智慧型系統概論 HW2

系級：通訊 4A　姓名：鄭筱筠　學號：01031012

一、 Object

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

f(x) = $-15\sin(2x)^2 - (x-2)^2 + 160$ ， $-10 \le x \le 10$

(1) BGA (10 bits)

(2) RGA

(3) EA

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

二、 Procedure

(一) 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。

(二) Equation

Fitness function：

$$f(x)^5 = [-15\sin(2x)^2 - (x-2)^2 + 160]^5，-10 \le x \le 10$$

1. **BGA：**

   Reproduction：RWS or TS

   Crossover：one-point、two-point、mask

   Mutation：bit、string、mask

2. **RGA：**

   Reproduction：RWS or TS

   Crossover：$x_1' = x_1 + \sigma(x_1 - x_2)$ ；$x_2' = x_1 - \sigma(x_1 - x_2)$，$-1 \le \sigma \le 1$

   Mutation： $x' = x + s * randomnoise$， $noise \in [-1\ 1]$

3. EA：

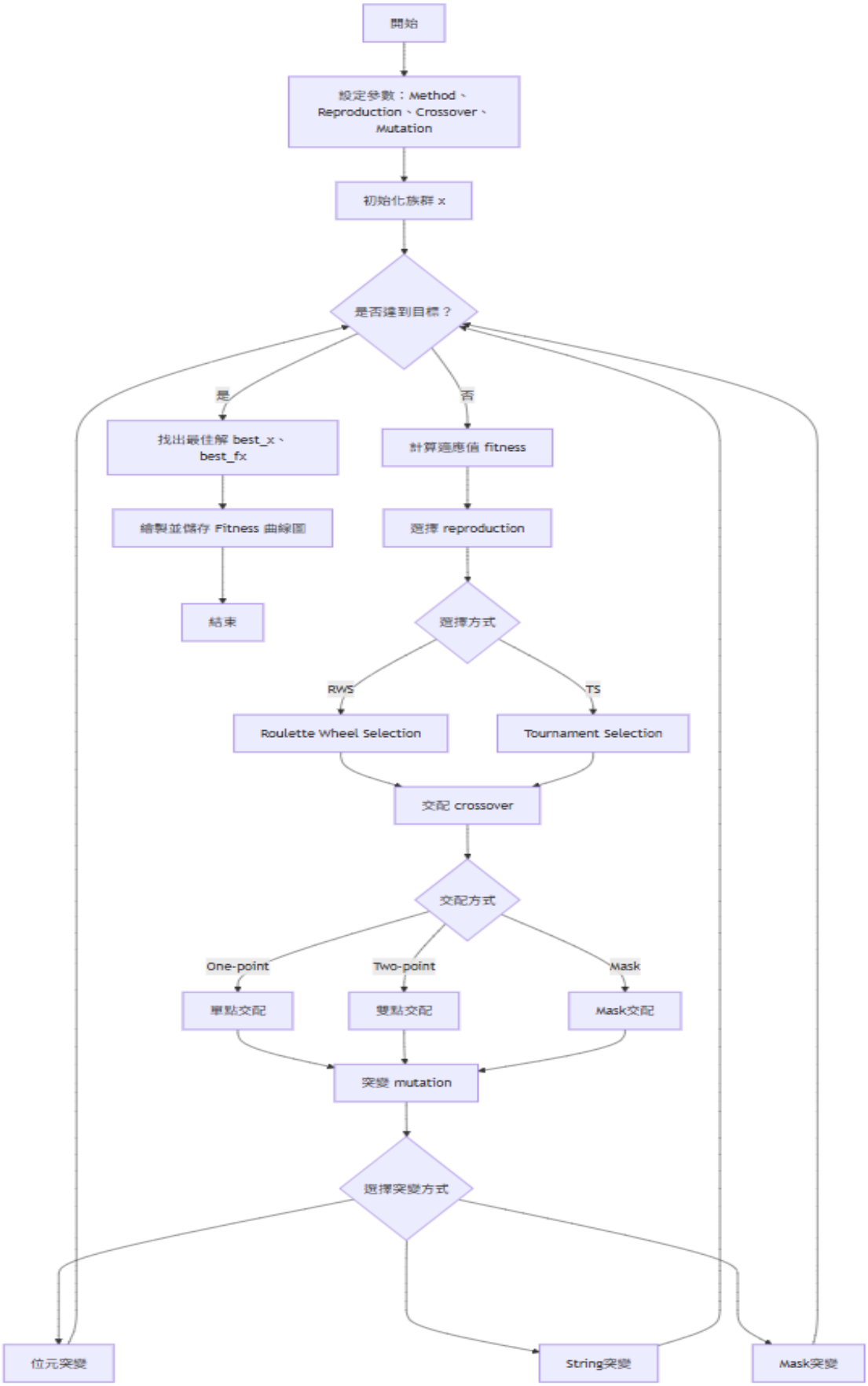
Reproduction：RWS or TS

Crossover：average crossover operator – $x = \frac{1}{2}\ [x_j + x_k]$

convex combination operator – $x = r\ x_j + (1-r)\ x_k$ ，$r \in (0,1)$

Mutation：$x' = x + rd$ ，$r \in (0,1)$ ，d: $randomly\ generated\ vector$

(三) Program flow chart：



三、 Simulation results

(一) 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)
```

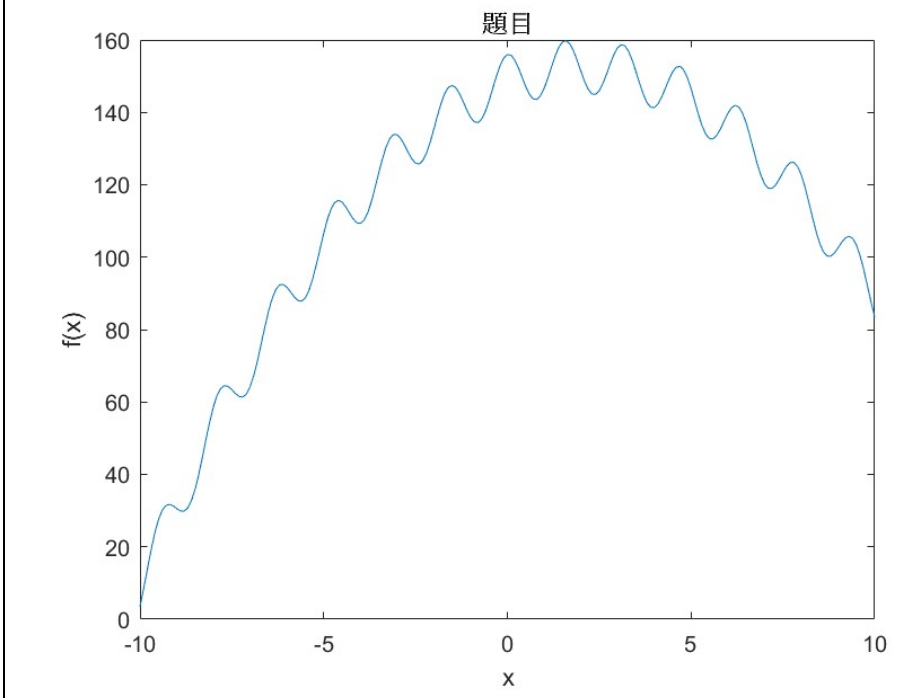
```
        intval = binary_x(i,2:5) * [8;4;2;1];
        fracval = binary_x(i,6:end) * (0.5.^(1:5))';
        val = intval + fracval;
        if binary_x(i,1) == 1
            val = -val;
        end
        x(i) = round(val,1);
    end
end

