智慧型系統概論HW2

系級：通訊4A 姓名：鄭筱筠 學號：01031012

1. **Object**

Find 𝑥 such that 𝑓(𝑥) is maximum. Try Roulette Wheel Selection(RWS) and Tournament Selection(TS).

f(𝑥) = ， −10 ≤𝑥 ≤ 10

(1) BGA (10 bits)

(2) RGA

(3) EA

Population size：10 , Crossover rate：0.8 , Mutation rate：0.01

1. **Procedure**
   1. **Method**

本次模擬使用基因演算法進行最佳化，涵蓋了三種演算法模式：二進位基因演算法、實數基因演算法與演化演算法。在選擇機制上，分別實作了輪盤式選擇與競爭式選擇兩種方式，以提升族群中優秀個體的生存機率。交配方法方面，模擬包含單點交配、雙點交配與mask交配三種策略，增強族群多樣性。突變操作則依不同設定，採用位元反轉、string flip與mask等三種突變方法，以防止早熟收斂並維持探索能力。整體流程包含初始化族群、重複進化、選擇、交配、突變，並持續追蹤最大適應度直到達成目標或迭代結束。

* 1. **Equation**

**Fitness function：**

，−10 ≤𝑥 ≤ 10

1. **BGA：**

Reproduction：RWS or TS

Crossover：one-point、two-point、mask

Mutation：bit、string、mask

1. **RGA：**

Reproduction：RWS or TS

Crossover：x1′ = x1 + 𝜎(x1 − x2) ；x2 ′ = x1− 𝜎(x1 − x2)，−1≤ 𝜎 ≤1

Mutation： x′ = x+𝑠∗𝑟𝑎𝑛𝑑𝑜𝑚𝑛𝑜𝑖𝑠𝑒， 𝑛𝑜𝑖𝑠𝑒 ∈ [−1 1]

1. **EA：**

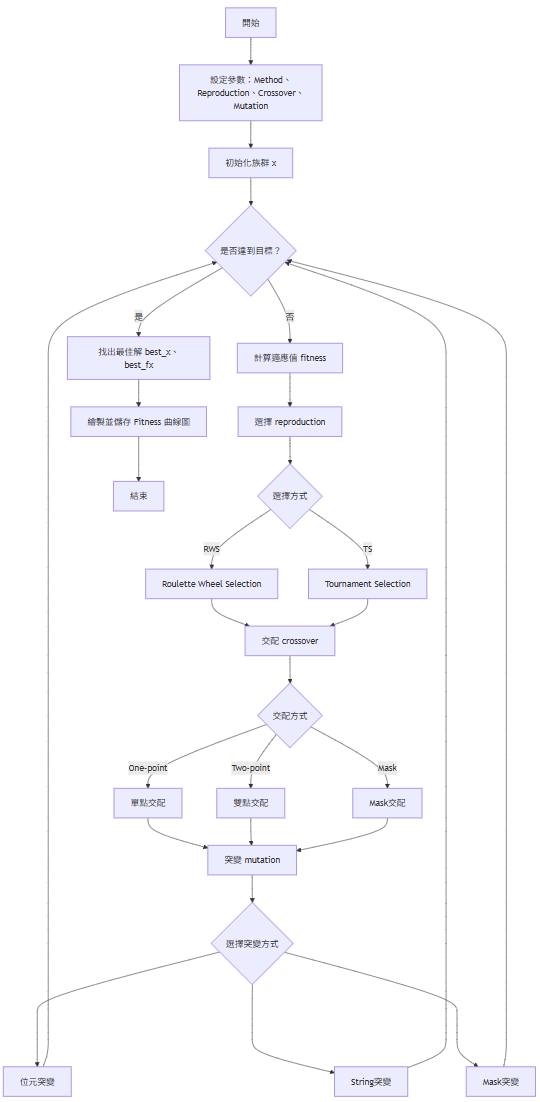
Reproduction：RWS or TS

Crossover：average crossover operator – 𝑥 = [xj+ xk]

convex combination operator – 𝑥 = 𝑟 xj +(1−𝑟) xk，𝑟 ∈ (0,1)

Mutation：

* 1. **Program flow chart：**



1. **Simulation results**
   1. **Programmer Code：**

clc;

clear;

close all;

% === 參數設定 ===

Method = 1; % BGA(1), RGA(2), EA(3)

