval] = ga(func, nvars, options);
time = toc;

end

```

Функція з налаштуваннями ga згідно Залачі 2

```

function [ time, fval ] = ga_var2( func, nvars, range, population_size,
generations )
% Задача 2 %
options = gaoptimset();
options = gaoptimset(options, 'SelectionFcn', @selectionroulette);
options = gaoptimset(options, 'CrossoverFcn', @crossoverintermediate);
options = gaoptimset(options, 'MutationFcn', @mutationuniform);
options = gaoptimset(options, 'PopInitRange', range);
options = gaoptimset(options, 'Generations', generations);
options = gaoptimset(options, 'PopulationSize', population_size);

tic
[x, fval] = ga(func, nvars, options);
time = toc;

end

```

Повертає статистичні дані для вказаної функції

```

function [ ret_fun ] = ga_static( func, nvars, k)

RANGE = [-50; 50];
POPULATION = 20;
GENERATIONS = 100;
F(1) = {@ga_var1};
F(2) = {@ga_var2};
N = 10;

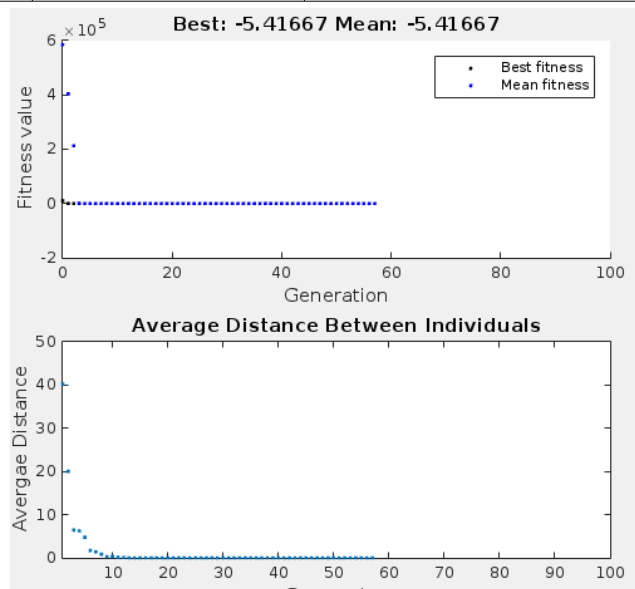
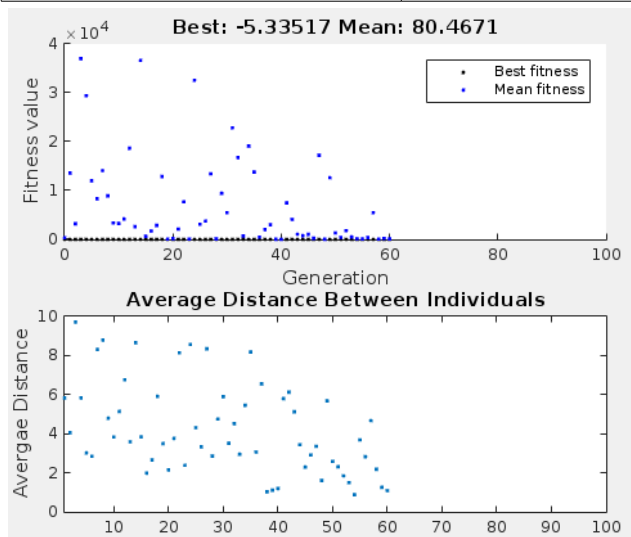
for var = 1:1:2
    time = []; fval = [];
    ga_var = F{var};
    for i = 1:1:N
        [ time_ret, fval_ret ] = ga_var( func, nvars, RANGE, POPULATION,
GENERATIONS);
        time(i) = time_ret;
        fval(i) = fval_ret;
    end;

    ret_fun(1, var) = roundn(mean(time), -4);
    ret_fun(2, var) = k*roundn(mean(fval), -4);
    ret_fun(3, var) = k*roundn(min(fval), -4);
    ret_fun(4, var) = abs((ret_fun(2, var) - ret_fun(3, var))/...
        ret_fun(3, var))*100;
end;