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
```

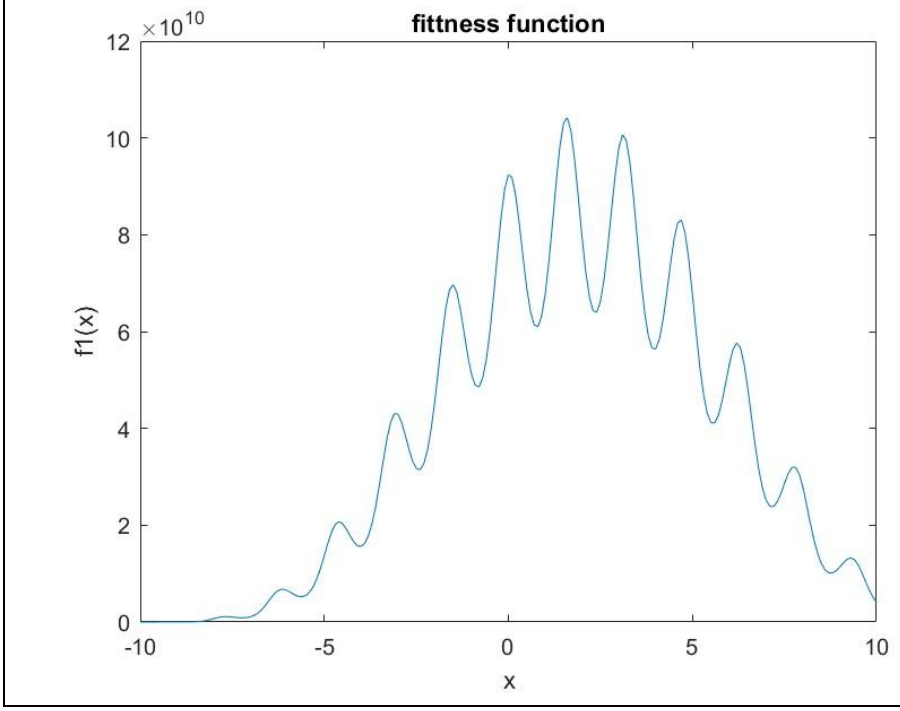
(二) Result：

1. 題目＆適應性函數

題目

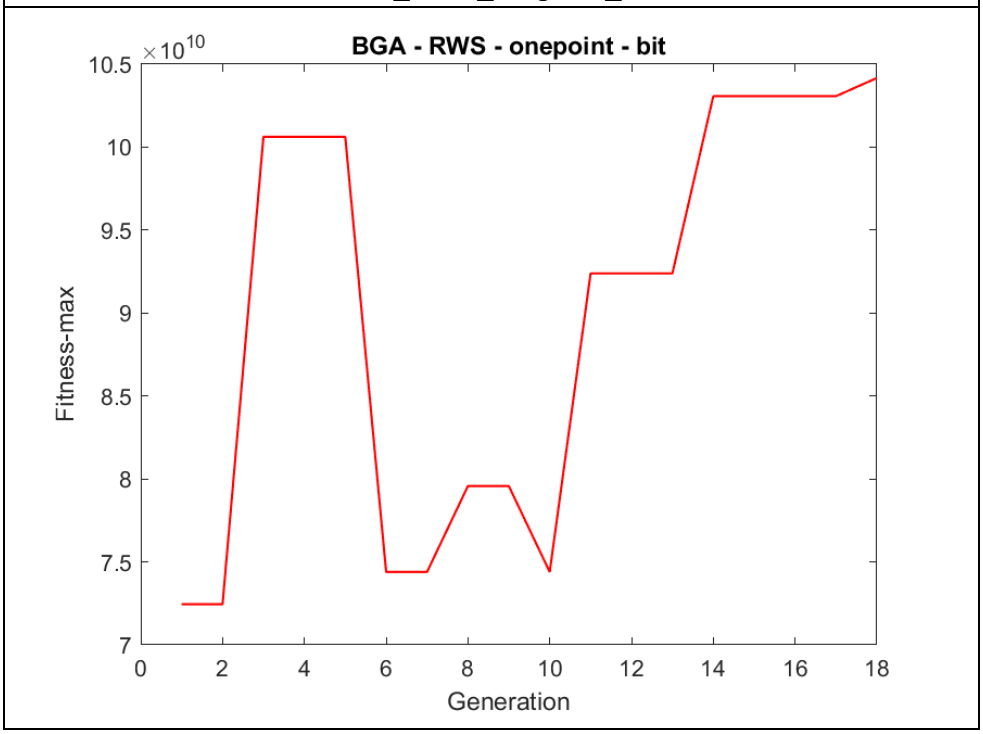


適應性函數



2. BGA

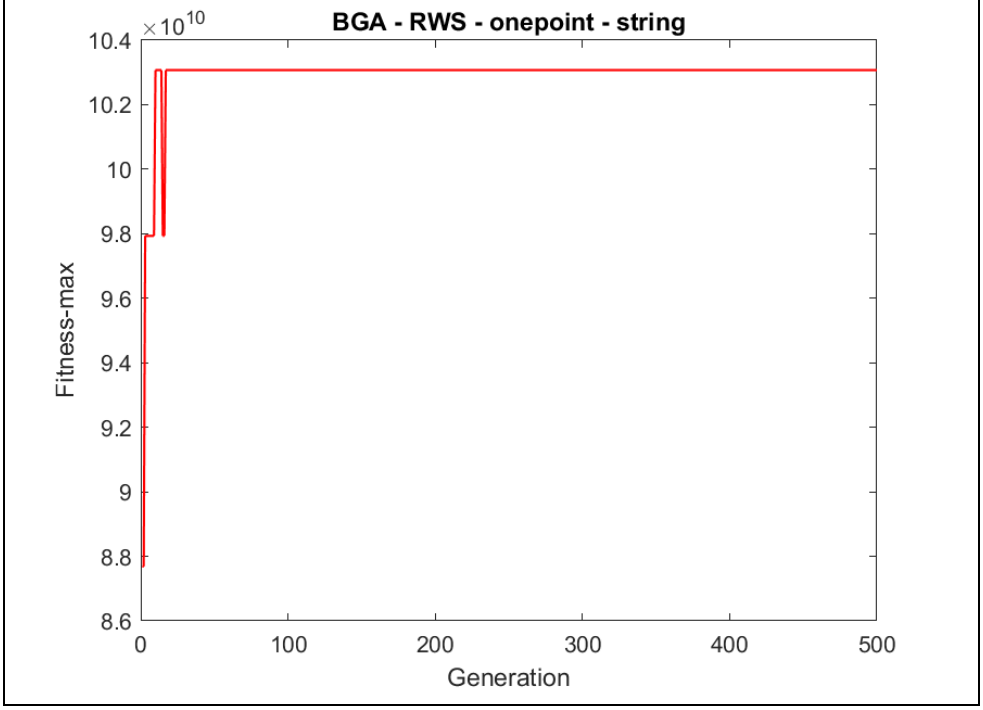
BGA_RWS_onepoint_bit



最好的 x = 1.6000

對應的 f(x) = 159.7889

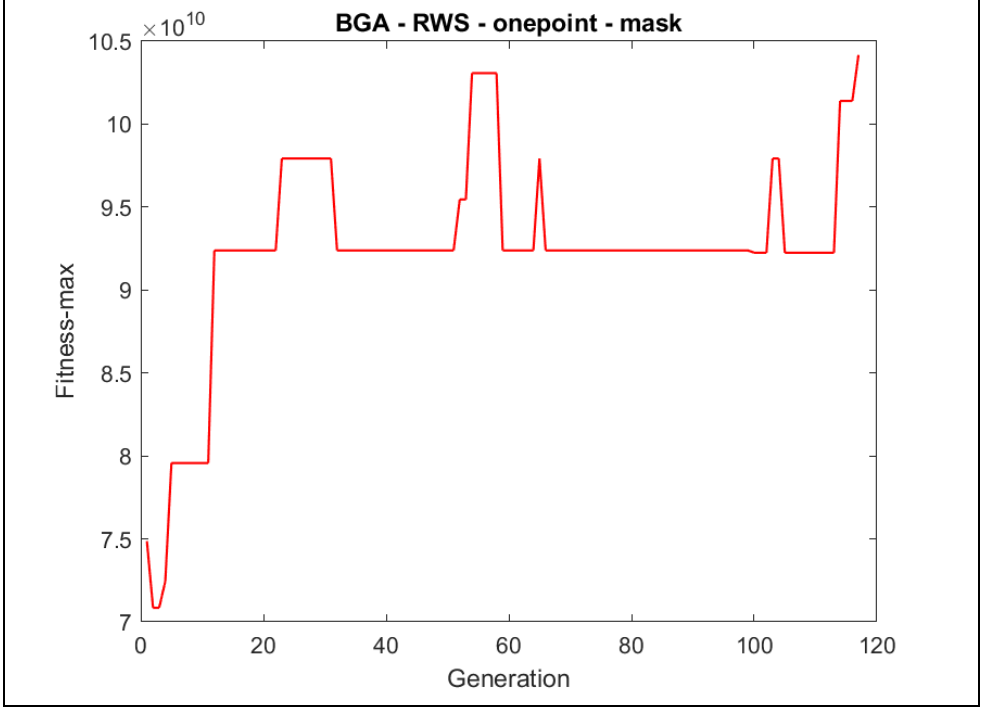
BGA_RWS_onepoint_ string



最好的 x = 1.5000

對應的 f(x) = 159.4513

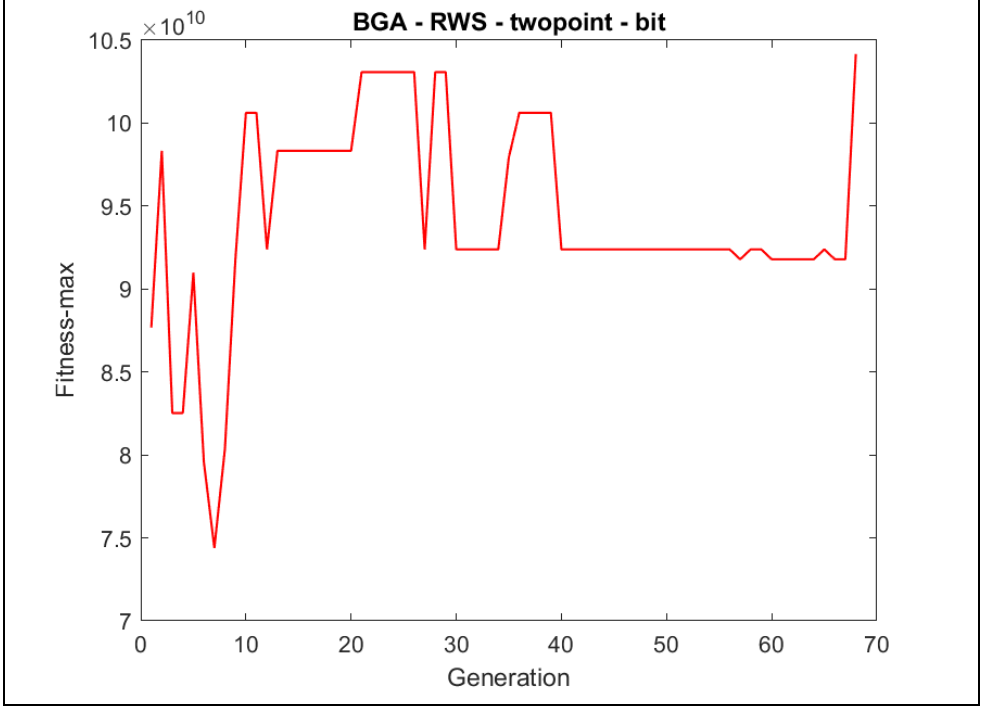
BGA_RWS_onepoint_mask



最好的 x = 1.6000

對應的 f(x) = 159.7889

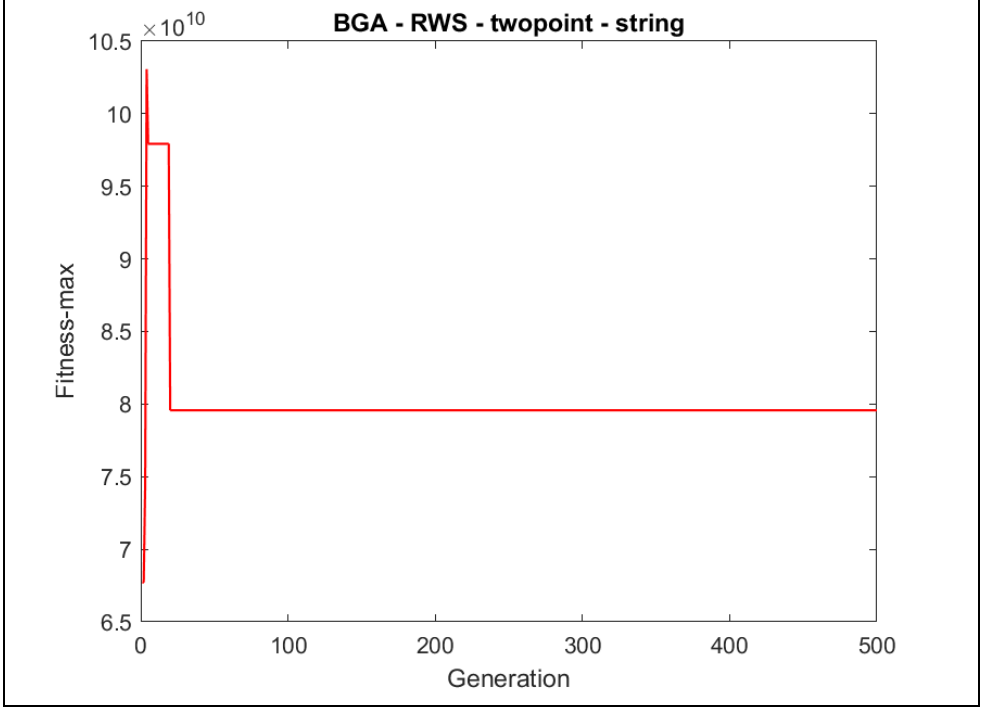
BGA_RWS_twopoint_bit



最好的 x = 1.6000

對應的 f(x) = 159.7889

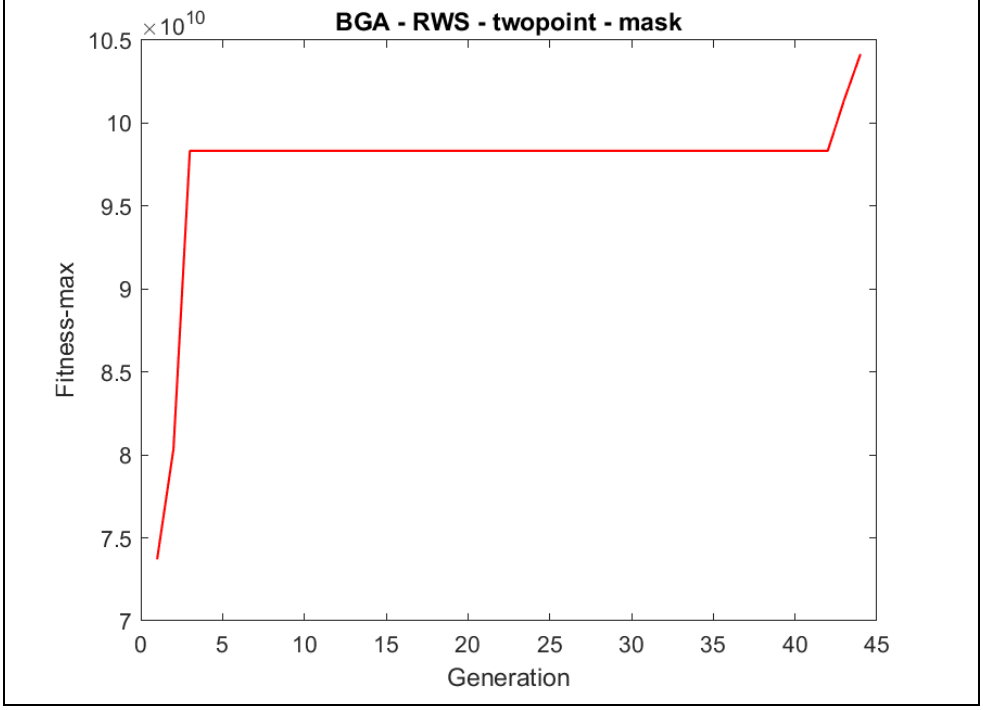
BGA_RWS_twopoint_string



最好的 x = 2.0000

對應的 f(x) = 151.4087

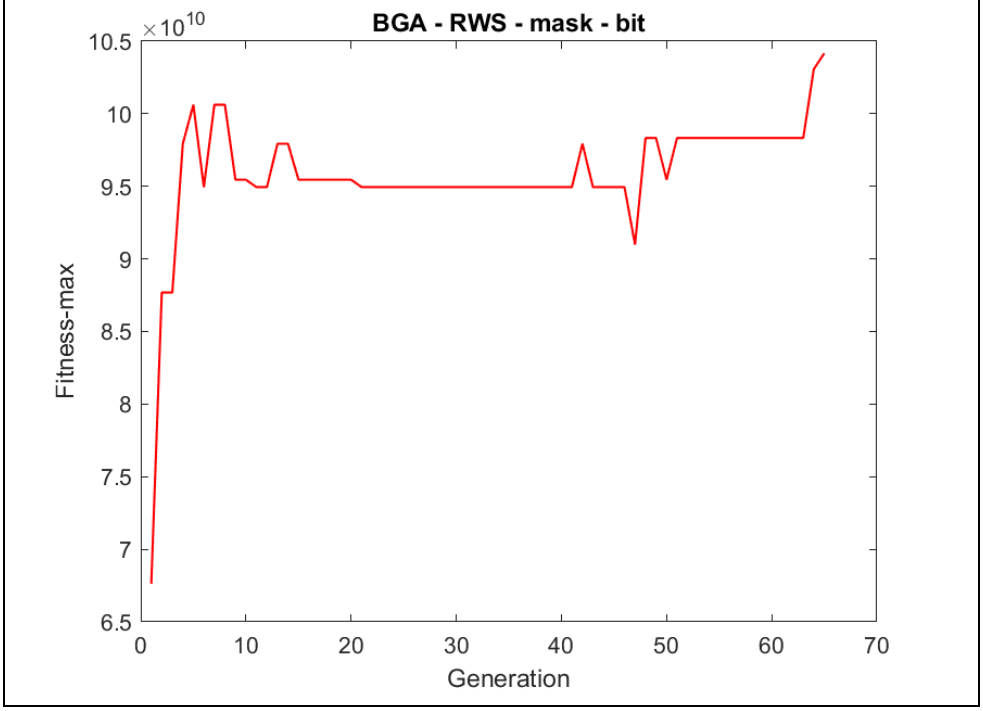
BGA_RWS_twopoint_mask



最好的 x = 1.6000

對應的 f(x) = 159.7889

