**ĐẠI HỌC QUỐC GIA TP. HỒ CHÍ MINH**

TRƯỜNG ĐẠI HỌC BÁCH KHOA

KHOA ĐIỆN – ĐIỆN TỬ

**BỘ MÔN TỰ ĐỘNG**

---------------o0o---------------

****

**BÁO CÁO BÀI TẬP NHÓM**

**THUẬT TOÁN DI TRUYỀN**

**GVHD:**

**SVTH:**

1. **Mai Thiện Quang – 1512640**
2. **Nguyễn Trọng Phúc – 1512534**
3. **Lương Hữu Phú Lộc –**
4. **Bùi Tấn Phát – 151**

**Đề bài:** Dùng thuật toán di truyền tìm x є [0; 5] sao cho hàm số

đạt giá trị lớn nhất.

# **I. Giải thuật di truyền dùng code Pyton:**

## **1. Giải thuật di truyền**

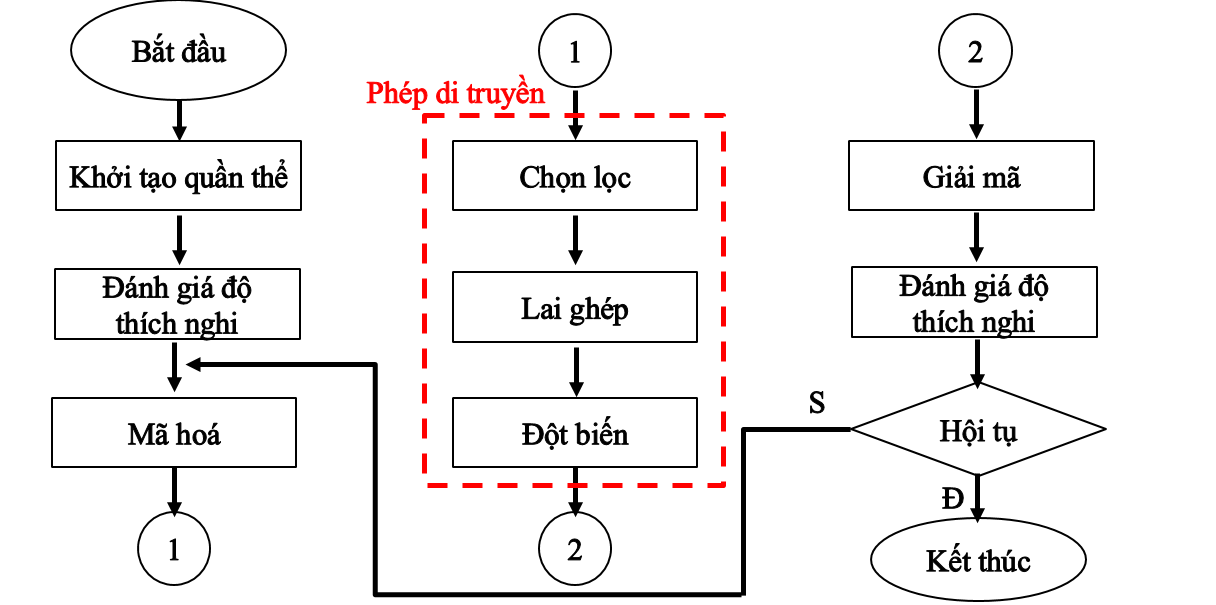
- Thông số khởi tạo:

+ Số lượng cá thể trong quần thể: 20

+ Số bit dùng để mã hóa: 10 bit

+ Tỉ lệ đột biến: 1%

- Thực hiện giải thuật:



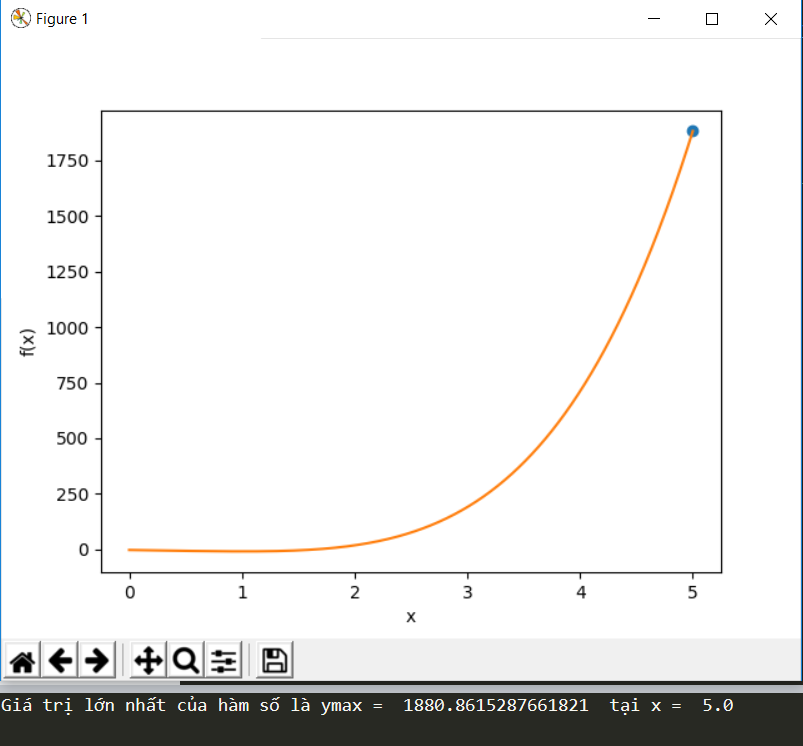
+ Giải thích về phép di truyền:

* Chọn lọc:
* Đầu tiên giữ lại ½ số lượng cá thể tốt nhất có trong quần thể (10 cá thể).
* Từ 10 cá thể này lựa chọn ngẫu nhiên để có được quần thể có 20 cá thể. Thuật toán như sau:
* Tạo ra 1 số nhiên x có giá trị từ 0.0 đến 1.0
* Sử dụng hàm **numpy.histogram**(x, [0.0, b, c, d, e, …, 1.0] ): x là giá trị ngẫu nhiên được tạo ở trên, tham số mảng là 1 mảng có 11 (tạo thành 10 khoảng ứng với 10 chỉ số) phần tử có giá trị tăng dần từ 0.0 đến 1.0. Chọn các phần tử của mảng sao cho khoảng cách liên tiếp của 2 phần tử trong mảng có chỉ số thấp sẽ có độ lớn lớn hơn 2 phần tử tiếp theo có chỉ số cao hơn. Hàm **numpy.histogram**() sẽ so sánh giá trị x nằm trong khoảng nào để trả về mảng có giá trị 1 tại vị trí của khoảng đó. Ví dụ: **count=numpy.histogram**(0.1, [0.0, 0.21, 0.39, 0.54, 0.67, 0.77, 0.84, 0.91, 0.94, 0.97, 1.0]) sẽ trả về mảng [1, 0, 0, 0, 0, 0, 0, 0, 0, 0]. Sau đó sử dụng hàm **indx=numpy.where**(count==1), hàm này sẽ trả về vị trí của phần tử có giá trị 1 trong mảng: indx=0. Lặp các bước trên 20 lần ta sẽ có được 20 chỉ số. Sử dụng các chỉ số trên để lấy ra cá thể thuộc 10 cá thể còn lại, ta sẽ có được quần thể mới gồm 20 cá thể.
* Thuật toán trên vì từ quần thể 20 cá thể ban đầu ta đã chọn ra được 10 cá thể tốt nhất và sắp xếp chúng với độ phù hợp giảm dần, do đó cá thể có độ phù hợp càng lớn thì chỉ số càng thấp. Tham số mảng truyền vào hàm **numpy.histogram**() có tính chất như trên để tăng xác suất giá trị ngẫu nhiên x rơi vào vùng có chỉ số thấp hay nói cách khác cá thể có độ phù hợp càng lớn thì tần số xuất hiện trong quần thể mới càng cao.
* Lai ghép: quần thể có 20 cá thể vừa được tạo ra ở bước trên, ta chọn ra 10 cặp cá thể để lai ghép tạo ra 20 cá thể mới.
* Tạo đột biến với tỉ lệ là 1% tổng số gen của quần thể ta được quần thể mới.

+ Điều kiện hội tụ: quần thể trải qua 60 thế hệ hoặc có 20 thế hệ liên tiếp cho ra độ phù hợp sai lệch nhỏ hơn 0.000005.

## **2. Đồ thị hàm số f(x) = 4x4 – 5x3 + e-2x – 7sin(x) – 3cos(x) với x є [0; 5]**

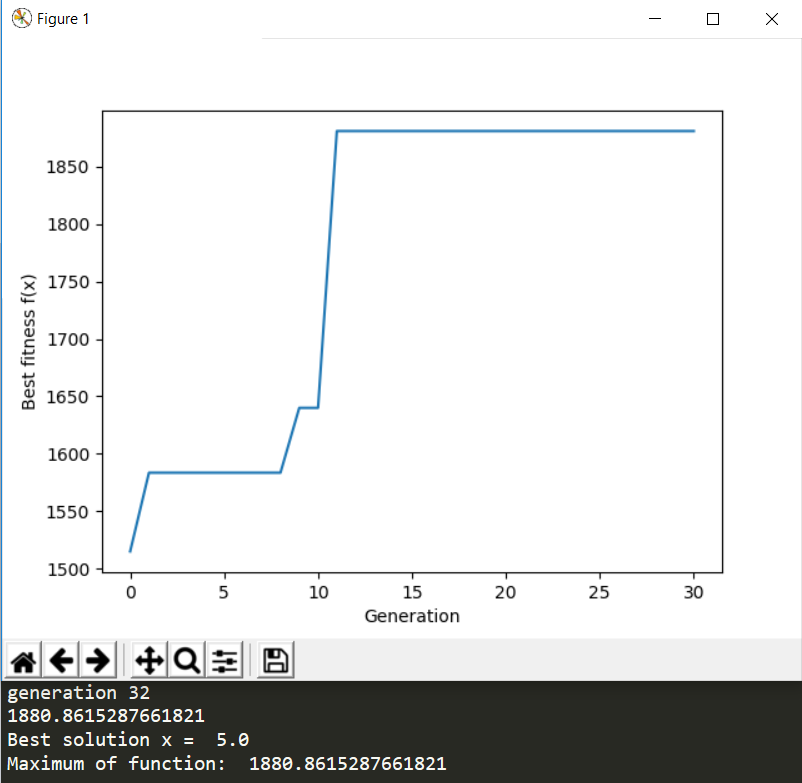
Hàm số đạt giá trị lớn nhất là **ymax = 1880.8615287661821** tại **x = 5.0**