Reproduction\_method = 1; % 選擇方式：RWS(1), TS(2)

Crossover\_method = 1; % 交配方式：onepoint(1), twopoint(2), mask(3)

Mutation\_method = 1; % 突變方式：bit(1), string(2), mask(3)

% === 題目 ===

px = -10:0.1:10;

py = -15\*(sin(2\*px).^2) - (px - 2).^2 + 160;

f1=(py).^5;

% === 初始化 ===

x = (randi([-100 100], 10, 1)) \* 0.1; % 第一代

cr = 0.8; % 交配率

mr = 0.01; % 突變率

generation\_max = [];

% === 進化過程 ===

for generation = 1:500

% 計算適應值

fitness = (-15\*(sin(2\*x).^2) - (x-2).^2 + 160).^5;

generation\_max(generation) = max(fitness);

% 停止條件

if generation\_max(generation) >= (-15\*(sin(2\*1.6))^2 - (1.6-2)^2 + 160)^5

break;

end

% 選擇

newx = reproduction(x, fitness, Reproduction\_method);

% 交配

if Method == 1

binary\_x = decimal\_to\_binary(newx);

binary\_x = crossover(binary\_x, Crossover\_method);

% 突變

binary\_x = mutation(binary\_x, Mutation\_method);

% 轉回十進制

x = binary\_to\_decimal(binary\_x);

else

x = crossover\_real(newx);

% 突變

if mod(generation, 10) == 0

x = mutation\_real(x);

end

end

end

% === 畫圖 ===

% figure; plot(1:generation, generation\_max, 'r');

% xlabel('generation'); ylabel('fitness-max');

% if Method==1

% title('BGA');

% elseif Method==2

% title('RGA');

% else

% title('EA');

% end

% === 找最好的解 ===

fitness\_final = (-15\*(sin(2\*x).^2) - (x-2).^2 + 160).^5;

[max\_fitness, idx\_best] = max(fitness\_final);

best\_x = x(idx\_best);

best\_fx = (-15\*(sin(2\*best\_x)^2) - (best\_x-2)^2 + 160);

fprintf('最好的 x = %.4f\n', best\_x);

fprintf('對應的 f(x) = %.4f\n', best\_fx);

% === 畫圖並存檔 ===

% figure;

% plot(1:generation, generation\_max, 'r', 'LineWidth', 1);

% xlabel('Generation');

% ylabel('Fitness-max');

% if Method==1

% title('BGA');

% filename = 'BGA\_result.png';

% elseif Method==2

% title('RGA');

% filename = 'RGA\_result.png';

% else

% title('EA');

% filename = 'EA\_result.png';

% end

% 儲存圖片

% saveas(gcf, filename);

% fprintf('圖已經儲存成檔案：%s\n', filename);

% === 畫圖 ===

% figure;

% plot(px, py); title('題目'); xlabel('x'); ylabel('f(x)');

% figure(2);

% plot(px,f1);

% title('fittness function');

% xlabel('x');

% ylabel('f1(x)');

% === 畫圖並存檔 ===

figure;

plot(1:generation, generation\_max, 'r', 'LineWidth', 1);

xlabel('Generation');

ylabel('Fitness-max');

method\_name = ["BGA", "RGA", "EA"];

repro\_name = ["RWS", "TS"];

cross\_name = ["onepoint", "twopoint", "mask"];

mutate\_name = ["bit", "string", "mask"];

title\_str = sprintf('%s - %s - %s - %s', ...

method\_name(Method), ...

repro\_name(Reproduction\_method), ...

cross\_name(Crossover\_method), ...

mutate\_name(Mutation\_method));

title(title\_str);

filename = sprintf('%s\_%s\_%s\_%s.png', ...

method\_name(Method), ...

repro\_name(Reproduction\_method), ...

cross\_name(Crossover\_method), ...

mutate\_name(Mutation\_method));

% 如果資料夾不存在就自動建立

if ~exist('result', 'dir')

mkdir('result');

end

saveas(gcf, fullfile('result', filename));

fprintf('圖已經儲存成 result\\%s\n', filename);

%% === Functions ===

function newx = reproduction(x, fitness, method)