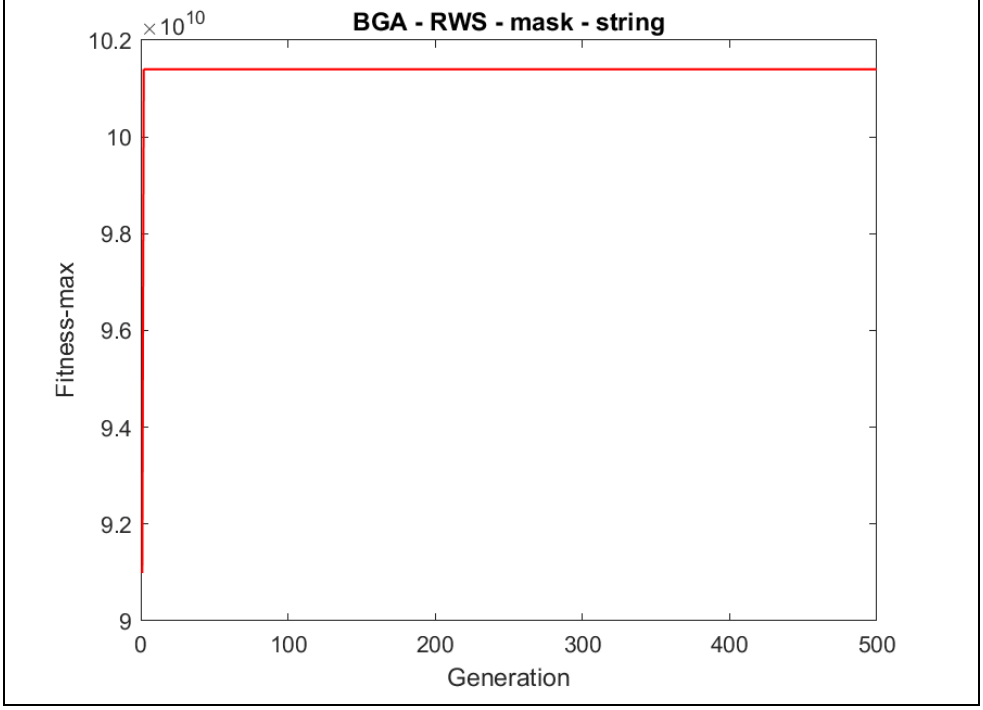
BGA_RWS_mask_bit



最好的 x = 1.6000

對應的 f(x) = 159.7889

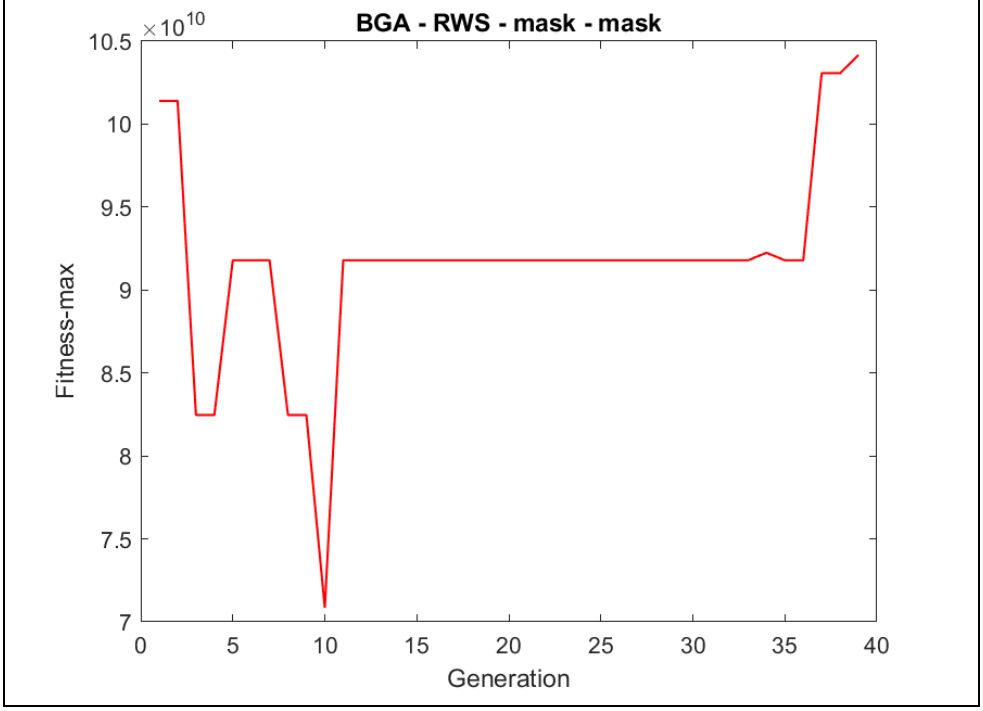
BGA_RWS_mask_string



最好的 x = 1.7000

對應的 f(x) = 158.9305

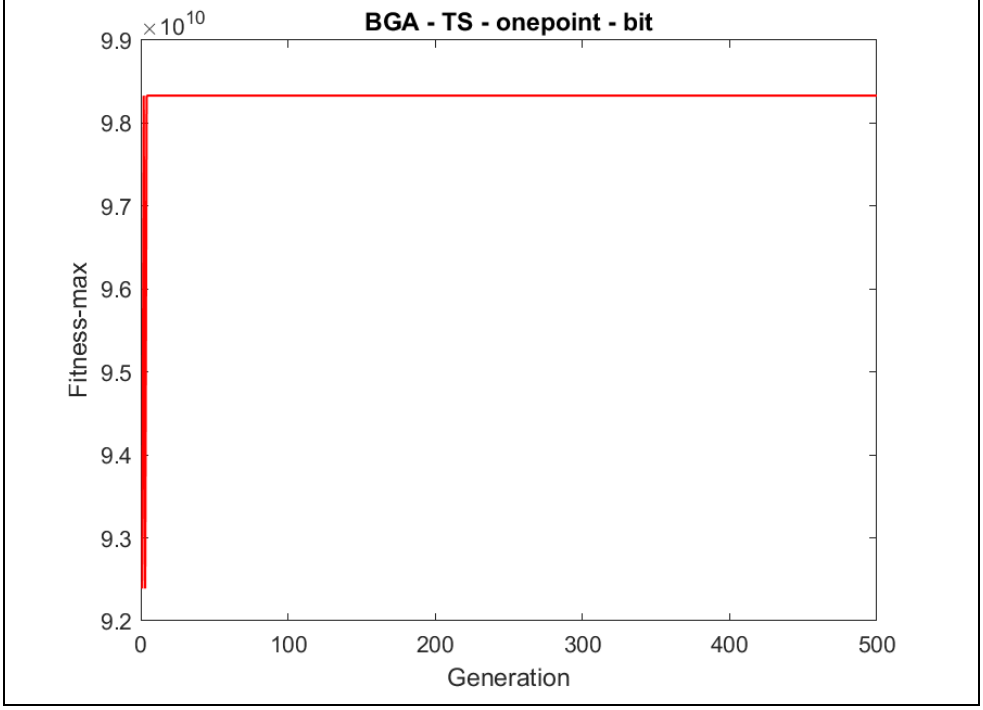
BGA_RWS_mask_mask



最好的 x = 1.6000

對應的 f(x) = 159.7889

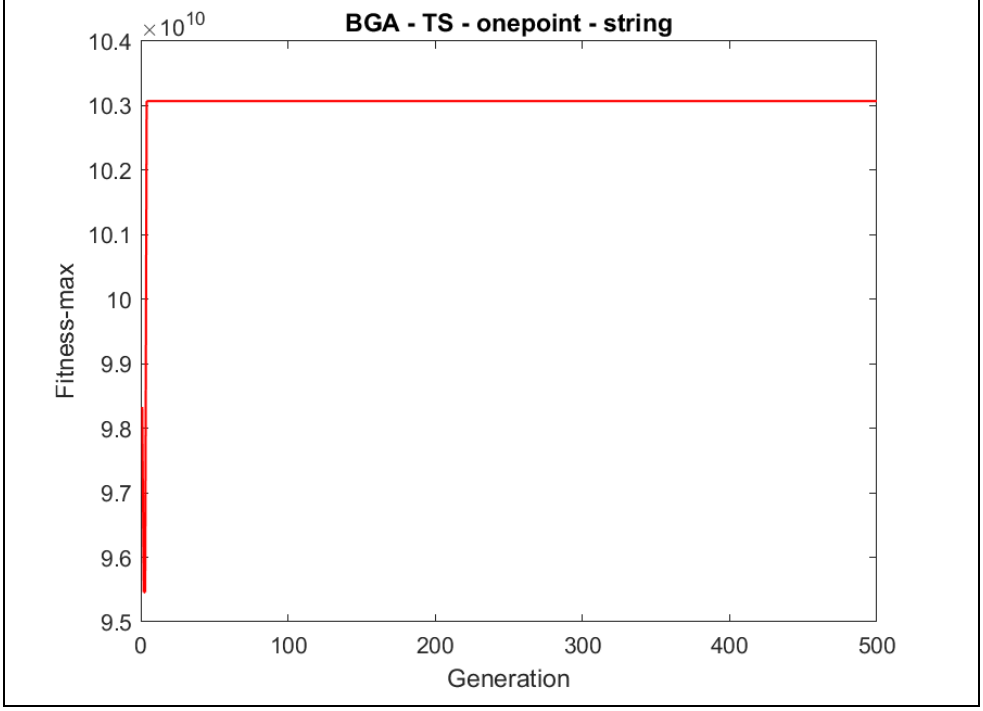
BGA_TS_onepoint_bit



最好的 x = 1.4000

對應的 f(x) = 157.9567

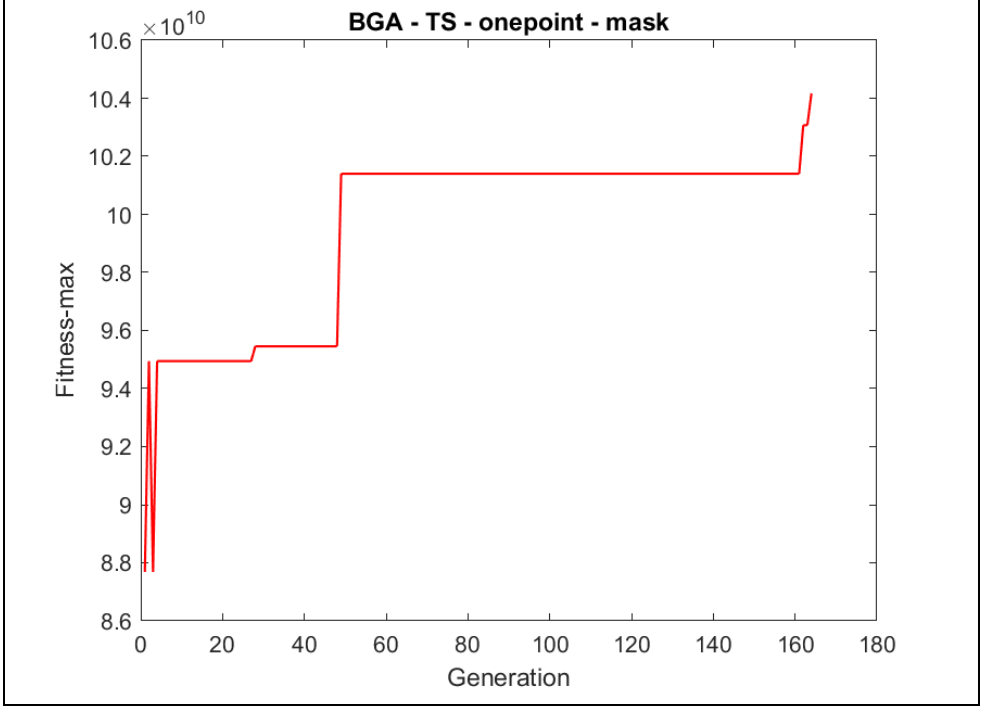
BGA_TS_onepoint_string



最好的 x = 1.5000

對應的 f(x) = 159.4513

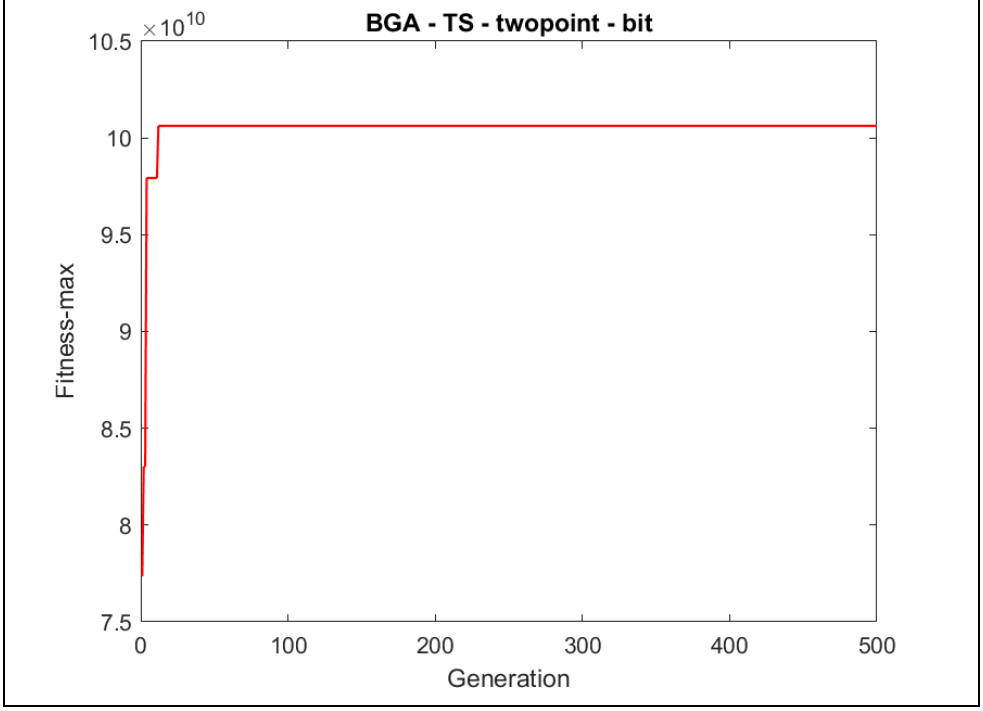
BGA_TS_onepoint_mask



最好的 x = 1.6000

對應的 f(x) = 159.7889

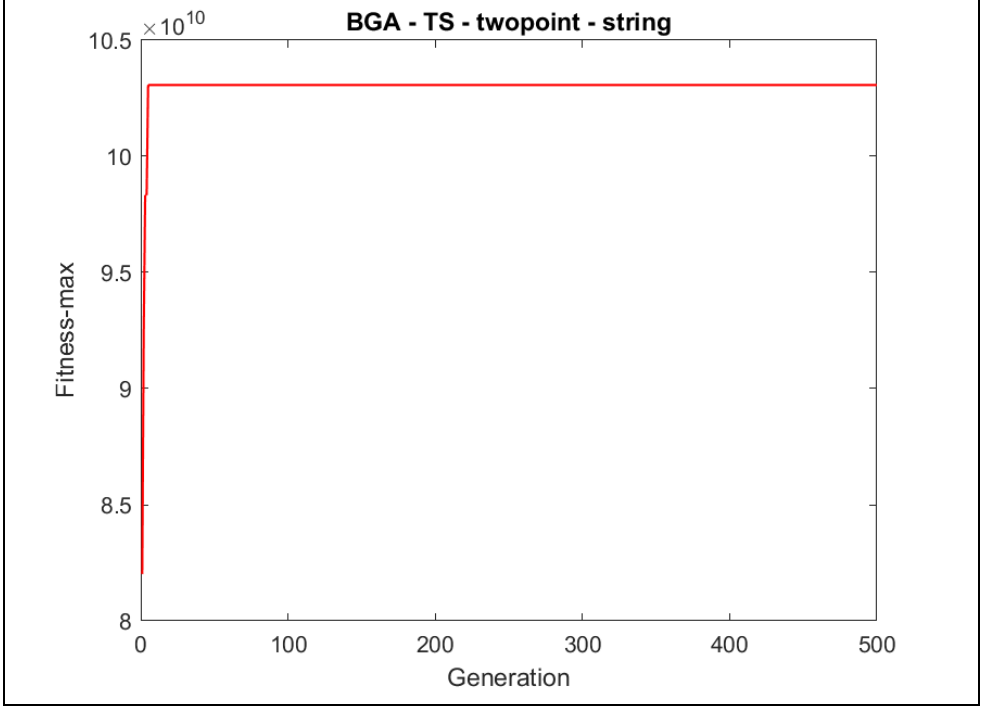
BGA_TS_twopoint_bit



最好的 x = 3.1000

對應的 f(x) = 158.6864

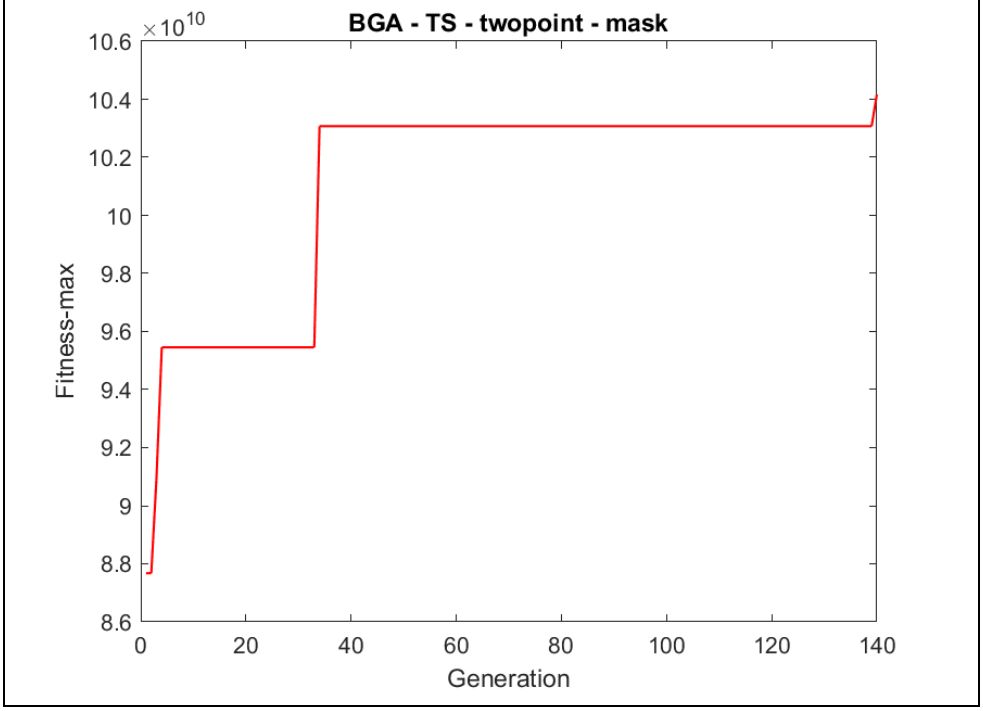
BGA_TS_twopoint_string



最好的 x = 1.5000

對應的 f(x) = 159.4513