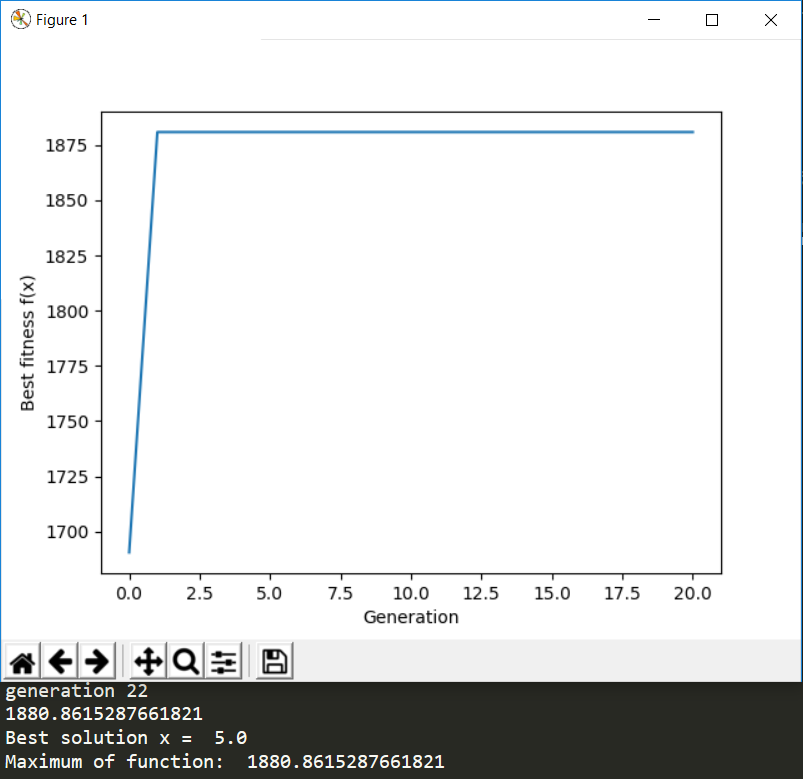
## **3. Chương trình thực hiện giải thuật bằng ngôn ngữ Python**

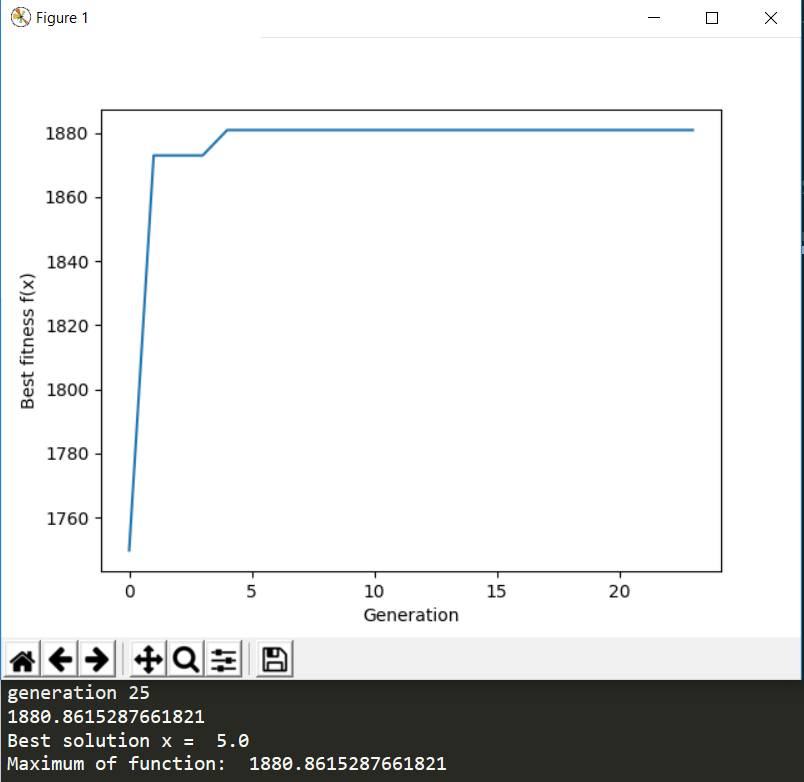
Tập tin HW1\_G1\_Code1 và tập tin HW1\_G1\_Code2 đính kèm.

