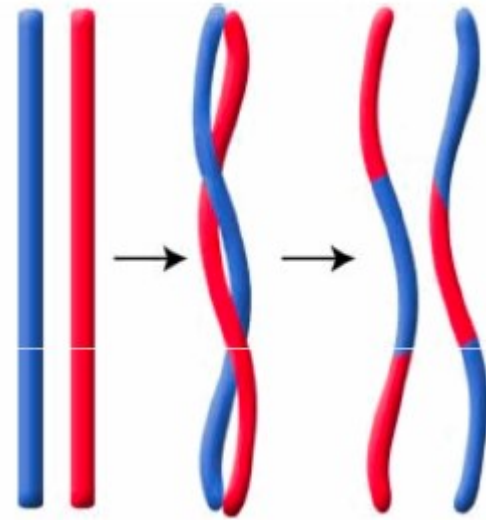


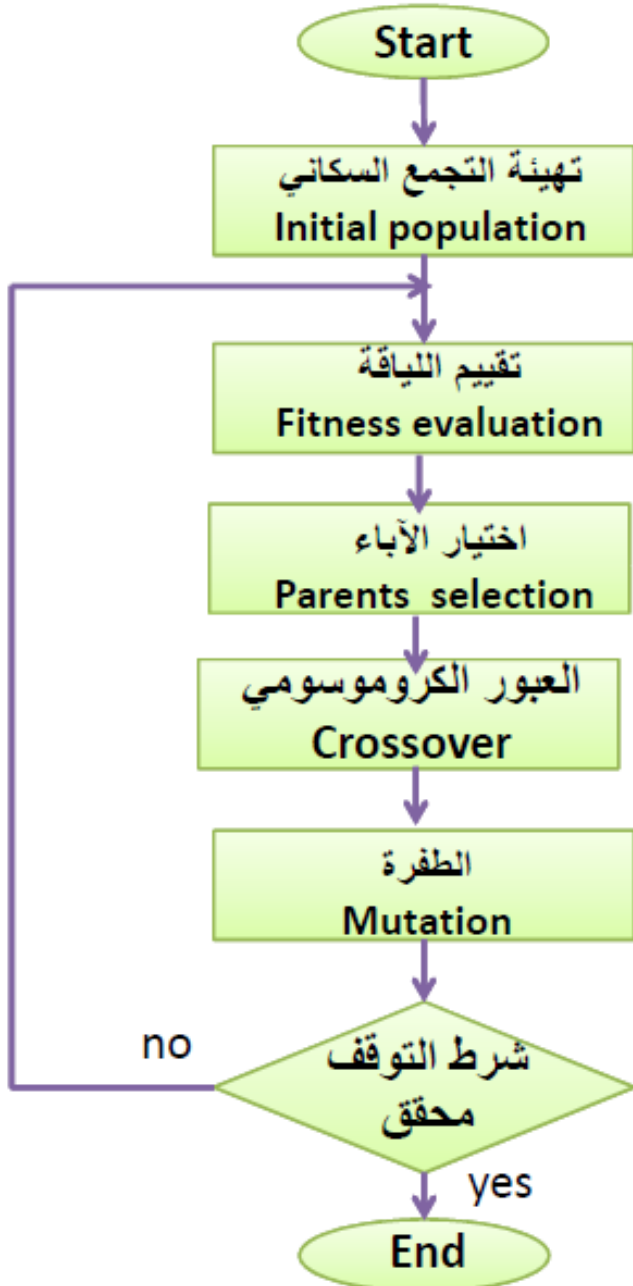
المتحركات العصبونية والعائمة
المحاضرة /9/ - عملي



مقدمة عن الخوارزميات الجينية:

- الخوارزميات الجينية هي نماذج حسابية تعتمد على علم الوراثة والتطور عند الأحياء.
- صفات كل كائن تكون مشفرة في جيناته (مورثاته)، سلسلة هذه الجينات تسمى كروموسوم (صبغي) Chromosome .
- يمكن تطبيق الخوارزميات الجينية لحل مسائل الأمثلة (إيجاد الحل الأمثل) Optimization.
- كمثال بسيط، لنفرض أننا نريد الحصول على القيمة العظمى للتابع، $y = 5 - (x - 3)^2$ ما هي قيمة x المناسبة؟
 - يمثل الرقم $x=1$ حلاً للمشكلة ، أيضاً $x=0$ يمثل حلاً، لكن الحل الأمثل بالتأكيد هو $x=3$.
- كل كروموسوم هو حل ممكن للمشكلة المطروحة.

المخطط الصندوقي لعمل الخوارزمية الجينية



1- توليد عدد كبير من الحلول الممكنة لمسألة أو مشكلة ما.

2- تقييم لياقة كل حل من هذه الحلول.

3- انتقاء الآباء أي الحلول ذات القيمة الأفضل (التطور الدارويني Darwinian Evolution).

4- إجراء عملية العبور الكروموسومي (التزاوج).

5- الطفرة إن وجدت.

ثم إعادة الخطوات من 2 إلى 5 حتى الوصول إلى الحل الأمثل.