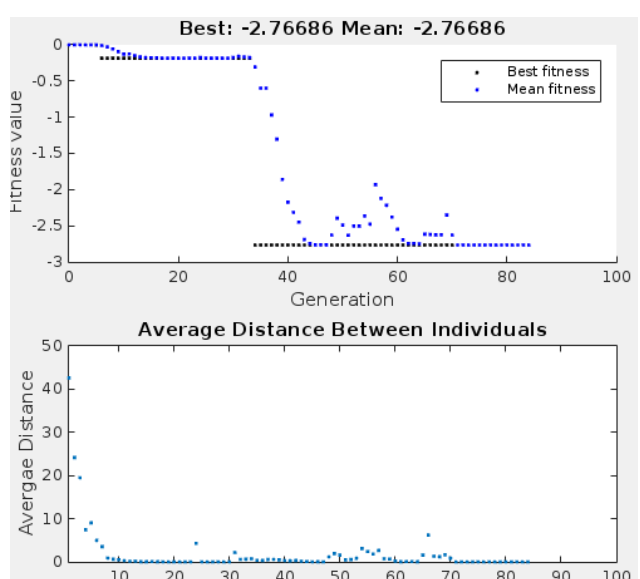
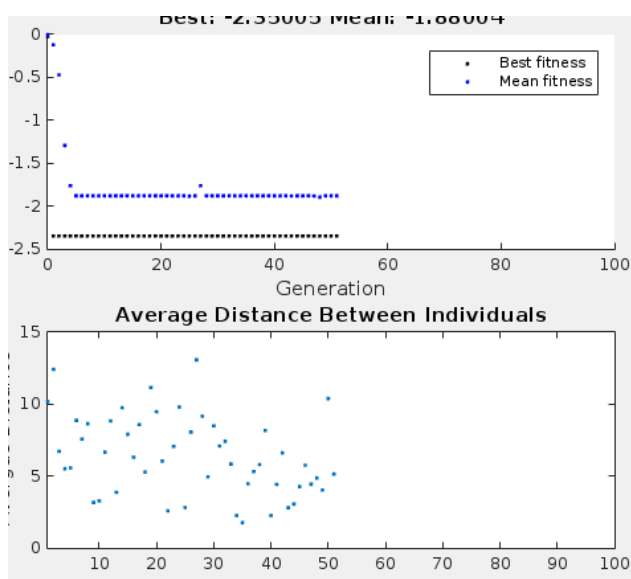
end

```

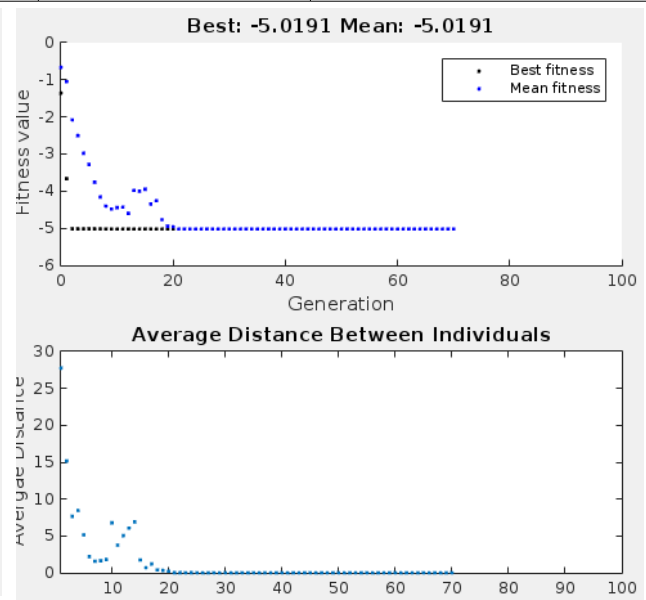
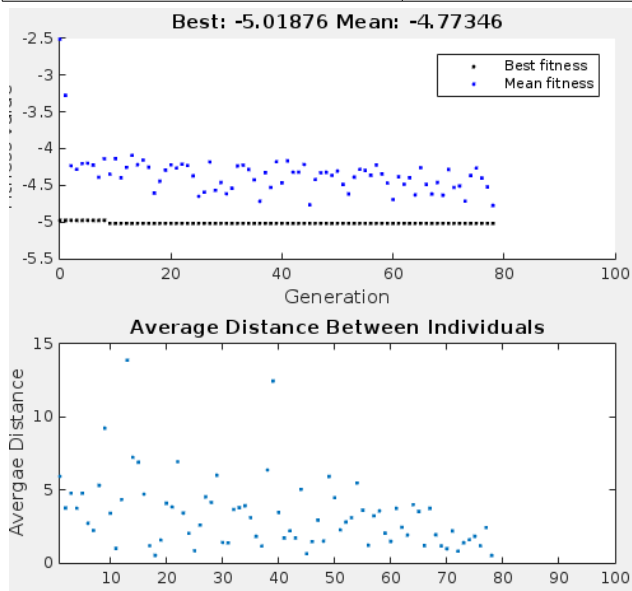
Функція №1 (min)			
	Задача 1	Задача 2	Реальне значення
Час виконання	0,3234	0,1146	
Середнє значення	-5,4086	-5,4114	-5,4167
Мінімальне значення	-5,4166	-5,4167	
Відносна похибка (%)	0,1477	0,0978	



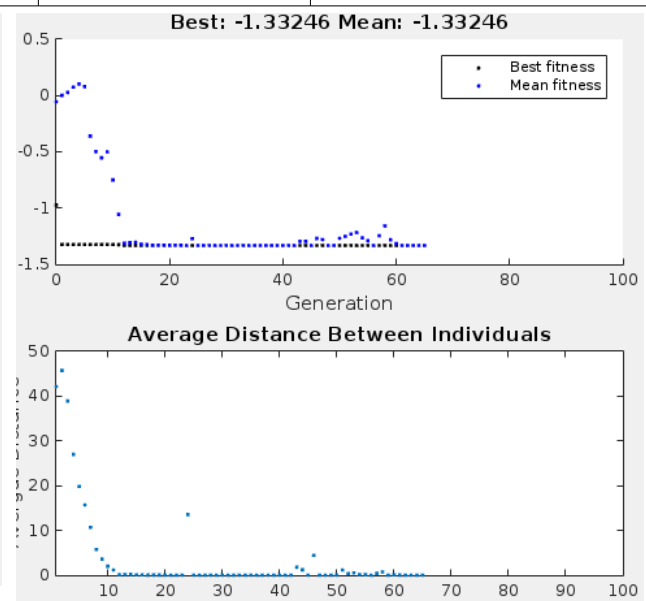
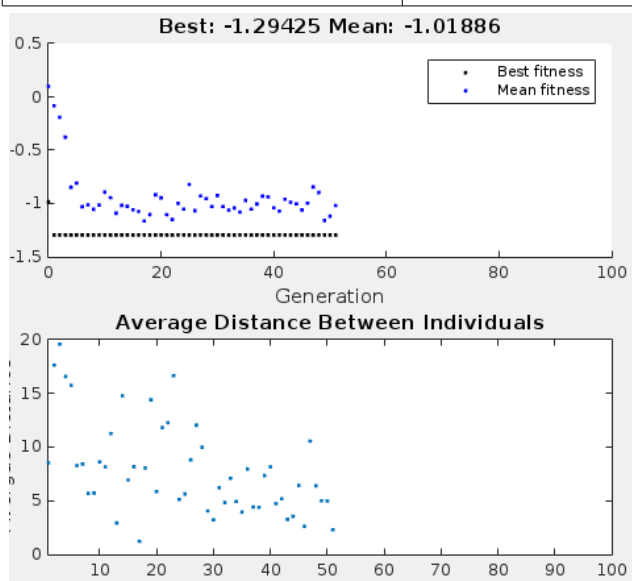
Функція №2 (min)			
	Задача 1	Задача 2	Реальне значення
Час виконання	0,3153	0,1684	
Середнє значення	-2,0262	-2,2284	-3,0801