% RWS 或 TS

newx = zeros(size(x));

if method == 1

% RWS

total\_fit = sum(fitness);

pick = rand(size(x)) \* total\_fit;

cumfit = cumsum(fitness);

for i = 1:length(x)

newx(i) = x(find(cumfit >= pick(i), 1));

end

else

% TS

for i = 1:length(x)

idx = randperm(length(x), 2);

if fitness(idx(1)) > fitness(idx(2))

newx(i) = x(idx(1));

else

newx(i) = x(idx(2));

end

end

end

end

function binary\_x = decimal\_to\_binary(x)

binary\_x = zeros(length(x),10);

for i=1:length(x)

temp = x(i);

if temp>=0

sgn = 0;

else

sgn = 1;

end

temp = abs(temp);

intpart = floor(temp);

fracpart = temp - intpart;

binint = dec2bin(intpart,4)-'0';

binfrac = zeros(1,5);

for j=1:5

fracpart = fracpart\*2;

binfrac(j) = floor(fracpart);

fracpart = fracpart - binfrac(j);

end

binary\_x(i,:) = [sgn binint binfrac];

end

end

function x = binary\_to\_decimal(binary\_x)

x = zeros(size(binary\_x,1),1);

for i=1:size(binary\_x,1)

intval = binary\_x(i,2:5) \* [8;4;2;1];

fracval = binary\_x(i,6:end) \* (0.5.^(1:5))';

val = intval + fracval;

if binary\_x(i,1) == 1

val = -val;

end

x(i) = round(val,1);

end

end

function binary\_x = crossover(binary\_x, method)

idx = randperm(size(binary\_x,1));

for i=1:2:8

if method == 1

% one point

point = randi([1 10]);

tmp = binary\_x(idx(i), point:end);

binary\_x(idx(i), point:end) = binary\_x(idx(i+1), point:end);

binary\_x(idx(i+1), point:end) = tmp;

elseif method == 2

% two points

pts = sort(randi([1 10],1,2));

tmp = binary\_x(idx(i), pts(1):pts(2));

binary\_x(idx(i), pts(1):pts(2)) = binary\_x(idx(i+1), pts(1):pts(2));

binary\_x(idx(i+1), pts(1):pts(2)) = tmp;

elseif method == 3

% mask

mask = randi([0 1],1,10);

tmp = binary\_x(idx(i),:);

binary\_x(idx(i),mask==1) = binary\_x(idx(i+1),mask==1);

binary\_x(idx(i+1),mask==1) = tmp(mask==1);

end

end

end

function binary\_x = mutation(binary\_x, method)

idx = randi(size(binary\_x,1));

if method == 1

% bit flip

pos = randi(10);

binary\_x(idx,pos) = 1 - binary\_x(idx,pos);

elseif method == 2

% 全翻

binary\_x(idx,:) = 1 - binary\_x(idx,:);

elseif method == 3

% mask xor

mask = randi([0 1],1,10);

binary\_x(idx,:) = xor(binary\_x(idx,:), mask);

end

end

function newx = crossover\_real(x)

idx = randperm(length(x));

newx = x;

for i=1:2:8

alpha = (randi([-10 10],1))\*0.1;

newx(idx(i)) = x(idx(i)) + round(alpha\*(x(idx(i))-x(idx(i+1))),1);

newx(idx(i+1)) = x(idx(i+1)) + round(alpha\*(x(idx(i))-x(idx(i+1))),1);

end

end

function x = mutation\_real(x)

idx = randperm(length(x));

S = rand();

noise = -1 + 2\*rand();

x(idx(1)) = x(idx(1)) + round(S\*noise,1);

end

* 1. **Result：**
     1. **題目＆適應性函數**

|  |
| --- |
| **題目** |
| **一張含有 文字, 圖表, 行, 繪圖 的圖片  AI 產生的內容可能不正確。** |
| **適應性函數** |
|  |