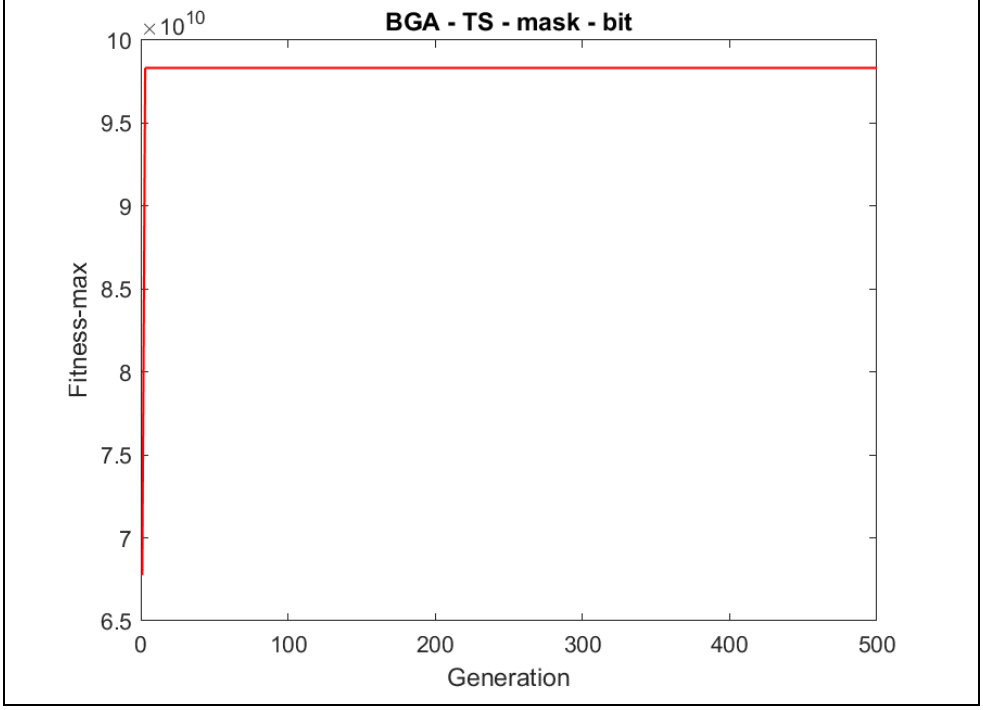
BGA_TS_twopoint_mask



最好的 x = 1.6000

對應的 f(x) = 159.7889

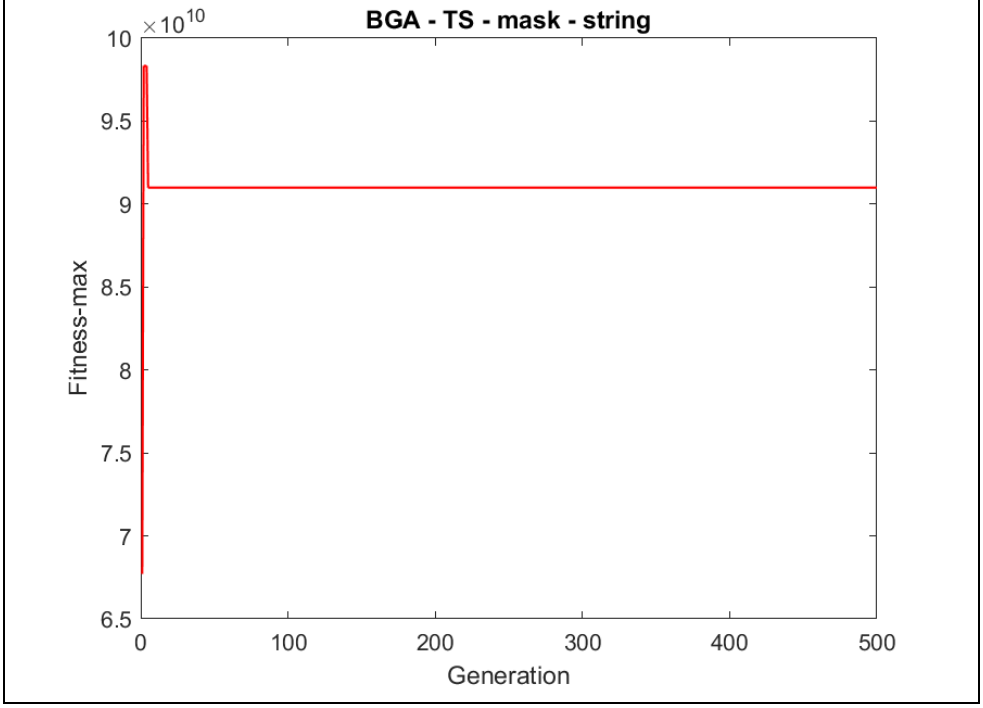
BGA_TS_mask_bit



最好的 x = 1.4000

對應的 f(x) = 157.9567

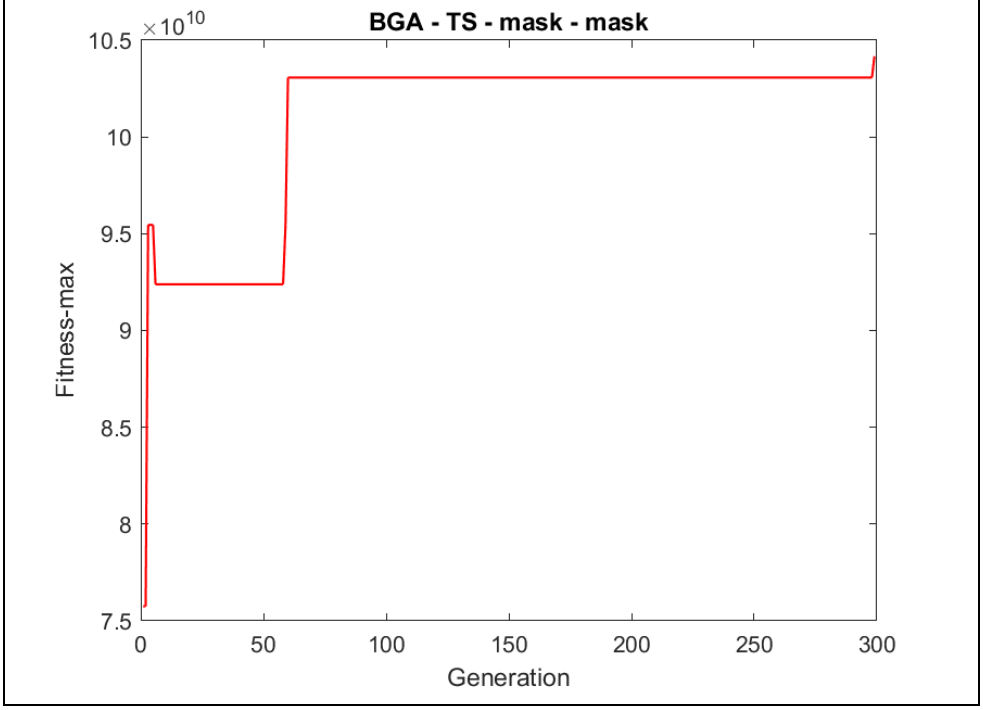
BGA_TS_mask_string



最好的 x = 1.3000

對應的 f(x) = 155.5239

BGA_TS_mask_mask

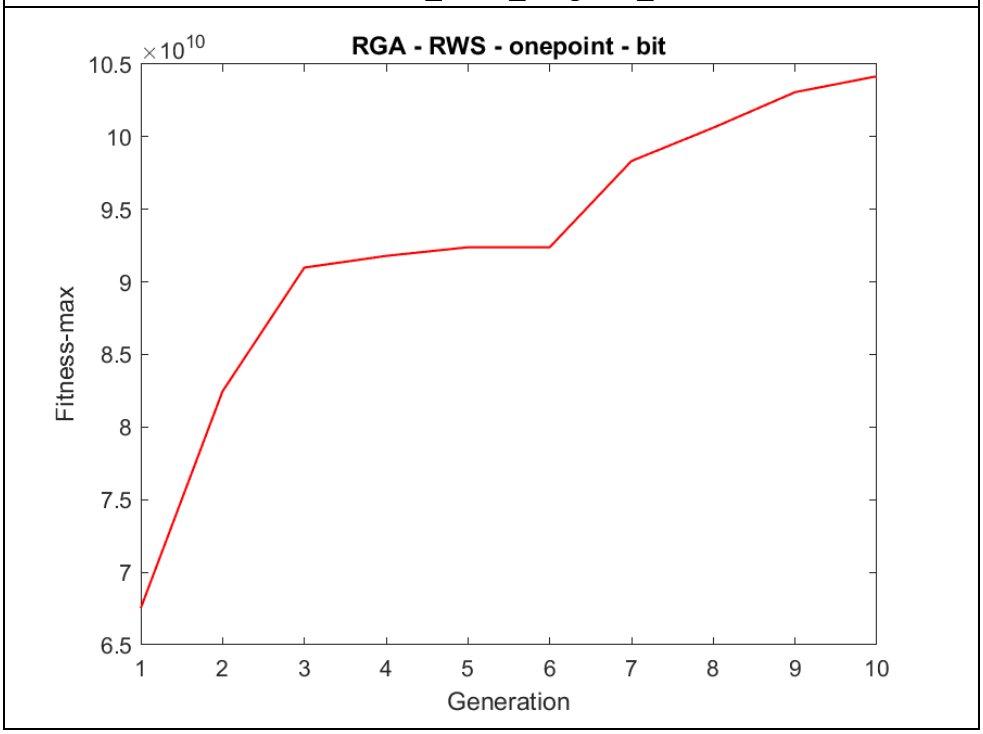


最好的 x = 1.6000

對應的 f(x) = 159.7889

3. RGA

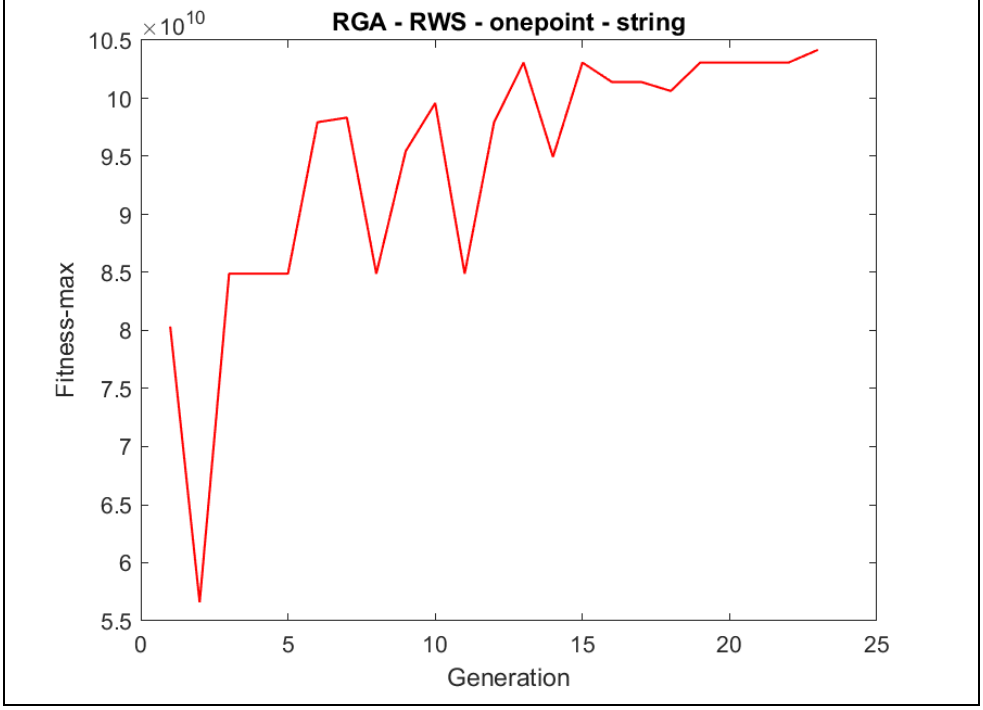
RGA_RWS_onepoint_bit



最好的 x = 1.6000

對應的 f(x) = 159.7889

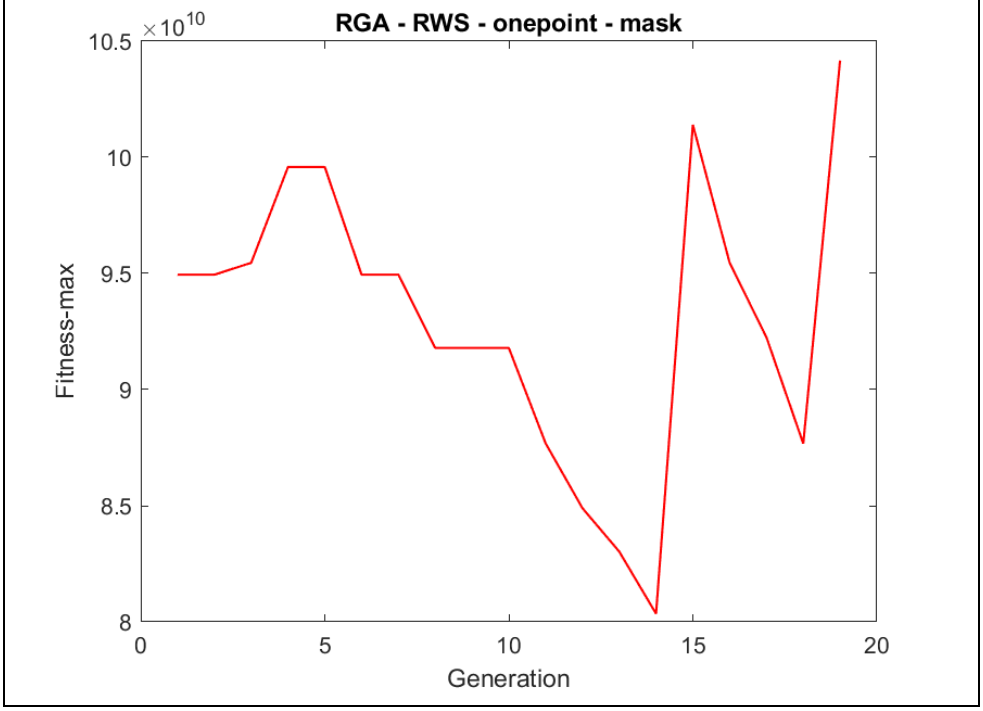
RGA_RWS_onepoint_string



最好的 x = 1.6000

對應的 f(x) = 159.7889

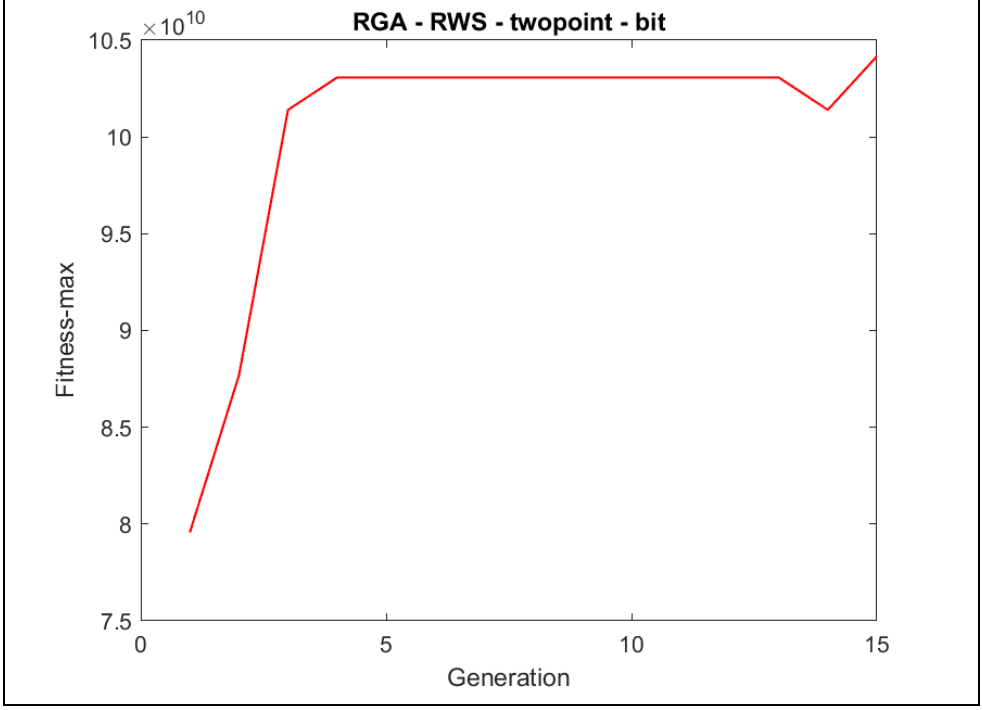
RGA_RWS_onepoint_mask



最好的 x = 1.6000

對應的 f(x) = 159.7889

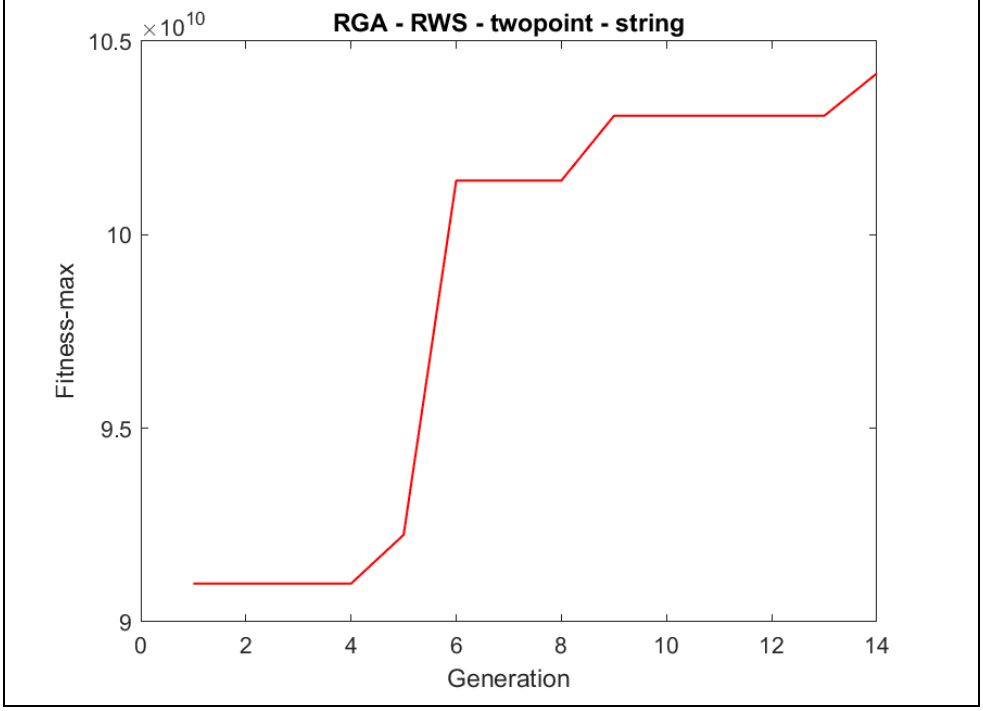
RGA_RWS_twopoint_bit



最好的 x = 1.6000