## **4. Kết quả bài toán thực hiện theo giải thuật di truyền.**

 - Lần 1: Hàm số đạt giá trị lớn nhất là **ymax = 1880.8615287661821** tại **x = 5.0**. Thuật toán hội tụ sau **32** thế hệ.

- Lần 2: Hàm số đạt giá trị lớn nhất là **ymax = 1880.8615287661821** tại **x = 5.0**. Thuật toán hội tụ sau **22** thế hệ.



 - Lần 3: Hàm số đạt giá trị lớn nhất là **ymax = 1880.8615287661821** tại **x = 5.0**. Thuật toán hội tụ sau **25** thế hệ.

## **4. Nhận xét**

- Thuật toán có thể khác nhau về số lượng thế hệ trải qua, có thể ít hoặc nhiều nhưng nhìn chung vẫn đưa ra cùng 1 kết quả chính xác.

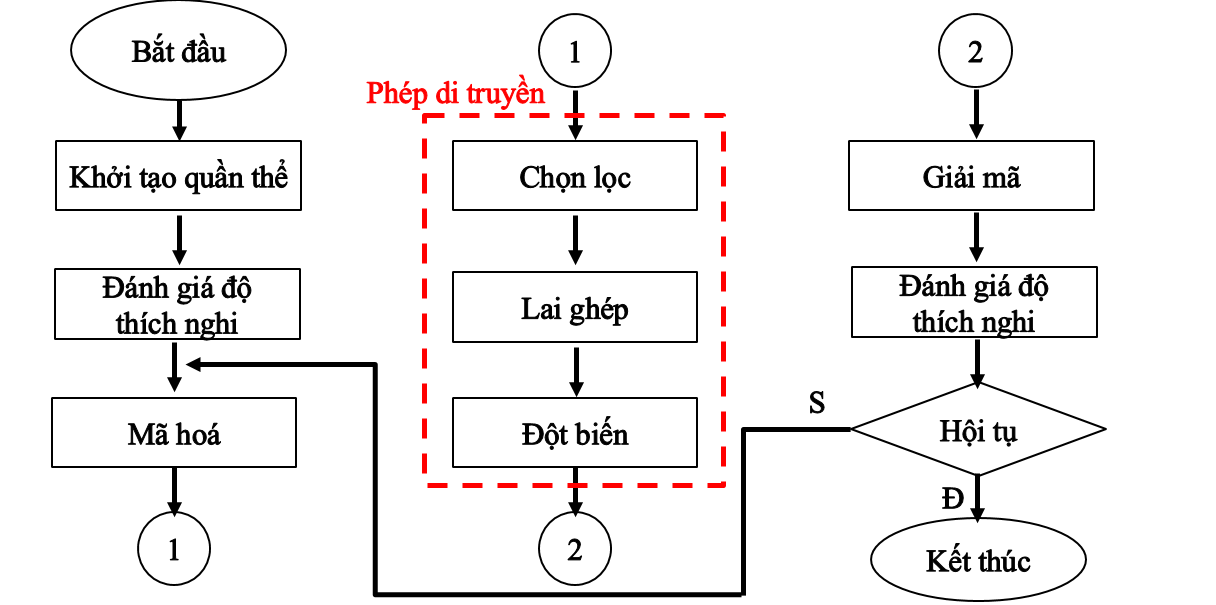
- Đồ thị biểu thị độ phù hợp tốt nhất qua từng thế hệ, ta thấy đồ thị vẫn còn thay đổi nhiều và chưa phẳng, tuy nhiên càng về sau thì đồ thị có xu hướng phẳng hơn và bám dần vào giá trị ymax của hàm số.

- Vì hàm số này là hàm đồng biến nên ymax tại x = 5.0, do đó khá dễ tìm nên do đó ta chưa thấy sự hiệu quả của thuật toán.

# **II. Tìm giá trị max bằng thuật toán GA sử dụng Matlab:**

## **Giải thuật**

* Thông số khởi tạo:
* Vùng khảo sát: [0,5]
* Hàm số khảo sát:
* Số lượng cá thể trong quần thể : 40
* Số lượng bit mã hóa: 10 bit 🡪 1024 số
* Tỉ lệ đột biến: 1%
* Thực hiện thuật toán GA: về cơ bản giống với lý thuyết đã học.