* + 1. **BGA**

|  |
| --- |
| BGA\_RWS\_onepoint\_bit |
|  |

最好的 x = 1.6000

對應的 f(x) = 159.7889

|  |
| --- |
| BGA\_RWS\_onepoint\_ string |
|  |

最好的 x = 1.5000

對應的 f(x) = 159.4513

|  |
| --- |
| BGA\_RWS\_onepoint\_mask |
|  |

最好的 x = 1.6000

對應的 f(x) = 159.7889

|  |
| --- |
| BGA\_RWS\_twopoint\_bit |
|  |

最好的 x = 1.6000

對應的 f(x) = 159.7889

|  |
| --- |
| BGA\_RWS\_twopoint\_string |
|  |

最好的 x = 2.0000

對應的 f(x) = 151.4087

|  |
| --- |
| BGA\_RWS\_twopoint\_mask |
|  |

最好的 x = 1.6000

對應的 f(x) = 159.7889

|  |
| --- |
| BGA\_RWS\_mask\_bit |
|  |

最好的 x = 1.6000

對應的 f(x) = 159.7889

|  |
| --- |
| BGA\_RWS\_mask\_string |
|  |

最好的 x = 1.7000

對應的 f(x) = 158.9305

|  |
| --- |
| BGA\_RWS\_mask\_mask |
|  |

最好的 x = 1.6000

對應的 f(x) = 159.7889

|  |
| --- |
| BGA\_TS\_onepoint\_bit |
|  |

最好的 x = 1.4000

對應的 f(x) = 157.9567

|  |
| --- |
| BGA\_TS\_onepoint\_string |
|  |

最好的 x = 1.5000

對應的 f(x) = 159.4513

|  |
| --- |
| BGA\_TS\_onepoint\_mask |
|  |

最好的 x = 1.6000

對應的 f(x) = 159.7889

|  |
| --- |
| BGA\_TS\_twopoint\_bit |
|  |

最好的 x = 3.1000

對應的 f(x) = 158.6864

|  |
| --- |
| BGA\_TS\_twopoint\_string |
|  |

最好的 x = 1.5000

對應的 f(x) = 159.4513

|  |
| --- |
| BGA\_TS\_twopoint\_mask |
|  |

最好的 x = 1.6000

對應的 f(x) = 159.7889

|  |
| --- |
| BGA\_TS\_mask\_bit |
|  |

最好的 x = 1.4000

對應的 f(x) = 157.9567

|  |
| --- |
| BGA\_TS\_mask\_string |
|  |

最好的 x = 1.3000

對應的 f(x) = 155.5239

|  |
| --- |
| BGA\_TS\_mask\_mask |
|  |

最好的 x = 1.6000

對應的 f(x) = 159.7889

* + 1. **RGA**

|  |
| --- |
| RGA\_RWS\_onepoint\_bit |
|  |

最好的 x = 1.6000

對應的 f(x) = 159.7889

|  |
| --- |
| RGA\_RWS\_onepoint\_string |
|  |

最好的 x = 1.6000

對應的 f(x) = 159.7889

|  |
| --- |
| RGA\_RWS\_onepoint\_mask |
|  |

最好的 x = 1.6000

對應的 f(x) = 159.7889

|  |
| --- |
| RGA\_RWS\_twopoint\_bit |
|  |

最好的 x = 1.6000

對應的 f(x) = 159.7889

|  |
| --- |
| RGA\_RWS\_twopoint\_string |
|  |

最好的 x = 1.6000

對應的 f(x) = 159.7889

|  |
| --- |
| RGA\_RWS\_twopoint\_mask |
|  |

最好的 x = 1.6000

對應的 f(x) = 159.7889

|  |
| --- |
| RGA\_RWS\_mask\_bit |
|  |

最好的 x = 1.6000

對應的 f(x) = 159.7889

|  |
| --- |
| RGA\_RWS\_mask\_string |
|  |

最好的 x = 1.6000

對應的 f(x) = 159.7889

|  |
| --- |
| RGA\_RWS\_mask\_mask |
|  |

最好的 x = 1.6000

對應的 f(x) = 159.7889

|  |
| --- |
| RGA\_TS\_onepoint\_bit |
|  |

最好的 x = 1.6000

對應的 f(x) = 159.7889

|  |
| --- |
| RGA\_TS\_onepoint\_string |
|  |

最好的 x = 1.6000

對應的 f(x) = 159.7889

|  |
| --- |
| RGA\_TS\_onepoint\_mask |
|  |