對應的 f(x) = 159.7889

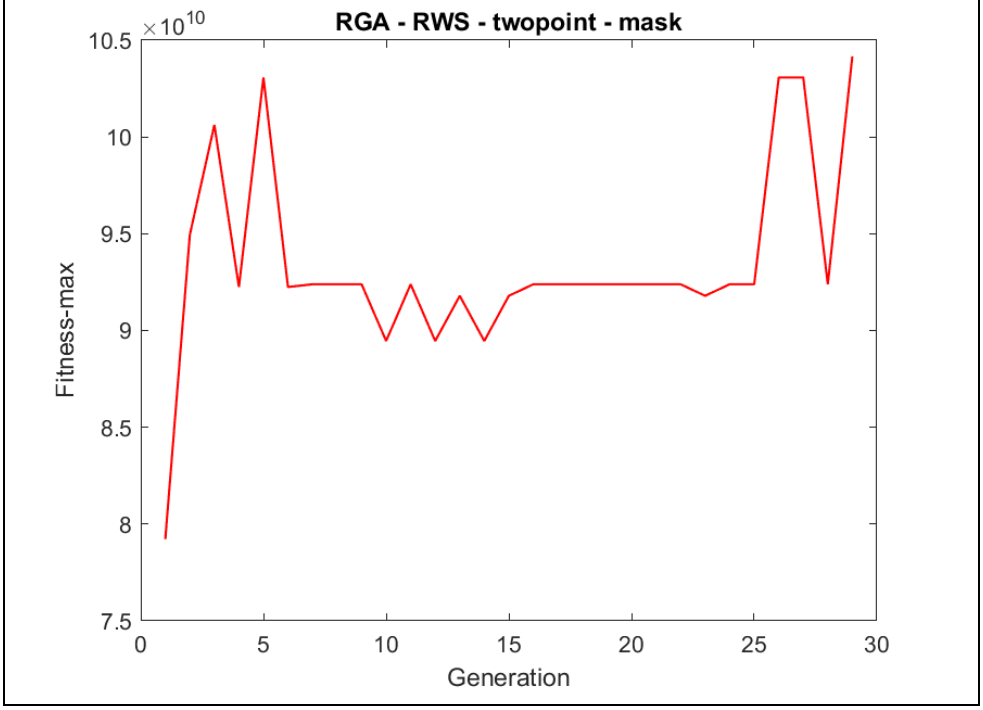
RGA_RWS_twopoint_string



最好的 x = 1.6000

對應的 f(x) = 159.7889

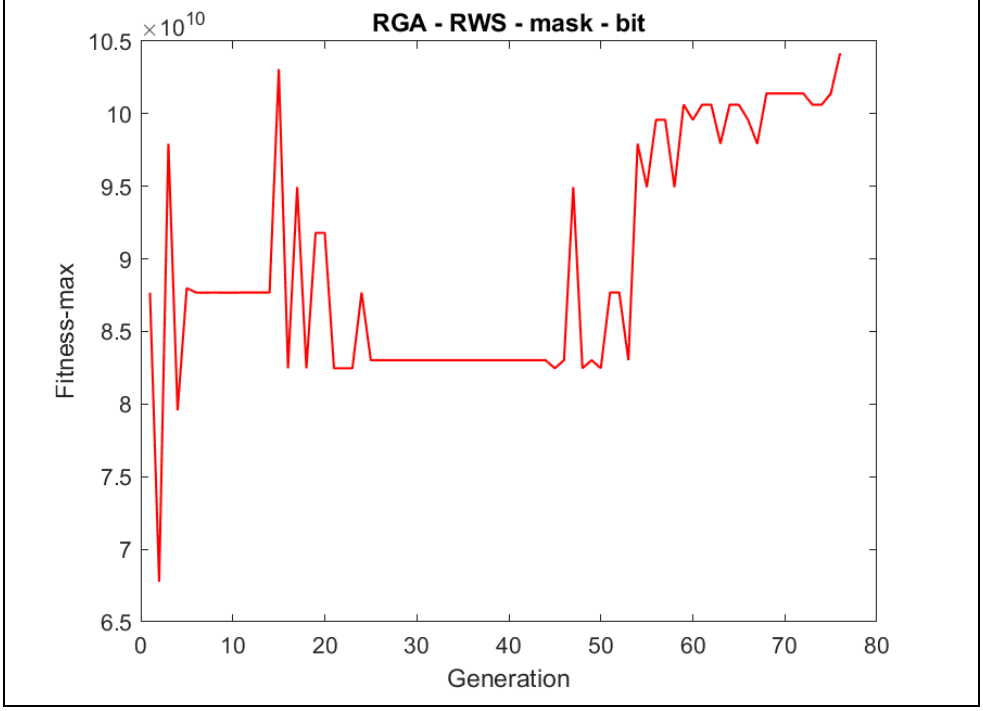
RGA_RWS_twopoint_mask



最好的 x = 1.6000

對應的 f(x) = 159.7889

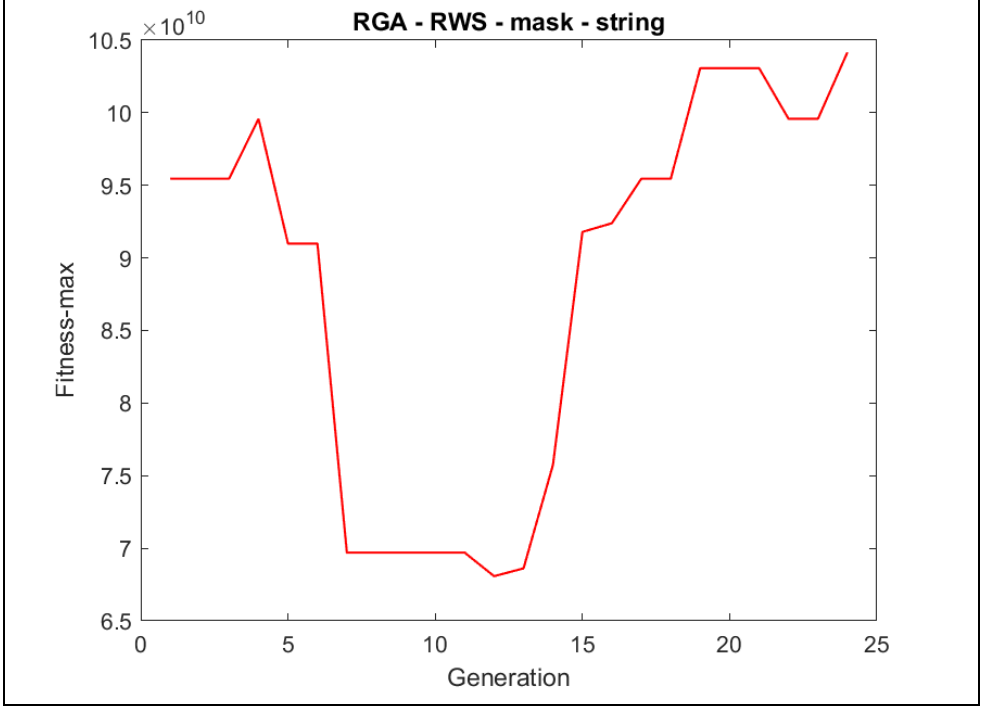
RGA_RWS_mask_bit



最好的 x = 1.6000

對應的 f(x) = 159.7889

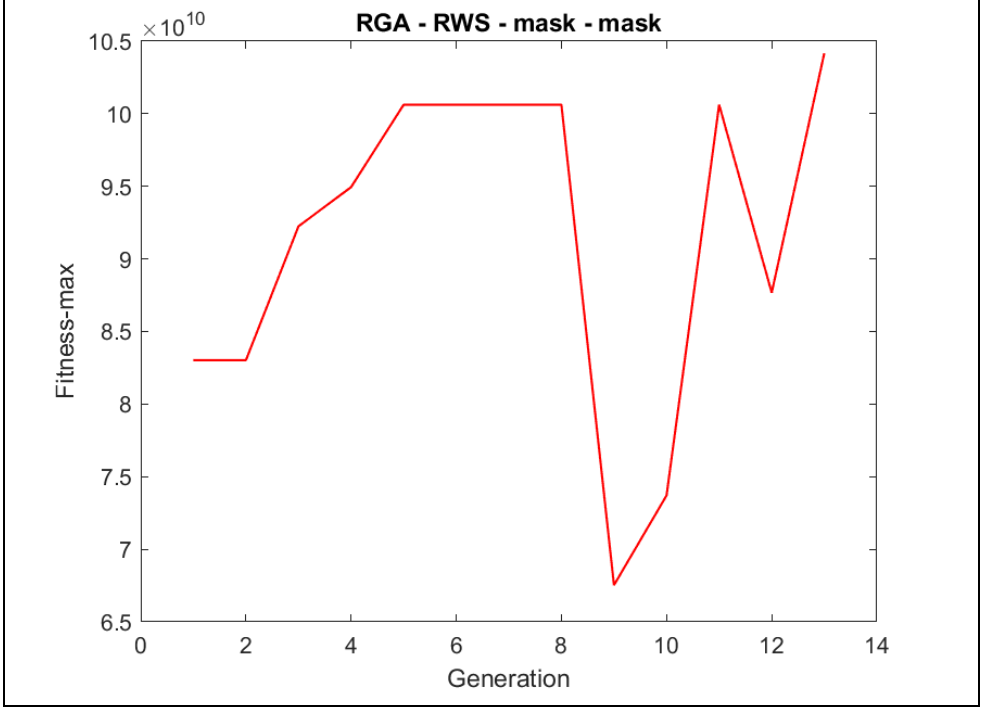
RGA_RWS_mask_string



最好的 x = 1.6000

對應的 f(x) = 159.7889

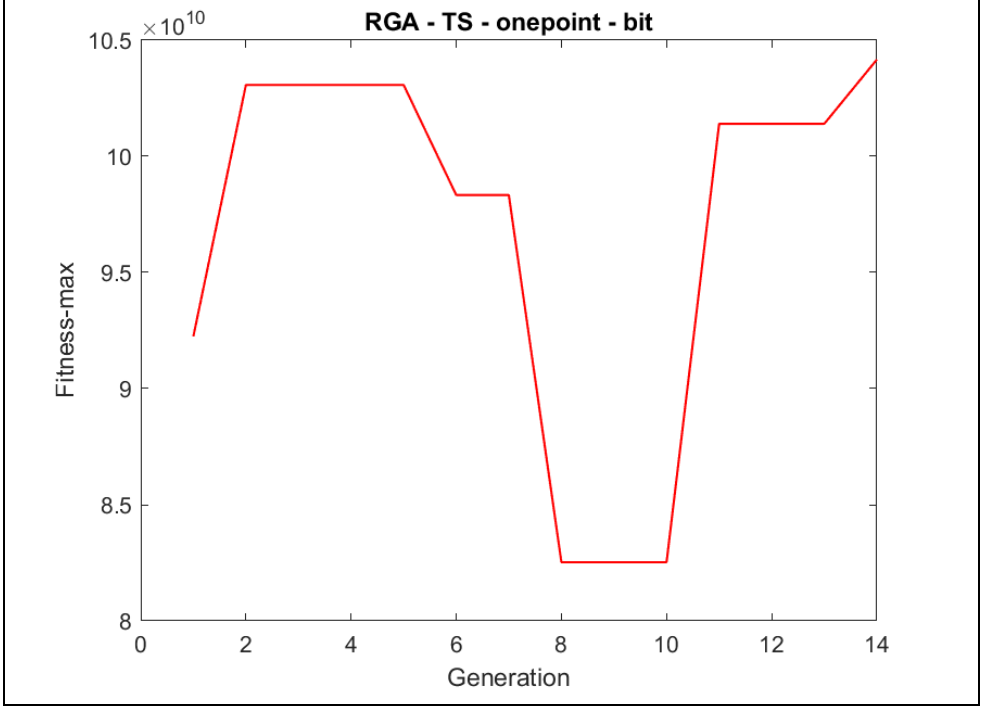
RGA_RWS_mask_mask



最好的 x = 1.6000

對應的 f(x) = 159.7889

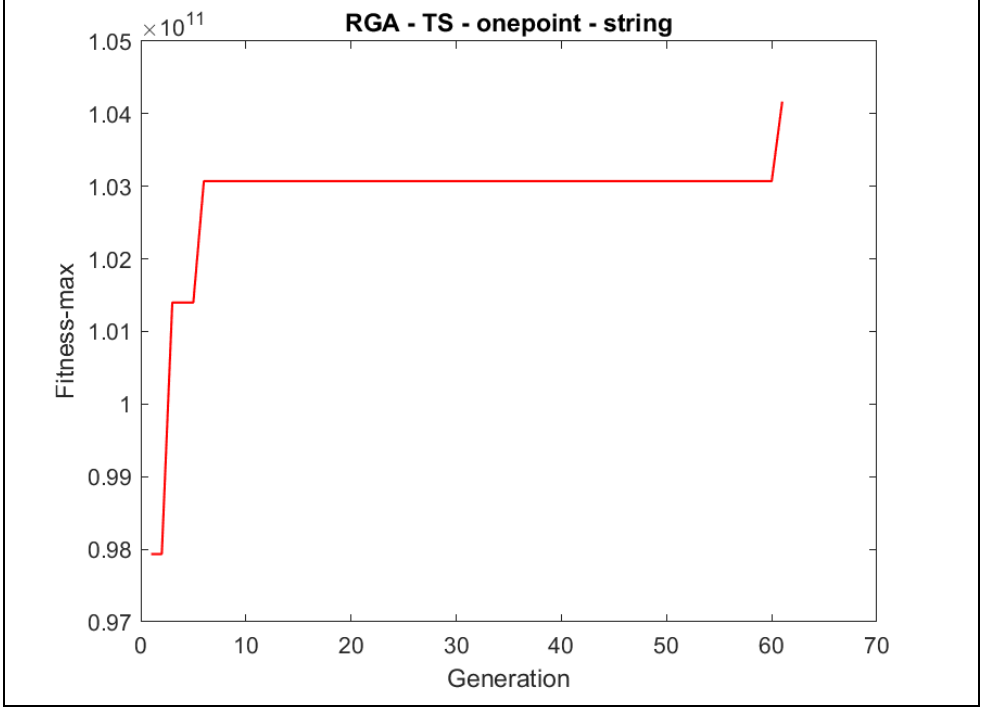
RGA_TS_onepoint_bit



最好的 x = 1.6000

對應的 f(x) = 159.7889

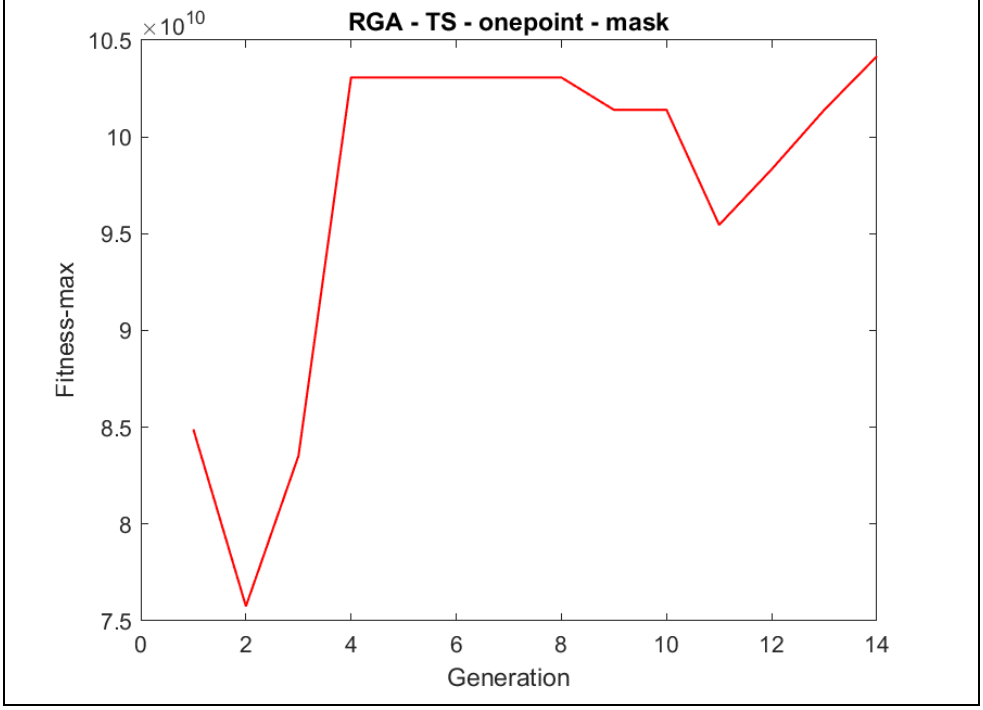
RGA_TS_onepoint_string



最好的 x = 1.6000

對應的 f(x) = 159.7889

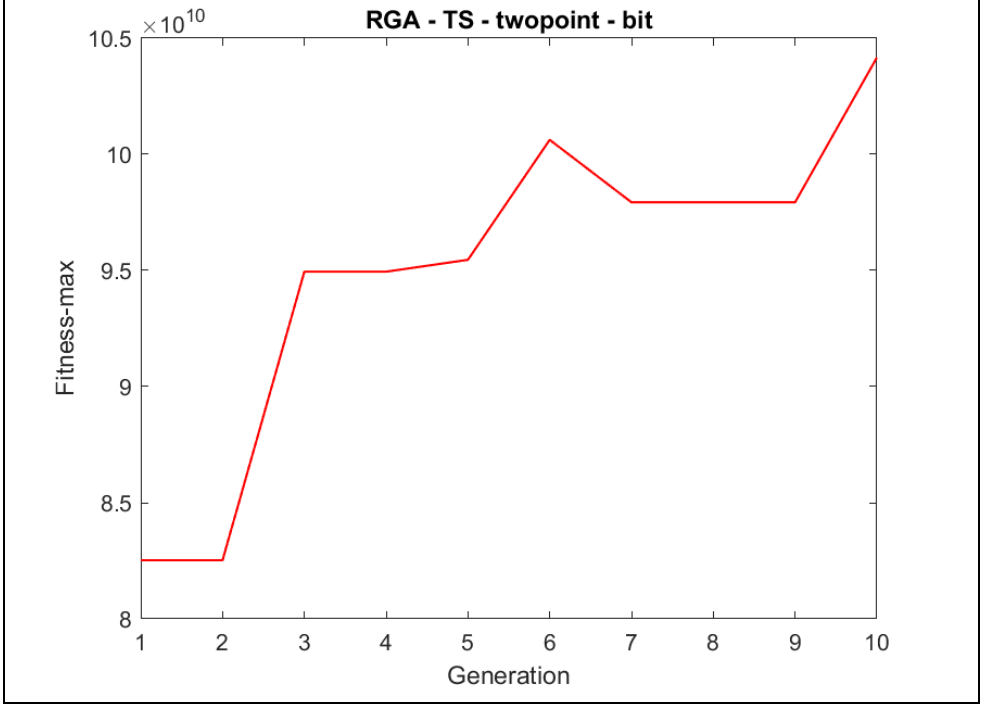
RGA_TS_onepoint_mask



最好的 x = 1.6000