الخوارزميات الجينية – أوامر ماتلاب:

- $[x, fval] = ga(@fitnessfun, nvars, options)$

- حيث:
- $@fitnessfun$ يشير إلى تابع اللياقة.
- $nvars$ هو عدد المتحولات المستقلة في تابع اللياقة.
- $options$ بنية تحوي خصائص الخوارزمية الجينية.
- x النقطة التي يصل عندها التابع إلى قيمته النهائية.
- $fval$ القيمة النهائية لتابع اللياقة. $optimtool('ga')$

• على نافذة الأوامر نكتب:

• `optimtool('ga')`

• يجب إدخال المعلومات:

• **Fitness function**

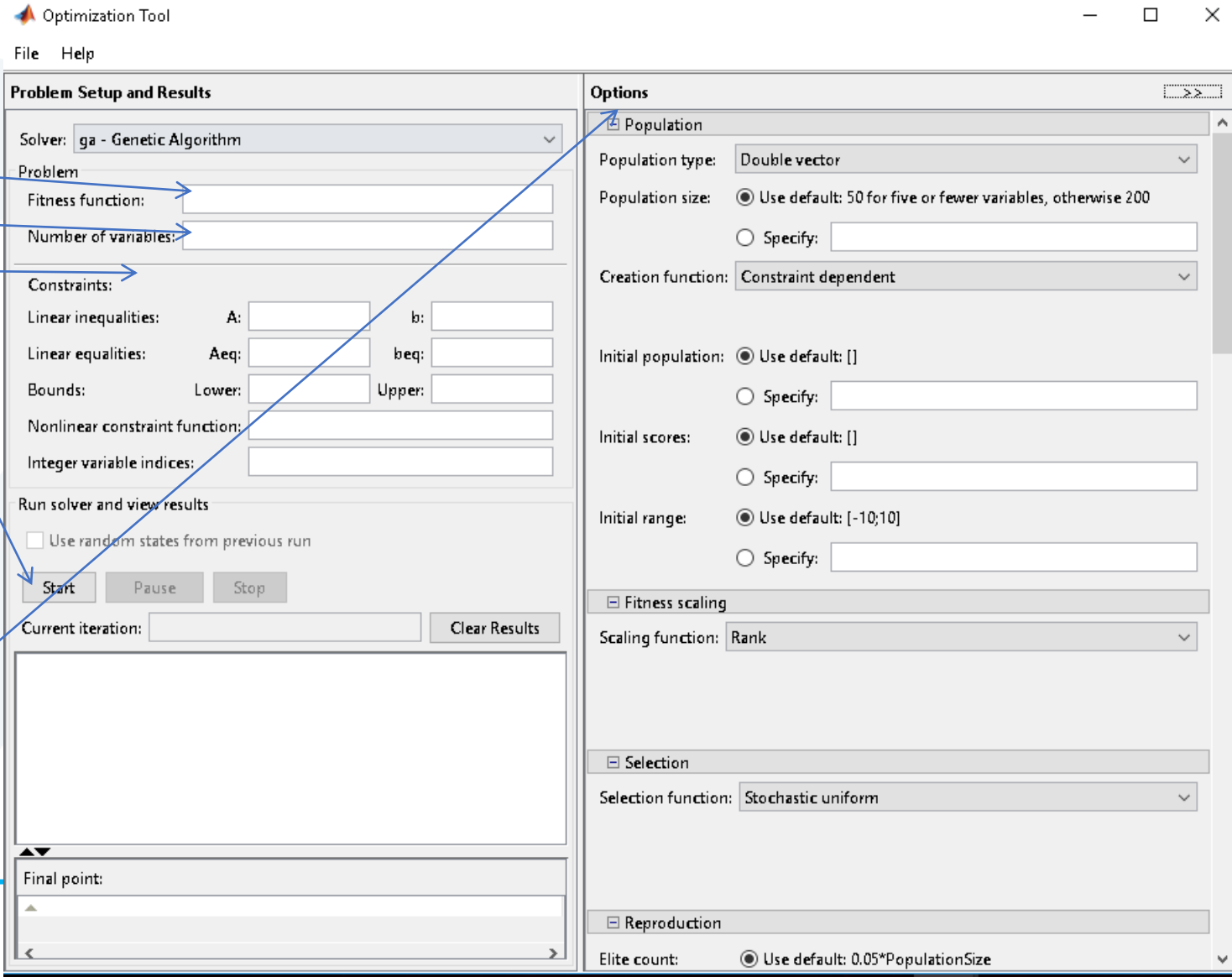
• **Number of variables**

• يمكن إدخال القيود الخطية وغير الخطية في لوحة القيود.

• التشغيل يتم بضغط Start وتظهر النتائج في لوحة التشغيل والعرض.

• تغيير الخيارات من اللوحة Options.

• لرسم النتائج من لوحة plot أسفل الخيارات Options.