Мінімальне значення	-3,0722	-3,0800	
Відносна похибка (%)	34,0473	27,6494	



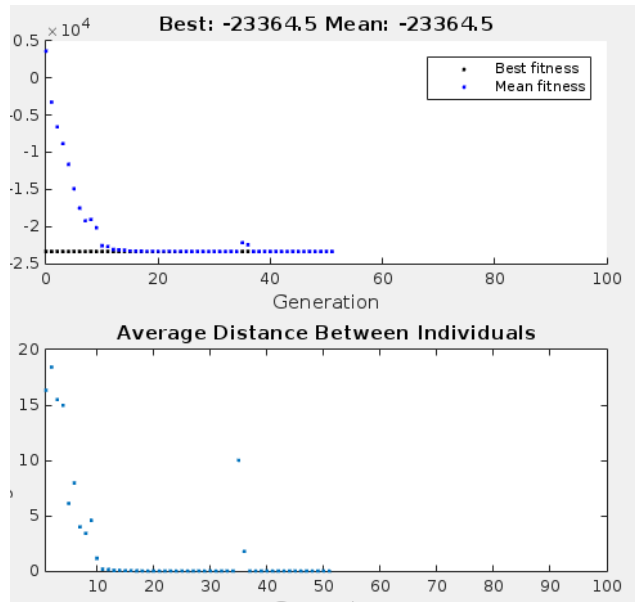
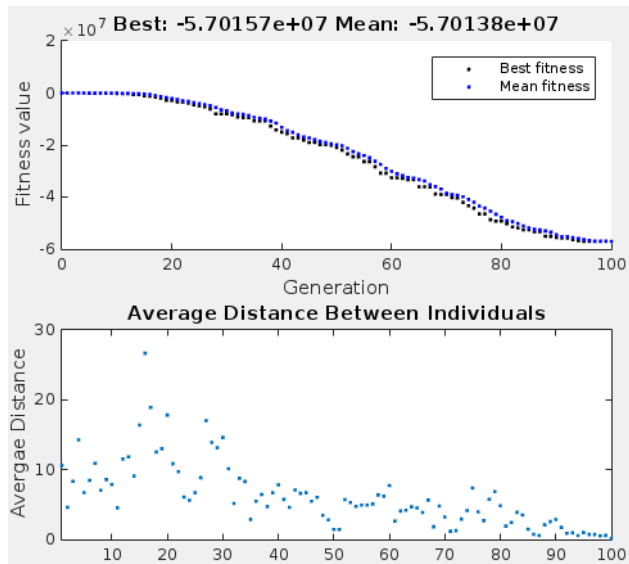
Функція №3 (max)			
	Задача 1	Задача 2	Реальне значення
Час виконання	0,2932	0,1142	
Середнє значення	5,0181	5,0182	5,0191
Мінімальне значення	5,0191	5,0191	
Відносна похибка (%)	0,0199	0,0179	



Функція №4 (max)			
	Задача 1	Задача 2	Реальне значення
Час виконання	0,2812	0,1375	
Середнє значення	1,0905	1,0905	1,4506
Мінімальне значення	1,3304	1,4506	
Відносна похибка (%)	18,0322	24,8242	



Функція №5 (min)			
	Задача 1	Задача 2	Реальне значення
Час виконання	0,2812	0,1135	
Середнє значення	-8,27E+07	-2,72E+04	---
Мінімальне значення	-1,13E+08	-3,26E+04	
Відносна похибка (%)	26,9582	16,4903	



Висновок

В MatLab вже реалізовані методи для використання генетичних алгоритмів під час пошуку оптимумів функції. Нам залишається тільки викликати їх з правильними параметрами і проаналізувати результат.

Початкова вибірка вибиралася із широкого діапазону значень, щоб віддалити її від оптимуму. Оптимум досягається швидше, якщо використовувати параметри із задачі 2, при цьому як мінімальне, так і середнє значення ближчі до реального оптимуму.