對應的 f(x) = 159.7889

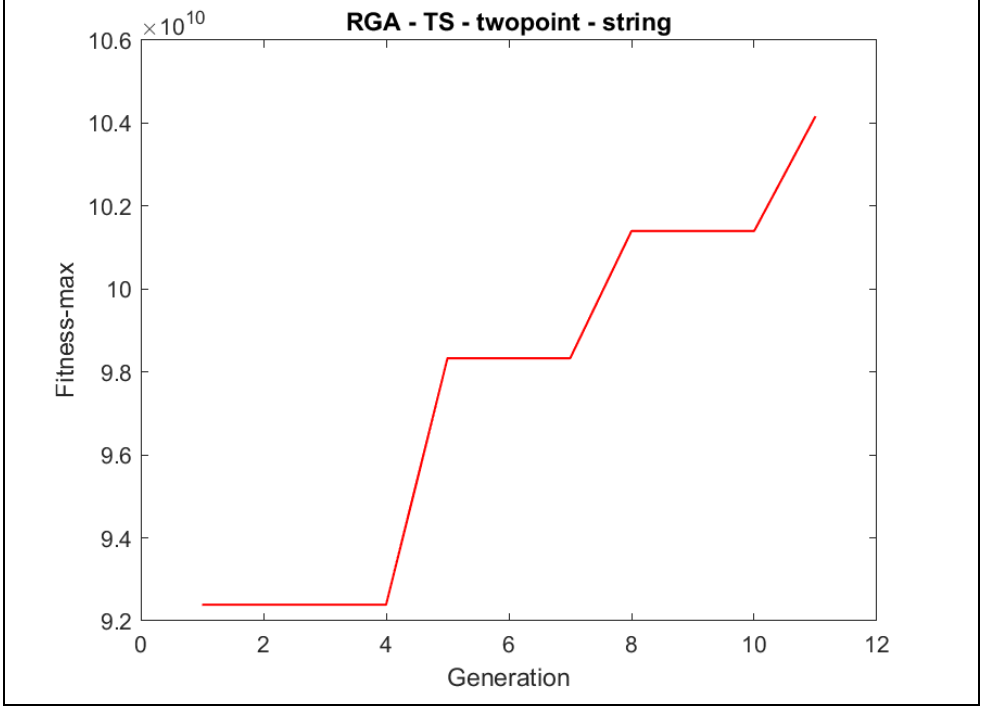
RGA_TS_twopoint_bit



最好的 x = 1.6000

對應的 f(x) = 159.7889

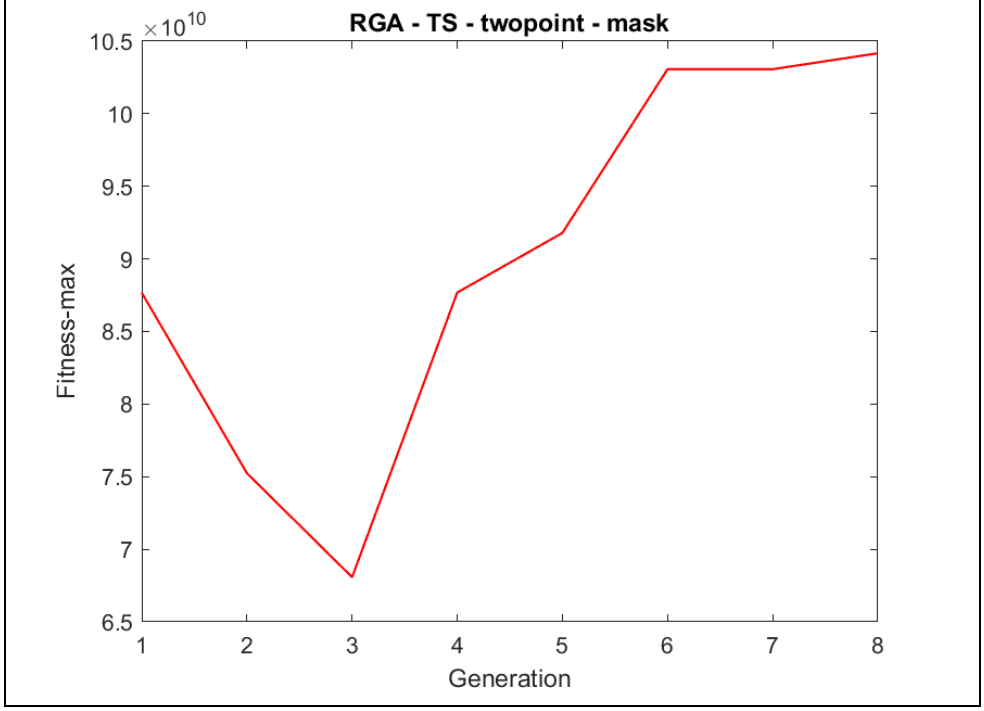
RGA_TS_twopoint_string



最好的 x = 1.6000

對應的 f(x) = 159.7889

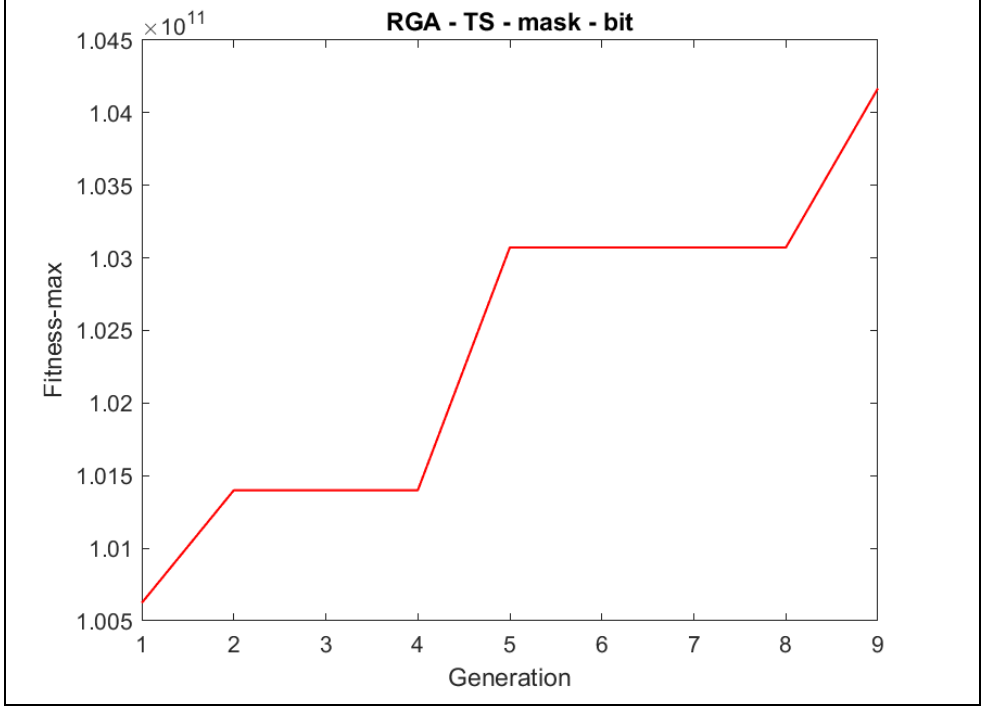
RGA_TS_twopoint_mask



最好的 x = 1.6000

對應的 f(x) = 159.7889

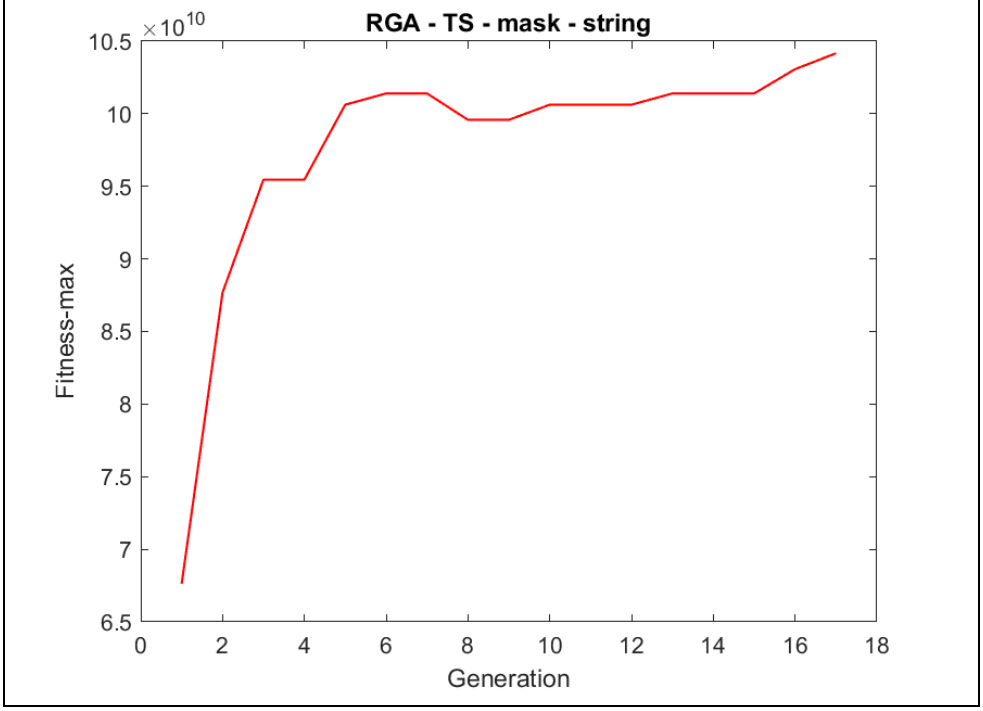
RGA_TS_mask_bit



最好的 x = 1.6000

對應的 f(x) = 159.7889

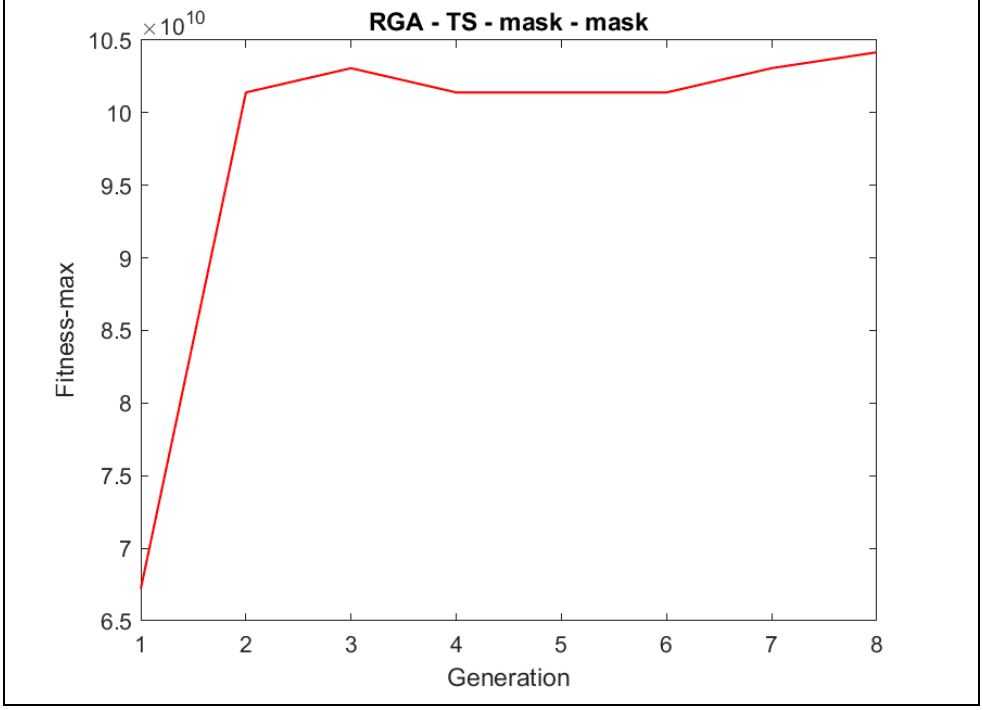
RGA_TS_mask_string



最好的 x = 1.6000

對應的 f(x) = 159.7889

RGA_TS_mask_mask

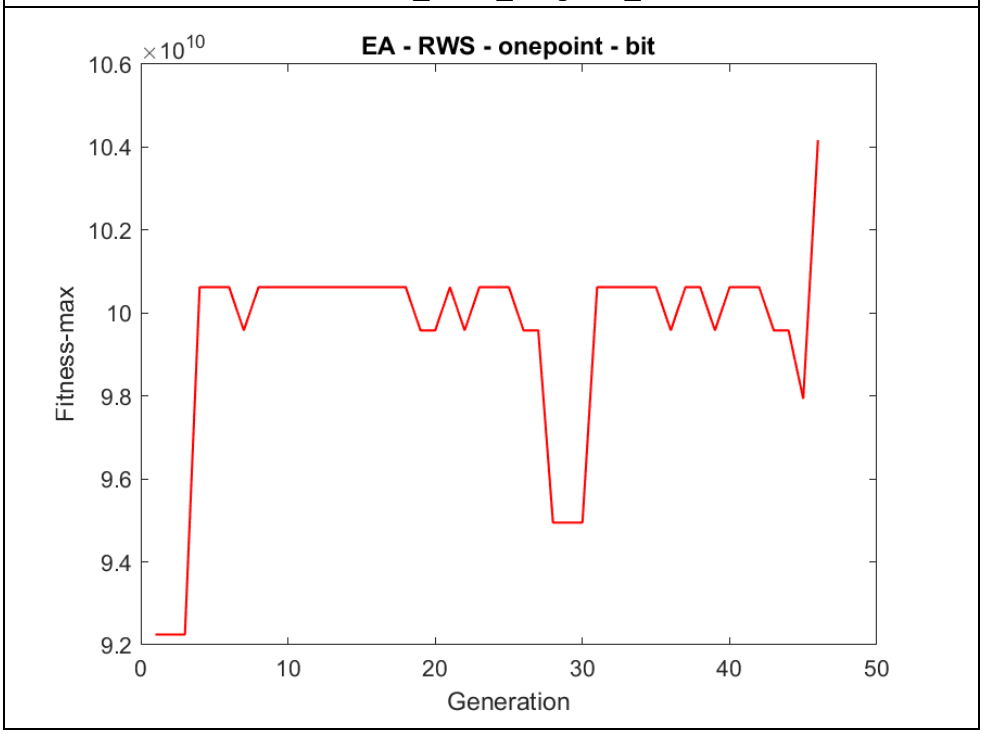


最好的 x = 1.6000

對應的 f(x) = 159.7889

4. EA

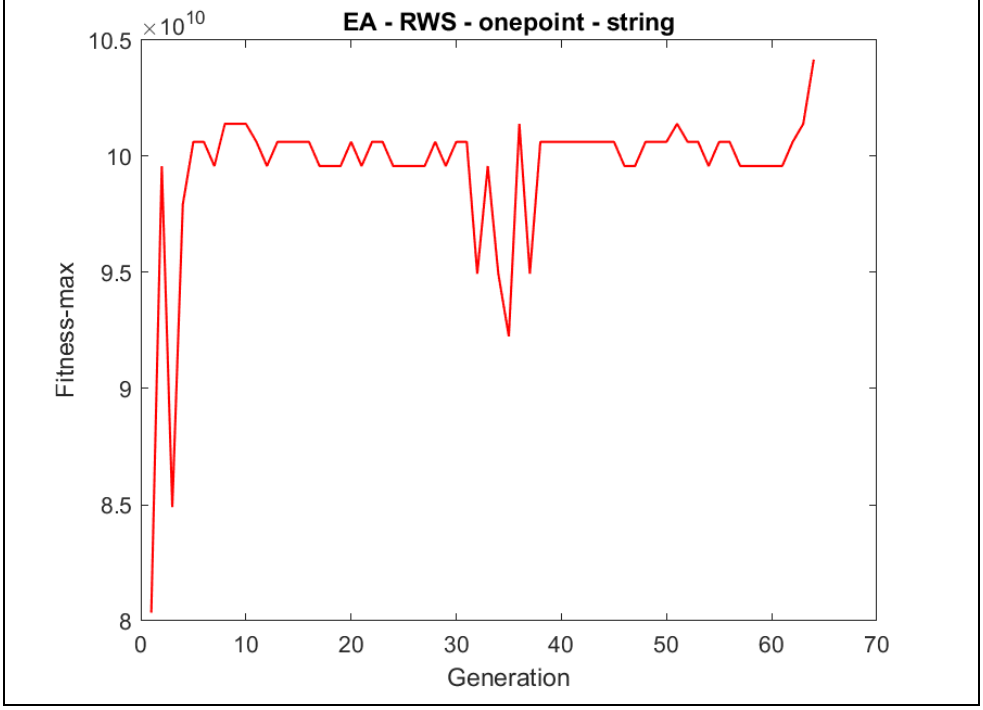
EA_RWS_onepoint_bit



最好的 x = 1.6000

對應的 f(x) = 159.7889

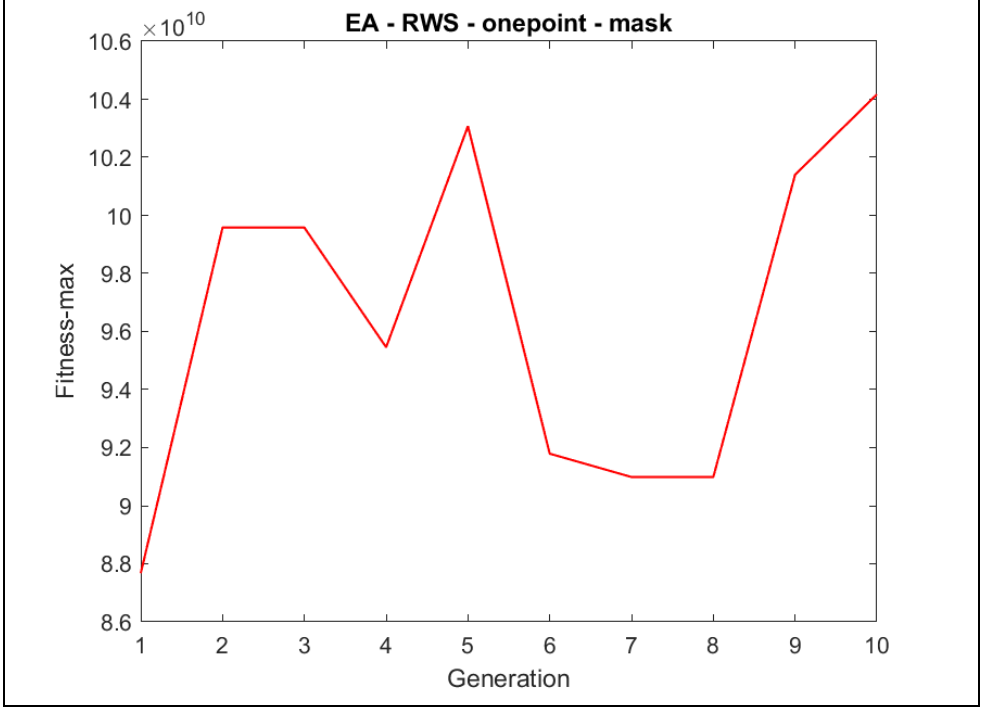
EA_RWS_onepoint_ string