最好的 x = 1.6000

對應的 f(x) = 159.7889

|  |
| --- |
| RGA\_TS\_twopoint\_bit |
|  |

最好的 x = 1.6000

對應的 f(x) = 159.7889

|  |
| --- |
| RGA\_TS\_twopoint\_string |
|  |

最好的 x = 1.6000

對應的 f(x) = 159.7889

|  |
| --- |
| RGA\_TS\_twopoint\_mask |
|  |

最好的 x = 1.6000

對應的 f(x) = 159.7889

|  |
| --- |
| RGA\_TS\_mask\_bit |
|  |

最好的 x = 1.6000

對應的 f(x) = 159.7889

|  |
| --- |
| RGA\_TS\_mask\_string |
|  |

最好的 x = 1.6000

對應的 f(x) = 159.7889

|  |
| --- |
| RGA\_TS\_mask\_mask |
|  |

最好的 x = 1.6000

對應的 f(x) = 159.7889

* + 1. **EA**

|  |
| --- |
| EA\_RWS\_onepoint\_bit |
|  |

最好的 x = 1.6000

對應的 f(x) = 159.7889

|  |
| --- |
| EA\_RWS\_onepoint\_ string |
|  |

最好的 x = 1.6000

對應的 f(x) = 159.7889

|  |
| --- |
| EA\_RWS\_onepoint\_mask |
|  |

最好的 x = 1.6000

對應的 f(x) = 159.7889

|  |
| --- |
| EA\_RWS\_twopoint\_bit |
|  |

最好的 x = 1.6000

對應的 f(x) = 159.7889

|  |
| --- |
| EA\_RWS\_twopoint\_string |
|  |

最好的 x = 1.6000

對應的 f(x) = 159.7889

|  |
| --- |
| EA\_RWS\_twopoint\_mask |
|  |

最好的 x = 1.6000

對應的 f(x) = 159.7889

|  |
| --- |
| EA\_RWS\_mask\_bit |
|  |

最好的 x = 1.6000

對應的 f(x) = 159.7889

|  |
| --- |
| EA\_RWS\_mask\_string |
|  |

最好的 x = 1.6000

對應的 f(x) = 159.7889

|  |
| --- |
| EA\_RWS\_mask\_mask |
|  |

最好的 x = 1.6000

對應的 f(x) = 159.7889

|  |
| --- |
| EA\_TS\_onepoint\_bit |
|  |

最好的 x = 1.6000

對應的 f(x) = 159.7889

|  |
| --- |
| EA\_TS\_onepoint\_string |
|  |

最好的 x = 1.6000

對應的 f(x) = 159.7889

|  |
| --- |
| EA\_TS\_onepoint\_mask |
|  |

最好的 x = 1.6000

對應的 f(x) = 159.7889

|  |
| --- |
| EA\_TS\_twopoint\_bit |
|  |

最好的 x = 1.6000

對應的 f(x) = 159.7889

|  |
| --- |
| EA\_TS\_twopoint\_string |
|  |

最好的 x = 1.6000

對應的 f(x) = 159.7889

|  |
| --- |
| EA\_TS\_twopoint\_mask |
|  |

最好的 x = 1.6000

對應的 f(x) = 159.7889

|  |
| --- |
| EA\_TS\_mask\_bit |
|  |

最好的 x = 1.6000

對應的 f(x) = 159.7889

|  |
| --- |
| EA\_TS\_mask\_string |
|  |

最好的 x = 1.6000

對應的 f(x) = 159.7889

|  |
| --- |
| EA\_TS\_mask\_mask |
|  |

最好的 x = 1.6000

對應的 f(x) = 159.7889

1. **Conclusion-analysis**

在本次模擬實作中，透過三種基因演算法與兩種選擇機制進行比較與分析，首先就三種基因演算法來看，EA整體表現最為穩定且具彈性。EA能夠更細緻地在連續空間中探索目標函數的極值位置，在我實作的EA版本中，不論選用 average 或 convex 交配方法，收斂速度與最終適應度表現都相當不錯，尤其搭配Tournament Selection時效果更為明顯。相比之下，BGA雖然結構簡單、實作容易，但在變異性與解析精度上稍顯不足，特別是當使用mask交配法與mask突變法時。而RGA則在穩定性方面表現最差，可能是因為RGA在crossover時易導致解空間劇烈跳動，若無適當控制，會使族群陷入震盪，進而影響整體效能。

綜合各種模擬結果來看，EA搭配Tournament Selection與average交配方式最能兼顧搜尋能力與收斂效率，其在多次執行中穩定地尋得近似最適解，且震盪幅度小。RGA因突變與交配導致過大解空間擾動，需精細調參才能達到良好表現。BGA雖然結構最簡單，但若採mask策略則需謹慎使用，以免延緩收斂。