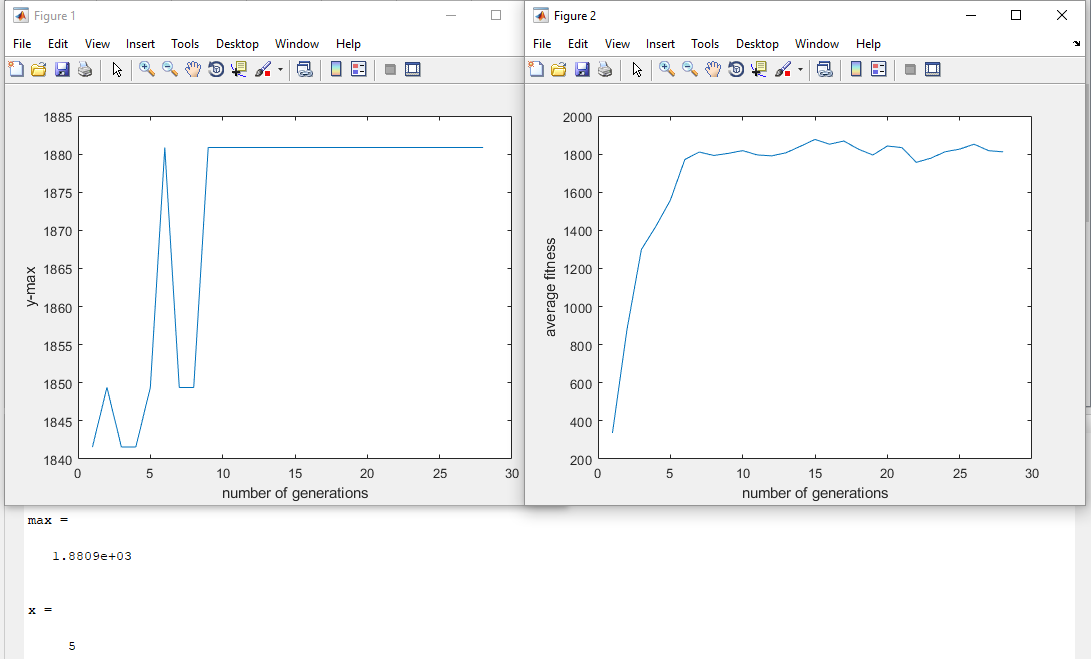
Có một số đặc điểm trong phần bài làm Matlab:

* Chọn lọc: chọn một nửa cá thể tốt và bỏ một nửa cá thể xấu.
* Lai ghép: bố và mẹ được chọn ngẫu nhiên từ các cá thể đã qua chọn lọc.
* Tiêu chuẩn xét hội tụ: Ở mỗi thế hệ, lưu lại giá trị max. Tính độ lệch chuẩn của 20 giá trị max của 20 thế hệ gần nhất, nếu nhỏ hơn 0.001 🡺 tìm được giá trị max của hàm số. (sự hội tụ của các giá trị max của 20 thế hệ gần nhất).

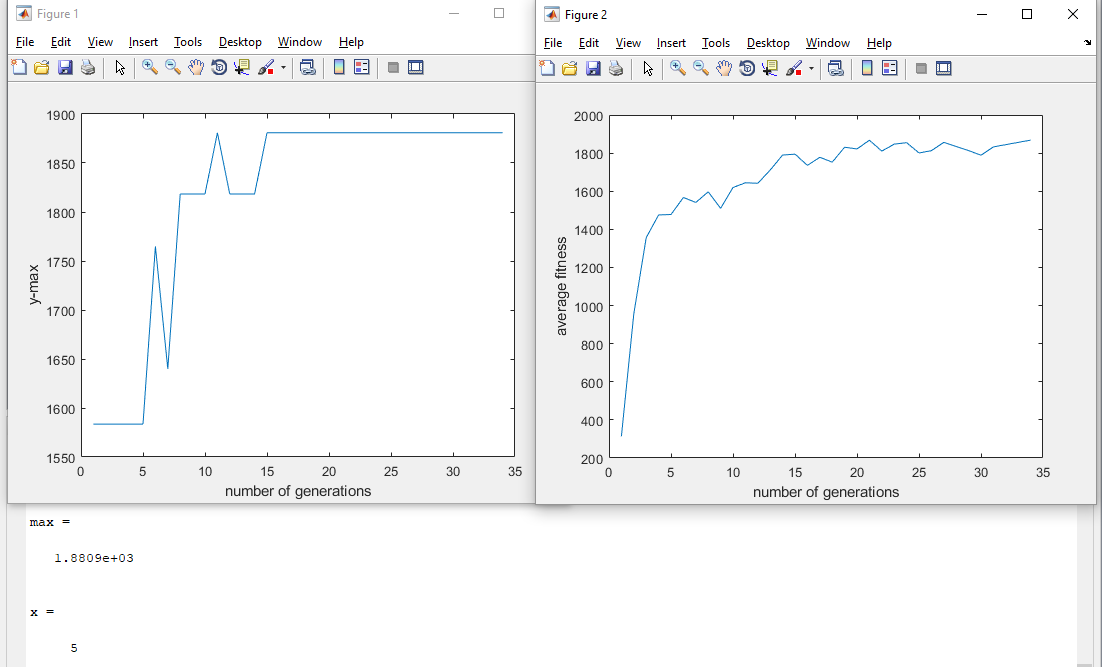
## **Code Matlab (Trong file đính kém)**