最好的 x = 1.6000

對應的 f(x) = 159.7889

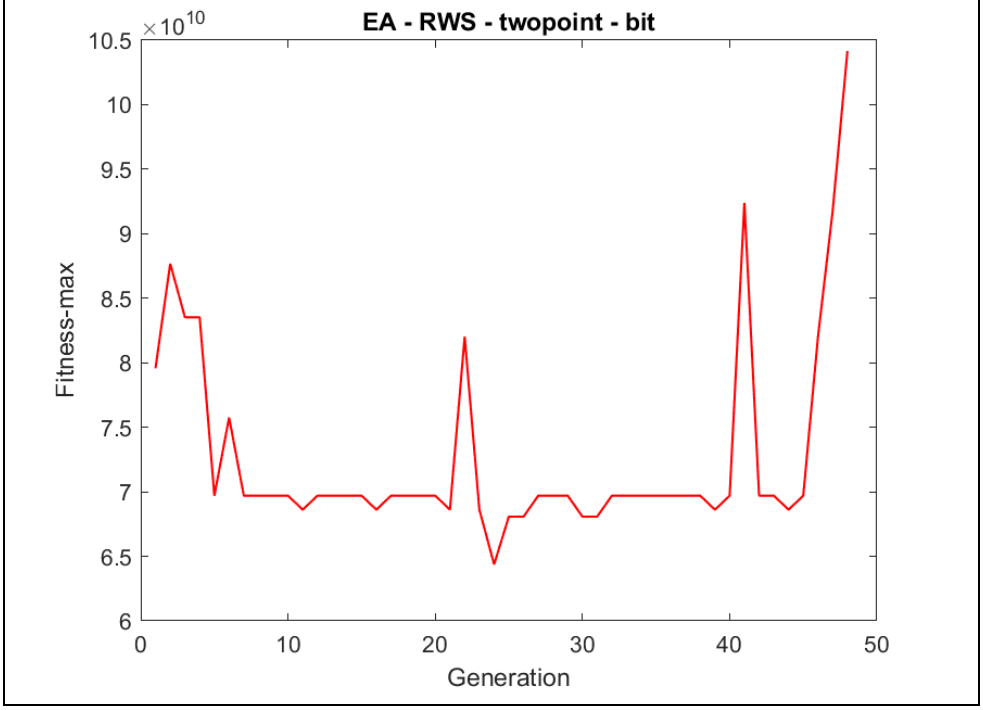
EA_RWS_onepoint_mask



最好的 x = 1.6000

對應的 f(x) = 159.7889

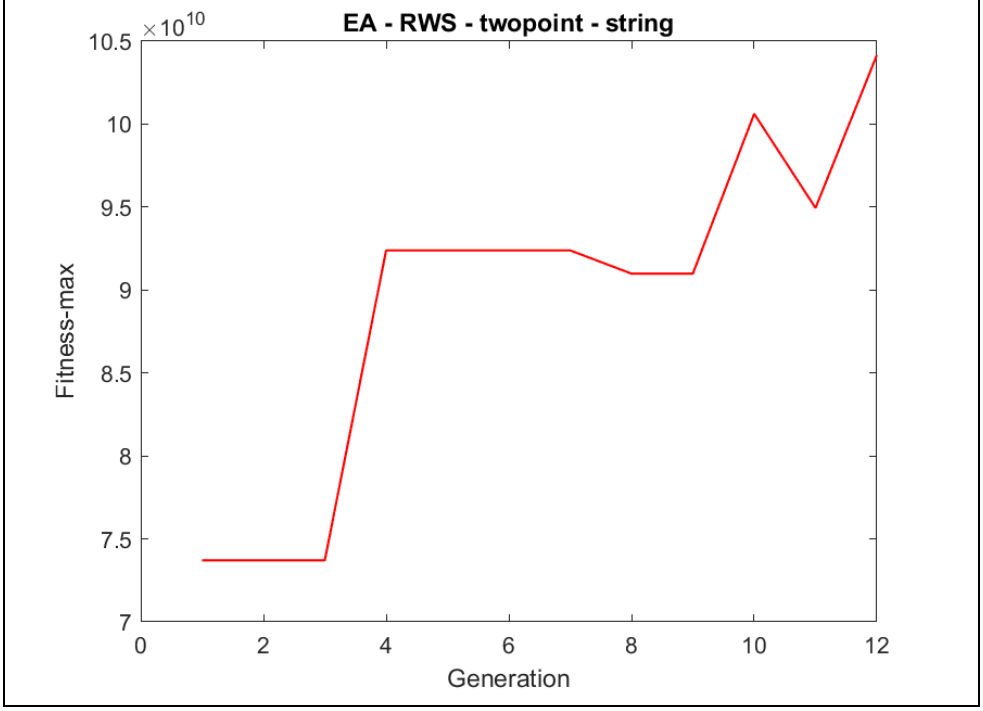
EA_RWS_twopoint_bit



最好的 x = 1.6000

對應的 f(x) = 159.7889

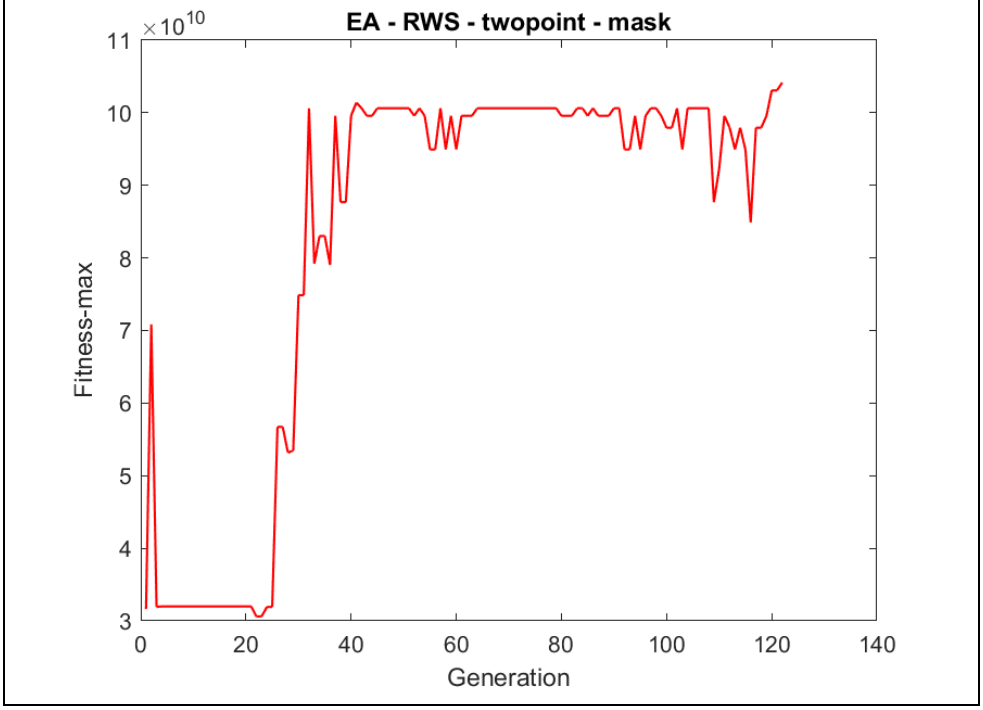
EA_RWS_twopoint_string



最好的 x = 1.6000

對應的 f(x) = 159.7889

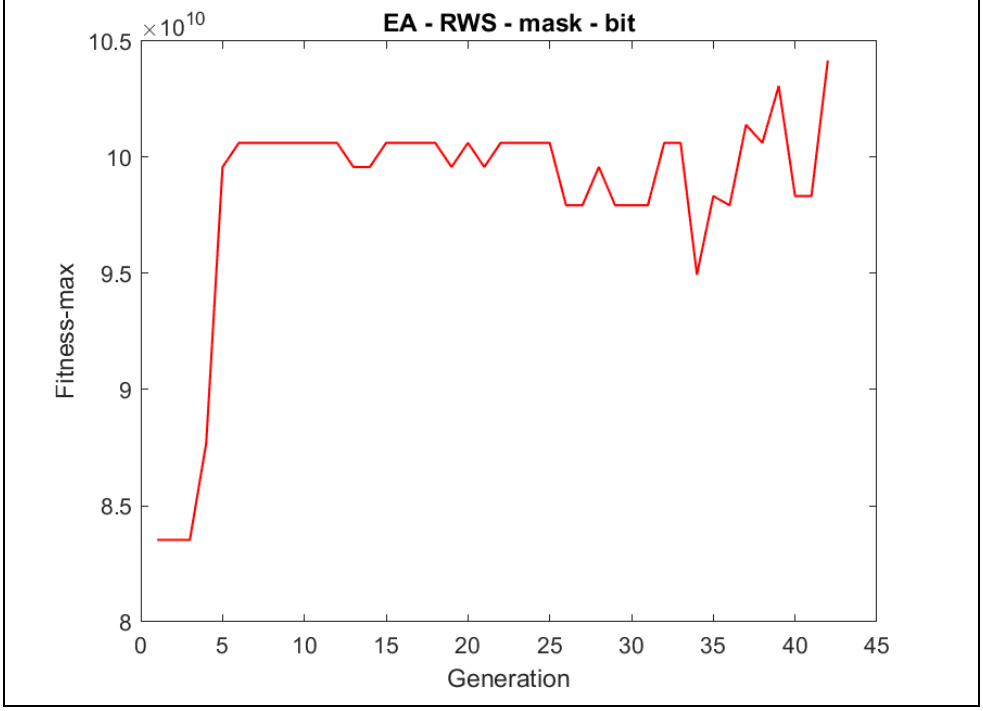
EA_RWS_twopoint_mask



最好的 x = 1.6000

對應的 f(x) = 159.7889

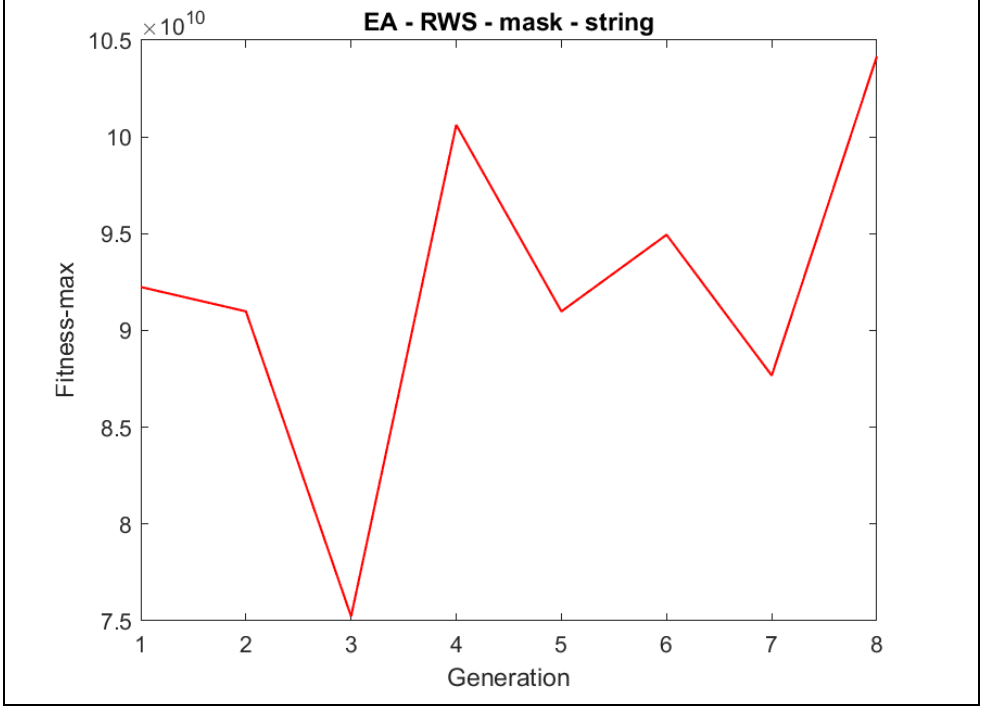
EA_RWS_mask_bit



最好的 x = 1.6000

對應的 f(x) = 159.7889

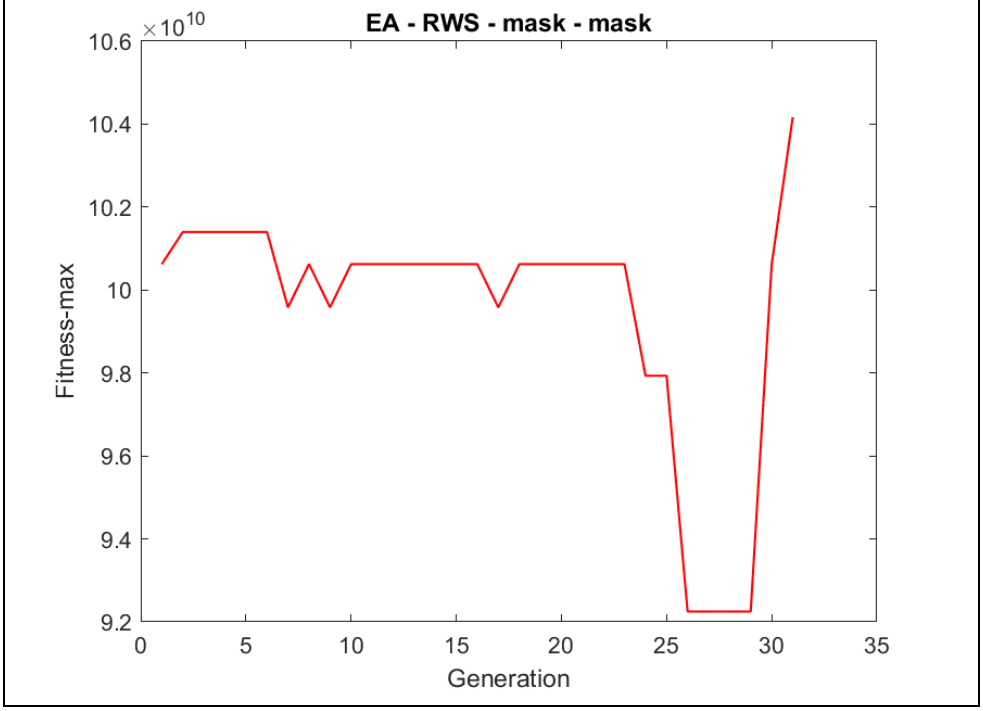
EA_RWS_mask_string



最好的 x = 1.6000

對應的 f(x) = 159.7889

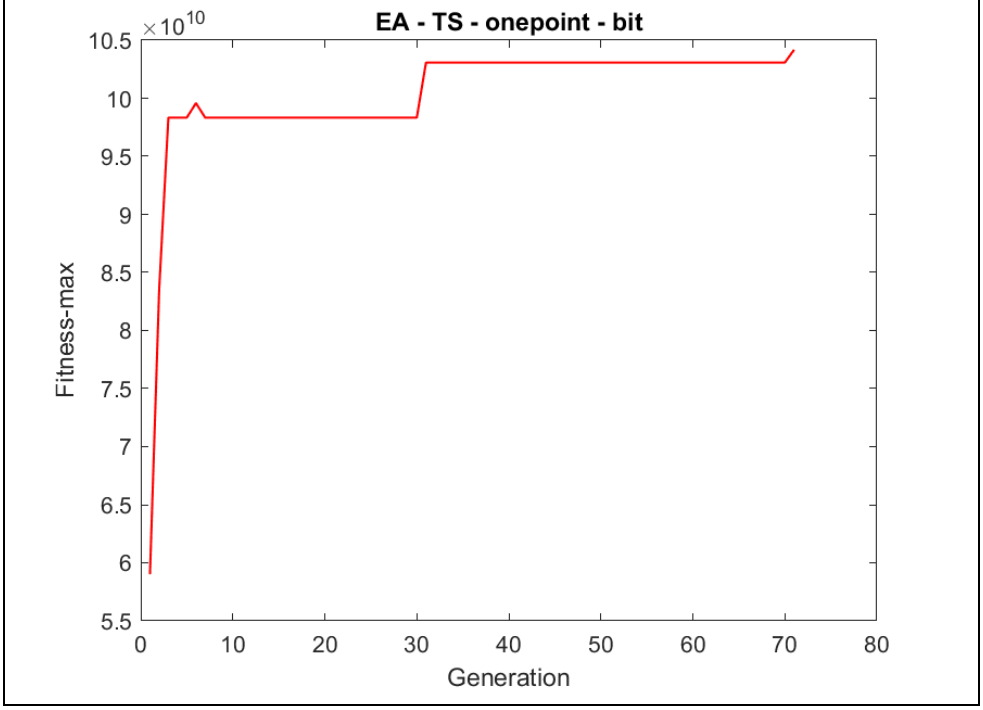
EA_RWS_mask_mask



最好的 x = 1.6000

對應的 f(x) = 159.7889

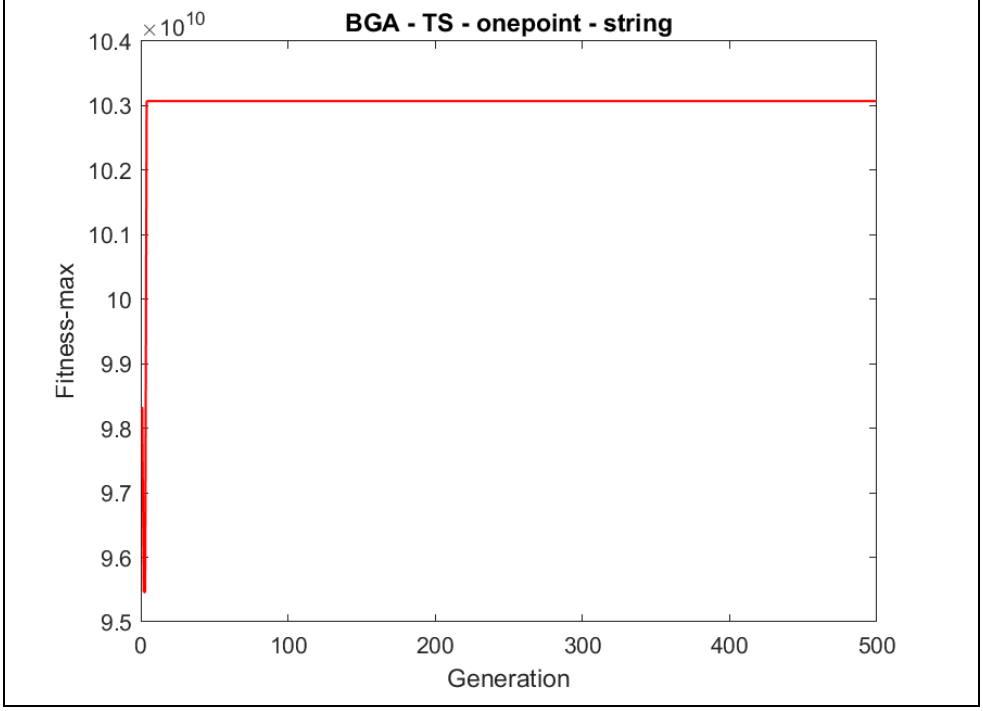
EA_TS_onepoint_bit



最好的 x = 1.6000