The screenshot shows the Optimization Tool interface with the following settings:

- Problem Setup and Results:**
 - Solver: ga - Genetic Algorithm
 - Problem: (empty)
 - Fitness function: (empty)
 - Number of variables: (empty)
 - Constraints:
 - Linear inequalities: A: (empty), b: (empty)
 - Linear equalities: Aeq: (empty), beq: (empty)
 - Bounds: Lower: (empty), Upper: (empty)
 - Nonlinear constraint function: (empty)
 - Integer variable indices: (empty)
 - Run solver and view results:
 - Use random states from previous run
 - Start, Pause, Stop buttons
 - Current iteration: (empty), Clear Results button
 - Final point: (empty)
- Options:**
 - Population:
 - Population type: Double vector
 - Population size: Use default: 50 for five or fewer variables, otherwise 200
 - Specify: (empty)
 - Creation function: Constraint dependent
 - Initial population: Use default: []
 - Specify: (empty)
 - Initial scores: Use default: []
 - Specify: (empty)
 - Initial range: Use default: [-10;10]
 - Specify: (empty)
 - Fitness scaling:
 - Scaling function: Rank
 - Selection:
 - Selection function: Stochastic uniform
 - Reproduction:
 - Elite count: Use default: 0.05*PopulationSize



جامعة
المنارة
MANARA UNIVERSITY

المثال 1 - إيجاد القيمة الصغرى لتابع Rastrigin:

$$Ras(x) = 20 + x_1^2 + x_2^2 - 10(\cos 2\pi x_1 + \cos 2\pi x_2)$$

Fitness function:

@rastriginsfcn

Number of variables:

2

Run solver

Use random states from previous run

Start

Pause

Stop

Current generation:

Status and results:

Clear Status

```
GA running.  
GA terminated.  
Fitness function value: 0.5461846729884883  
Optimization terminated: average change in the
```

Final point:

1	2
0.00218	0.05266

[x fval exitflag] = ga(@rastriginsfcn, 2)

المثال الثاني - إيجاد القيمة الصغرى لتابع:

- $f(x) = 100(x_1^2 - x_2)^2 + (1 - x_1)^2$

- ننشئ تابع جديد في ماتلاب ونكتب تابع اللياقة فيه:

- `function y = simple_fitness(x)`

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

- ننشئ ملف جديد نكتب ضمنه التعليمات التالية:

- `FitnessFunction = @simple_fitness;`

- `numberOfVariables = 2;`

- `[x,fval] = ga(FitnessFunction,numberOfVariables)`

المثال الثالث – إضافة القيود:

• نفترض نفس تابع اللياقة في المثال السابق مع إضافة القيود التالية:

- $x_1 \cdot x_2 + x_1 - x_2 + 1.5 \leq 0$ (nonlinear constraint)
- $10 - x_1 \cdot x_2 \leq 0$ (nonlinear constraint)
- $0 \leq x_1 \leq 1$ (bound)
- $0 \leq x_2 \leq 13$ (bound)

• ننشئ تابع جديد في ماتلاب ونكتب القيود فيه:

- ```
function [c, ceq] = simple_constraint(x)
c = [1.5 + x(1)*x(2) + x(1) - x(2);
 -x(1)*x(2) + 10];
ceq = [];
```



## المثال الثالث – تنمة:

- ثم نكتب التعليمات التالية:

- ObjectiveFunction = @simple\_fitness;  
nvars = 2;           % Number of variables  
LB = [0,0];         % Lower bound  
UB = [1,13];        % Upper bound  
ConstraintFunction = @simple\_constraint;  
[x,fval] = ga(ObjectiveFunction,nvars,[],[],[],[],LB,UB, ...  
ConstraintFunction)

## المثال الثالث – إضافة الطفرة:

- عند الحاجة لإضافة طفرة نضيف التعليمة التالية:
- `options = gaoptimset('MutationFcn',@mutationadaptfeasible);`
- ونعدل التعليمة الأساسية للخوارزمية الجينية لتصبح:
- `[x,fval] = ga(ObjectiveFunction,nvars,[],[],[],[],LB,UB, ...  
ConstraintFunction,options)`

## المثال الثالث – رسم النتائج:

- لرسم النتائج نضيف التعليمة التالية:
- `options = gaoptimset('PlotFcns',{@gaplotbestf,@gaplotstopping});`
- ونعدل التعليمة الأساسية للخوارزمية الجينية لتصبح:
- `[x,fval] = ga(ObjectiveFunction,nvars,[],[],[],[],LB,UB, ...  
ConstraintFunction,options)`

## تعديل بعض خصائص الخوارزميات الجينية:

- تحديد عدد التجمع السكاني:

```
options = gaoptimset(options,'PopulationSize',10);
```

- تحديد مجال التجمع السكاني:

```
options = gaoptimset(options,'PopInitRange',[-1 0;1 2]);
```

- تعديل معيار التوقف:

```
options = gaoptimset(options,'Generations',150,'StallGenLimit', 100);
```

- في النهاية ننفذ تعليمة الخوارزمية الجينية مع إضافة الخيارات:

```
[x,Fval,exitFlag,Output] = ga(FitnessFunction,numberOfVariables,[],[],[],...
[],[],[],[],options);
```



GOOD LUCK ..