## **Kết quả mô phỏng:**

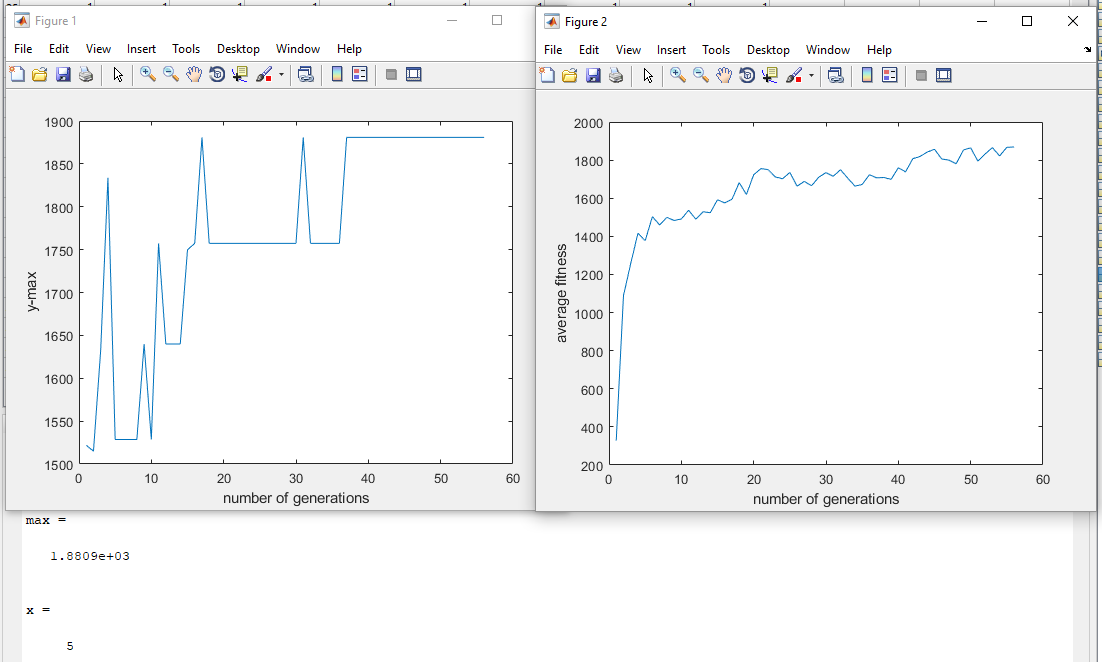
* 2 đồ thị thể hiện giá trị max qua mỗi thế hệ và giá trị trung bình của hàm số qua mỗi thế hệ
* Đồ thị thực của hàm số 🡪 max tại x=5.
* Lần 1:



* Lần 2:



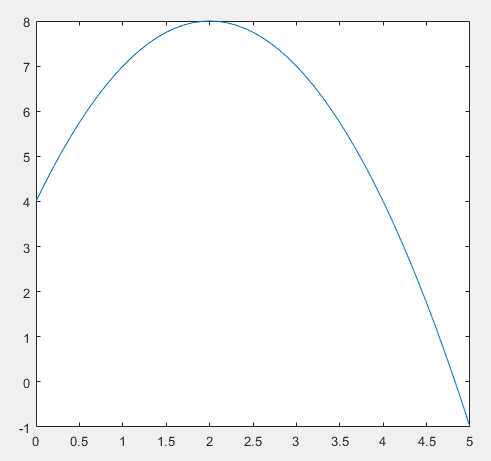
* Lần 3:

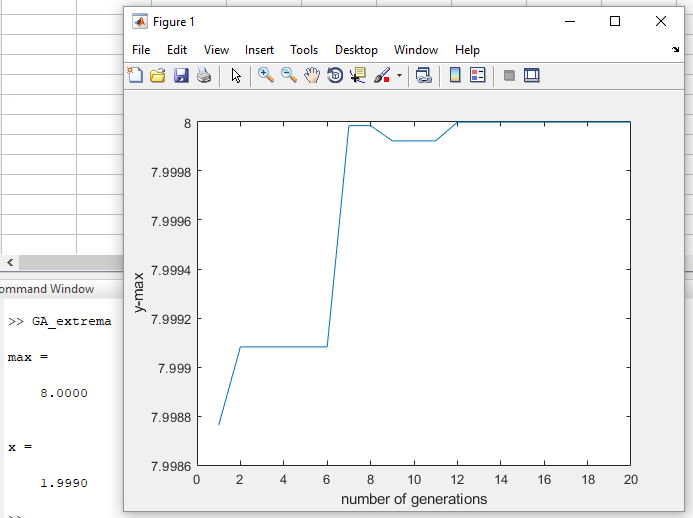


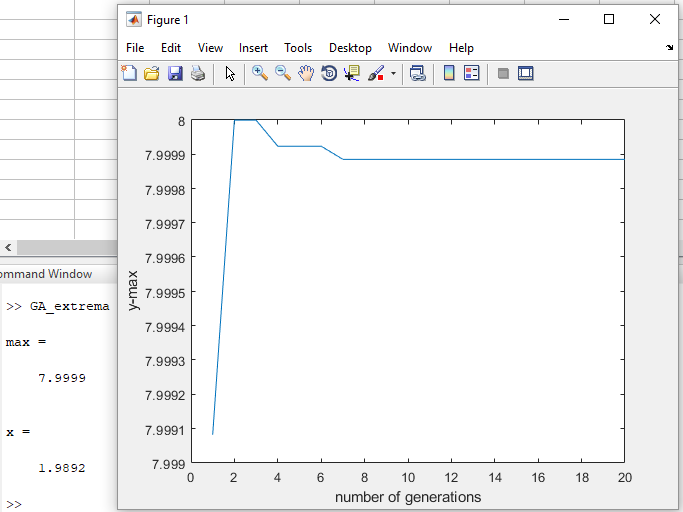
* Nhận xét:
* Thuật toán có thể khác nhau về số lượng thế hệ trải qua, có thể ít hoặc nhiều nhưng nhìn chung vẫn đưa ra cùng 1 kết quả chính xác.
* Đồ thị bên phải biểu thị fitness trung bình, ta thấy đồ thị vẫn còn thay đổi nhiều và chưa phẳng, tuy nhiên càng về sau thì đồ thị có xu hướng phẳng hơn và bám dần vào giá trị max của hàm số.
* Vì hàm số này là hàm đồng biến nên max tại x = 5.0, do đó khá dễ tìm nên do đó ta chưa thấy sự hiệu quả của thuật toán.

## **Ví dụ khác**

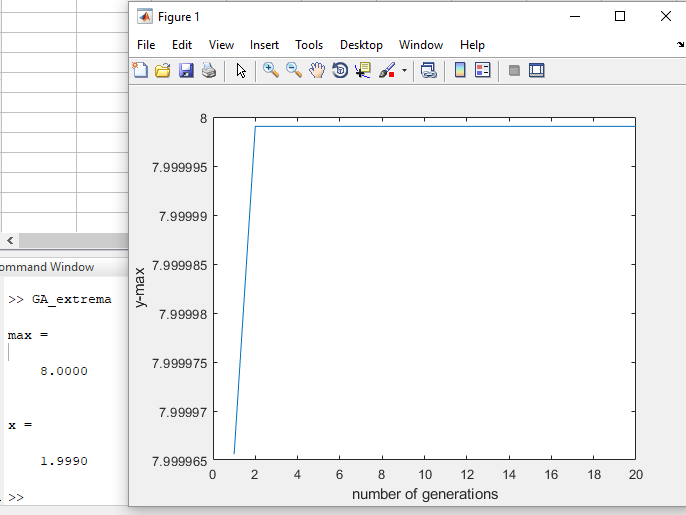
* Tìm giá trị lớn nhất của hàm trong khoảng [0,5]
* Hàm số thực và kết quả: max(f(x)) = 8 tại x = 2.0



* Lần 1: max = 8.0 tại x = 1.999
* Lần 2: max = 7.9999 tại x = 1.9892



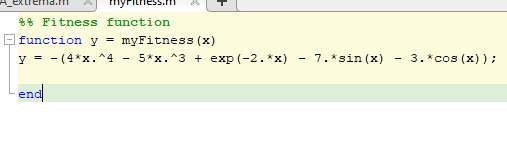
* Lần 3:



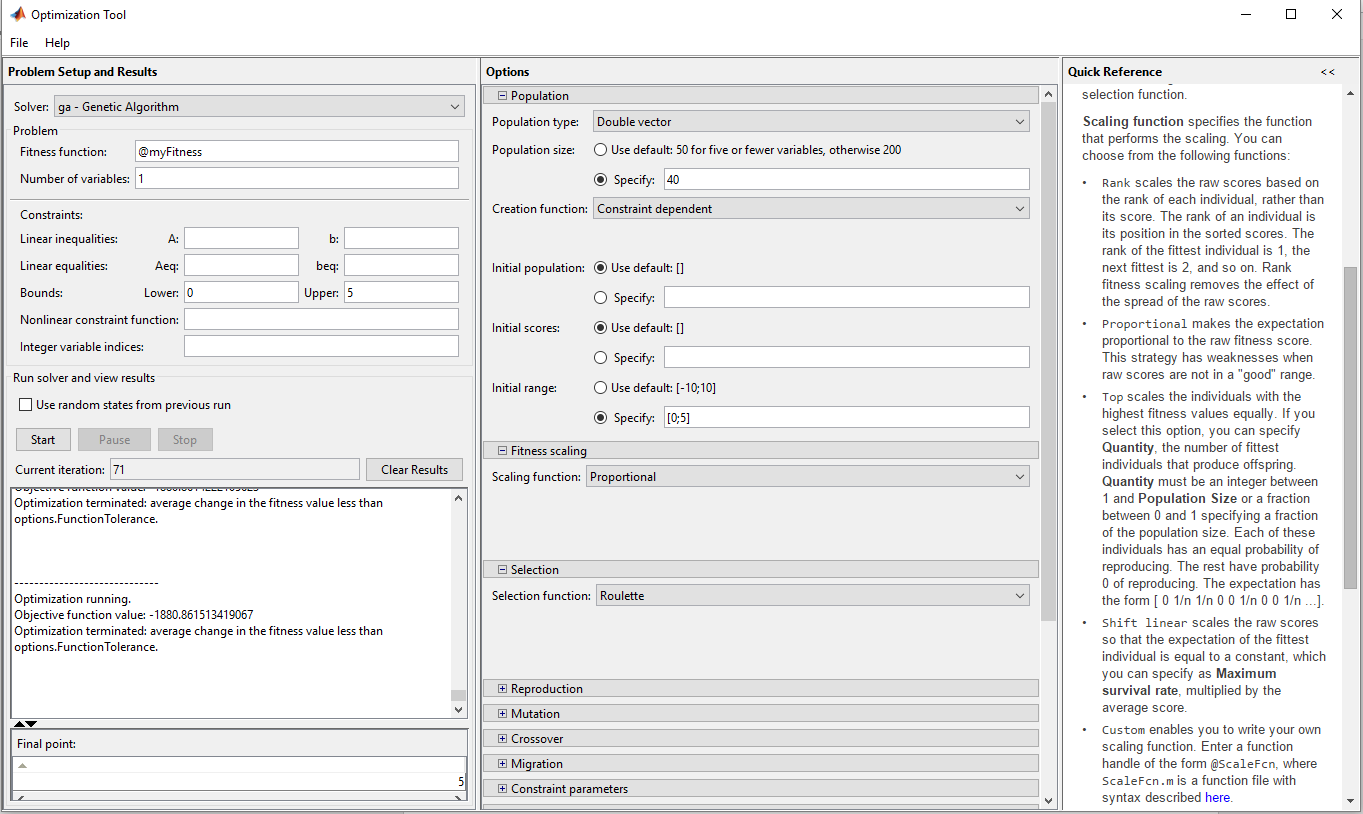
* Nhận xét: Nhìn chung thuật toán vẫn tìm ra được giá trị max của hàm số (dù có sai số là rất nhỏ)

# **III. Tìm giá trị max sử dụng Toolbox GA Optimization Matlab**

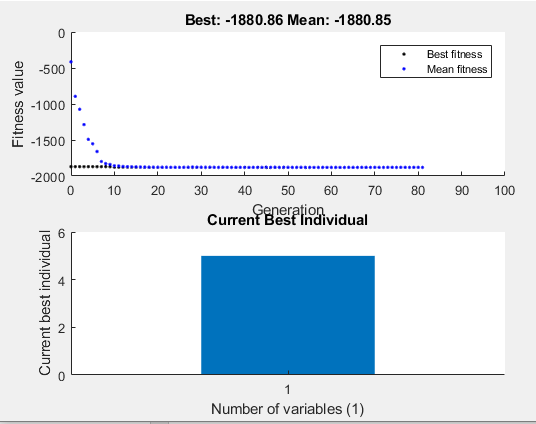
* Lưu ý: Toolbox chỉ áp dụng cho bài toán tìm min của hàm fitness bằng thuật toán ga
* Do đó muốn tìm max của hàm f(x) 🡪 ta tìm min của hàm –f(x).
* Đầu tiên, toolbox yêu cầu đầu vào là fitness function, là một file (.m). Ta sẽ viết một file (.m) có chứa hàm fitness là –f(x).



* Sử dụng toolbox Optimization chọn ga, nhập tên file chứa hàm, tùy chọn các thông số.



* Sau khi thiết lập các thông số, bấm Start. Ở đây chúng ta sẽ quan sát giá trị fitness tốt nhất và giá trị fitness trung bình qua mỗi thế hệ.



* Giá trị min của hàm –f(x) = -1880.86 hội tụ tại x = 5
* Giá trị max của hàm f(x) = 1880.86 hội tụ tại x = 5

**Nhận xét:**

* Khi sử dụng Toolbox, có sự hội tụ cả về giá trị fitness tốt nhất và giá trị fitness trung bình 🡪 cách làm đáng tin cậy hơn code đã viết ở phần trên. Nhóm đã thử cho hội tụ giá trị fitness trung bình nhưng gần như là không thể hội tụ lại (có thể do phần code chưa tối ưu)
* Nhìn chung kết quả khi sử dụng code tự viết và sử dụng Toolbox có thể tìm ra được giá trị lớn nhất của một hàm số qua giải thuật di truyền tương đối chính xác.