對應的 f(x) = 159.7889

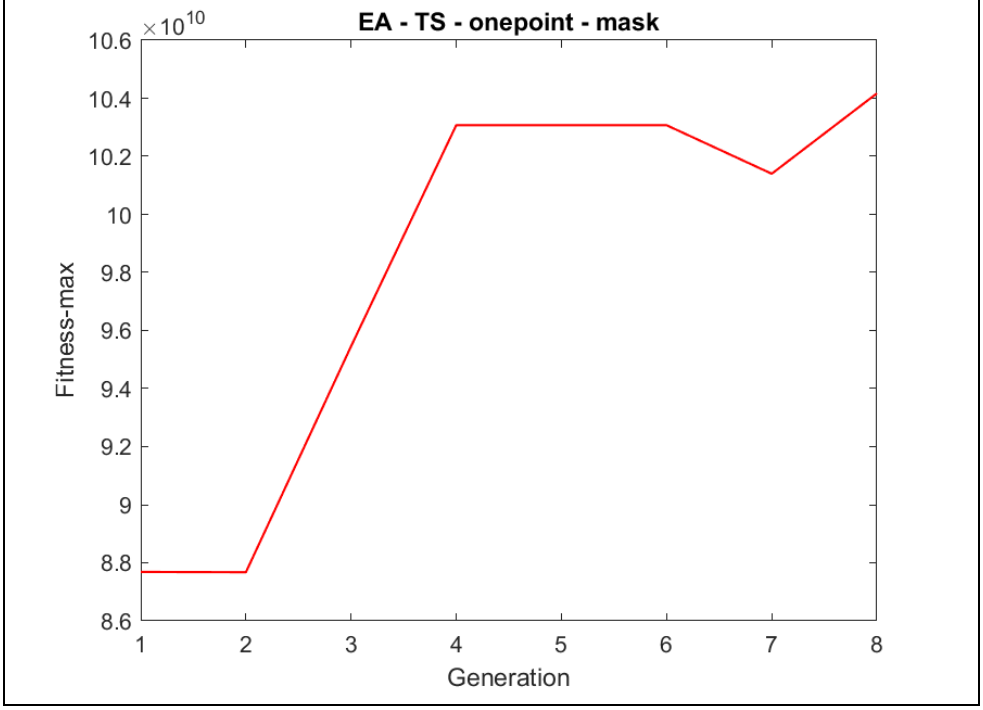
EA_TS_onepoint_string



最好的 x = 1.6000

對應的 f(x) = 159.7889

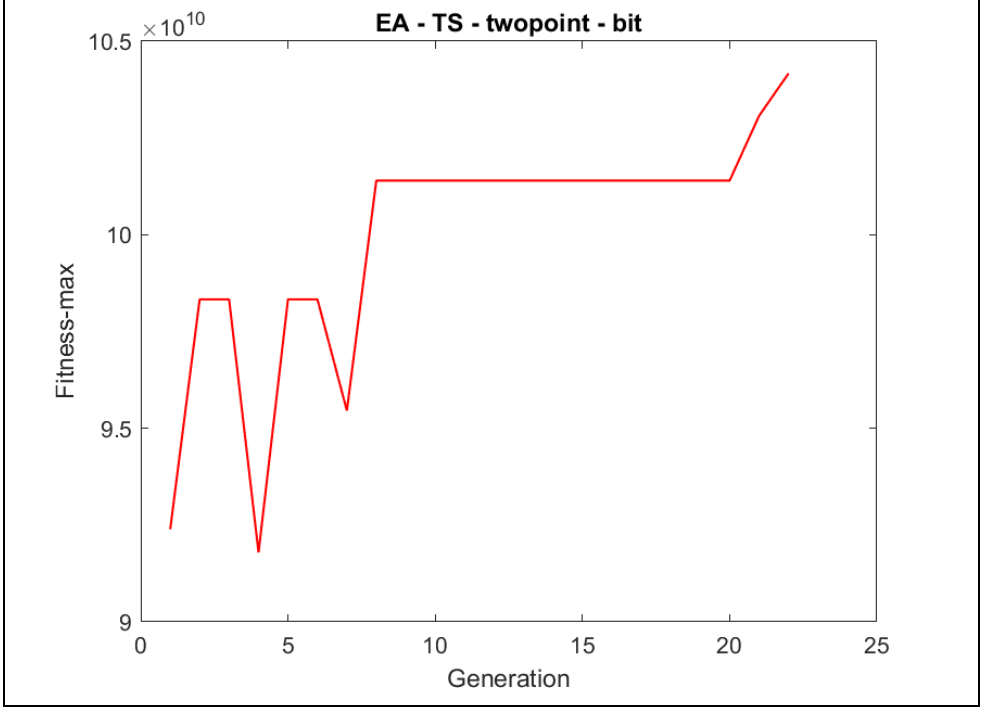
EA_TS_onepoint_mask



最好的 x = 1.6000

對應的 f(x) = 159.7889

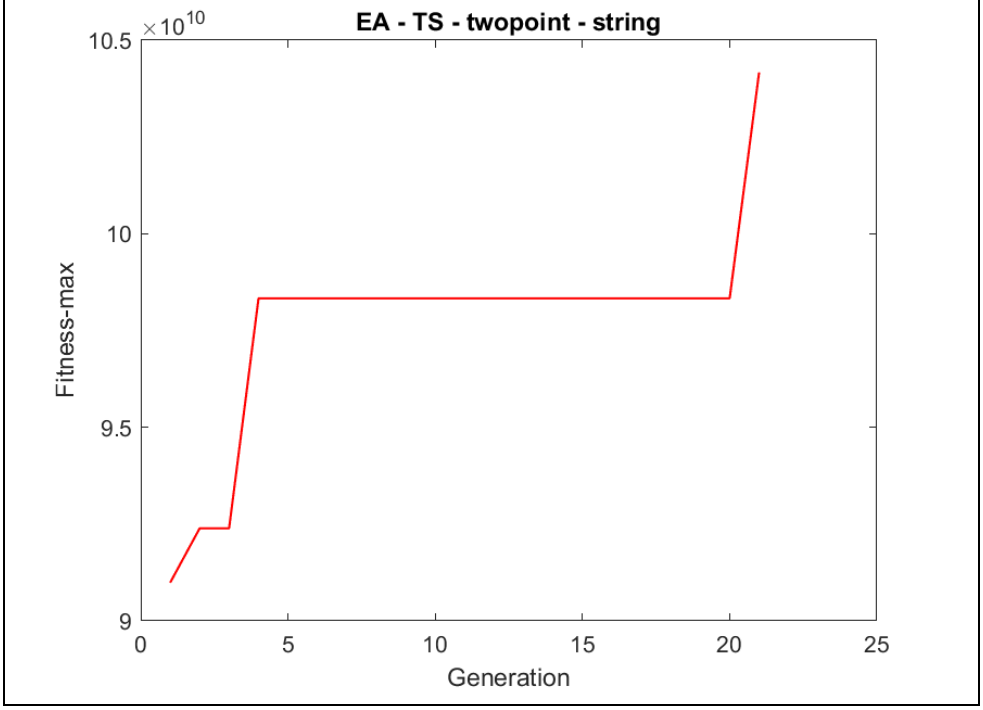
EA_TS_twopoint_bit



最好的 x = 1.6000

對應的 f(x) = 159.7889

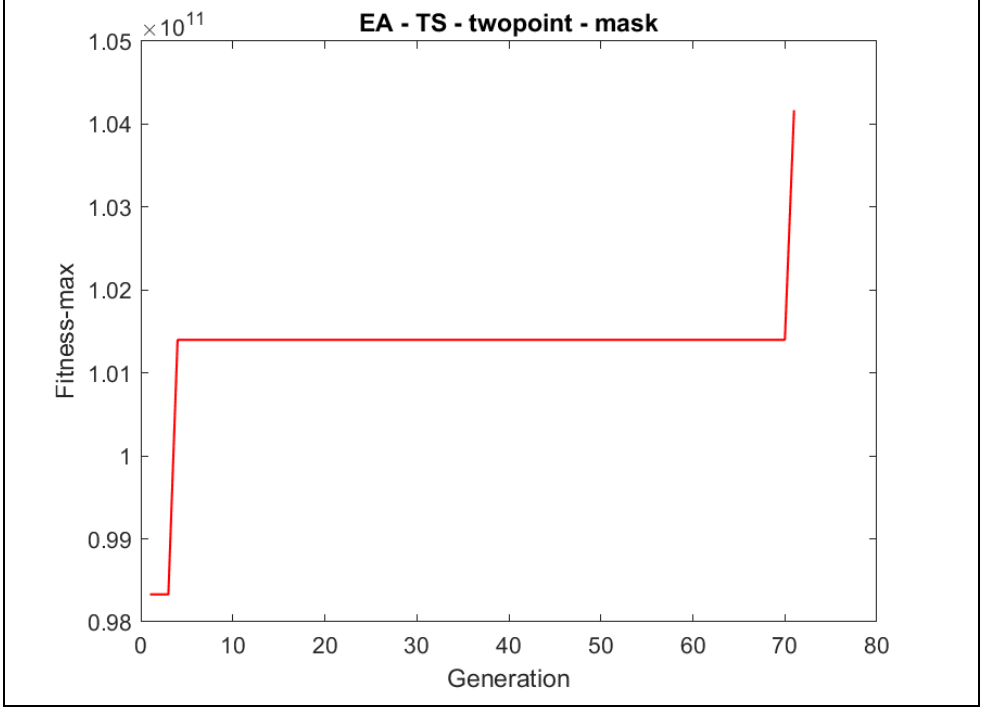
EA_TS_twopoint_string



最好的 x = 1.6000

對應的 f(x) = 159.7889

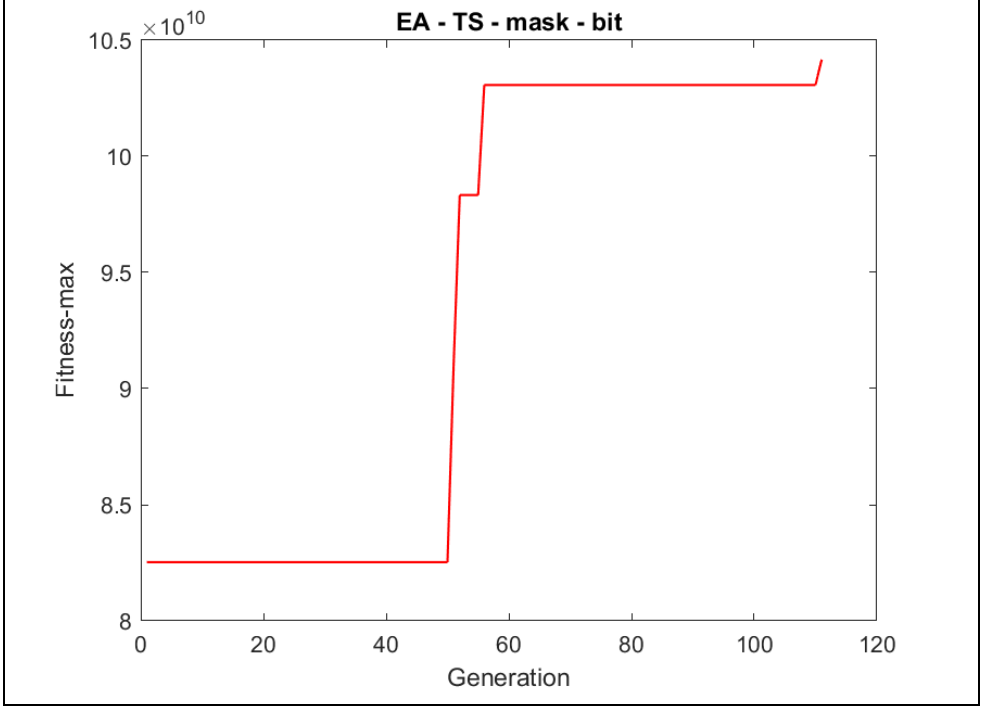
EA_TS_twopoint_mask



最好的 x = 1.6000

對應的 f(x) = 159.7889

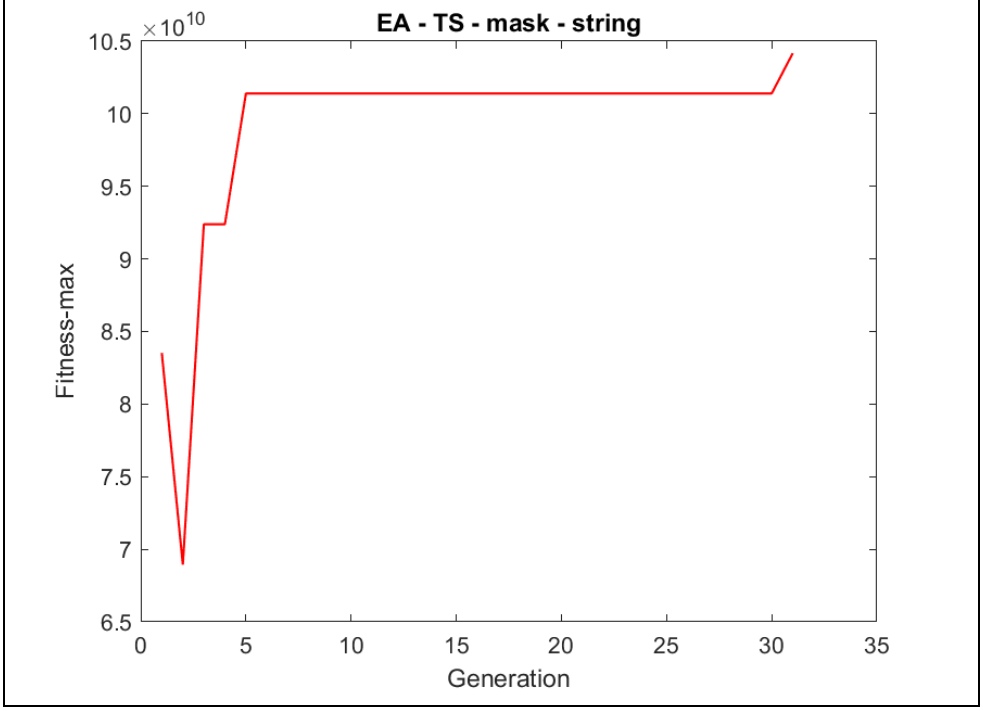
EA_TS_mask_bit



最好的 x = 1.6000

對應的 f(x) = 159.7889

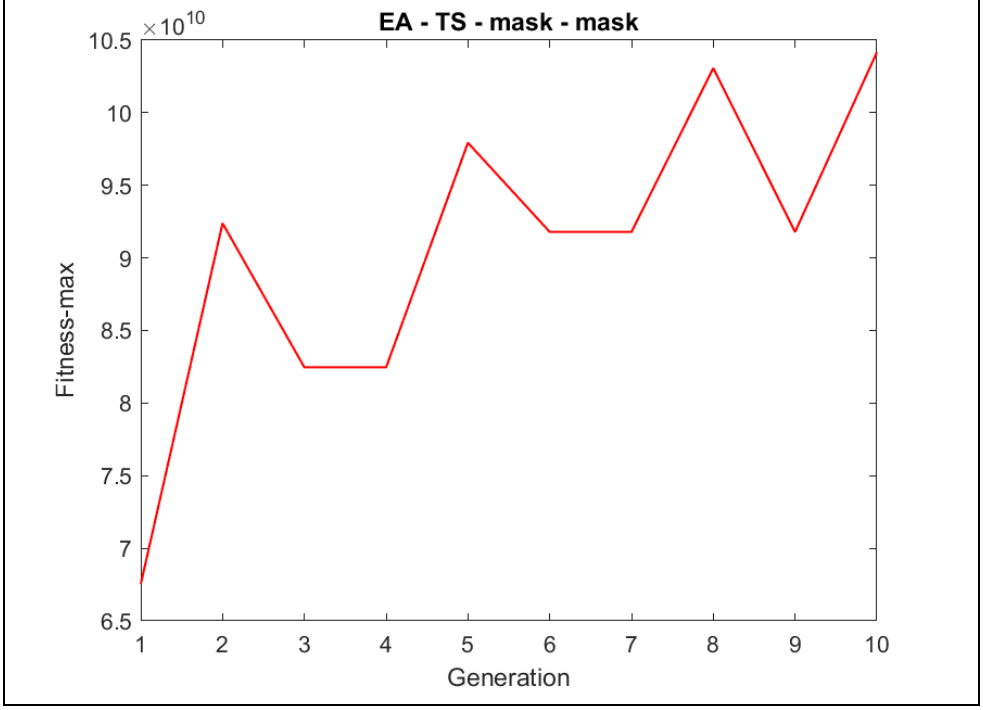
EA_TS_mask_string



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

四、 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。