The Continuous Genetic Algorithm

Dr H.K. Lam

Department of Engineering
King's College London

Office S2.14, Strand Building, Strand Campus

Email: hak-keung.lam@kcl.ac.uk
https://nms.kcl.ac.uk/hk.lam

Nature-Inspired Learning Algorithms (7CCSMBIM)

Outline

Learning Aims and Objectives

Aims

- To understand the process of the continuous genetic algorithms.
- To apply the continuous genetic algorithm to optimisation problems.
- To know the limitations of the continuous genetic algorithms.

Objectives

- To study how the continuous genetic algorithm works in details.
- To consider a number of applications and formulate as minimisation problems.

The Continuous Genetic Algorithm

Introduction

The Continuous Genetic Algorithm

- Requires less storage than the binary GA.
  - A single floating-point number v.s. N_{bit} of '0's and '1's.
- Allows representation to the machine precision.

- Inherently runs faster than the binary GA as no encoding and decoding needed.
- Deals with complex problem with high dimensionality.

- More logical to represent variables by floating-point numbers when the problems are continuous.

The Continuous Genetic Algorithm

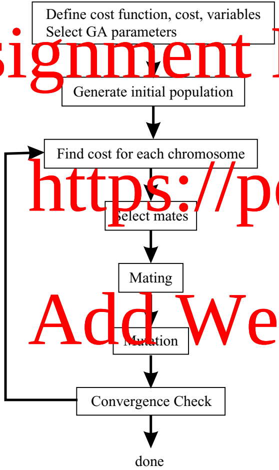

Figure 1: Flowchart of a continuous genetic algorithm

- The optimisation variables are represented by *chromosome*.

$$chromosome = [p_1, p_2, \cdots, p_{N_{var}}]$$

- Each gene (p_i, i = 1, 2, $\cdots$, N_{var}) is a real-coded variable.
- The cost is evaluated by a cost (fitness) function.

$$cost = f(chromosome) = f(p_1, p_2, \cdots, p_{N_{var}})$$

- Variable values are represented as floating-point numbers; no longer need to consider how many bits are necessary to accurately represent a value.

- No encoding and decoding before cost function evaluation.
- Only limited to the internal precision and round-off error of computers.
- Natural form of real-valued cost function can be used directly.

- The GA starts with an initial population with N_{pop} chromosomes with an $N_{pop} \times N_{var}$ matrix filled with randomly generated real values.

Example: A cost function: $cost = f(x,y) = x\sin(4x) + 1.1y\sin(2y)$ subject to $0 \leq x \leq 10$ and $0 \leq y \leq 10$.

$$chromosome = [x, y]$$

x	y	Cost
6.9745	0.8342	3.4766
0.30359	9.6828	5.5408
2.402	9.3359	-2.2528
0.18758	8.9371	-8.0108
2.6974	6.2647	-2.8957
5.613	0.1289	-2.4601
7.7246	5.5655	-9.8884
6.8537	9.8784	13.752

Table 1: Example initial population

- N_{pop} chromosomes are ranked from lowest cost to highest cost.
- Only the best are selected to continue, while the rest are discarded.

n	x	y	Cost
1	7.7246	5.5655	-9.8884
2	0.1876	8.9371	-8.0108
3	2.6974	6.2647	-2.8957
4	5.6130	0.1289	-2.4601

Table 2: Surviving chromosomes after a 50 % selection rate.

Selection

- **Pairing from top to bottom** until the top N_{keep} chromosomes are selected for mating.

- **Random pairing:** All chromosomes have the same probabilities to mate.
- **Weighted random pairing** (roulette wheel)
  - Rank weighting
  - Cost weighting

- **Tournament selection** picks randomly a small subset of two or three, and the chromosome with the lowest cost in this subset becomes a parent.

1) **Swapping**

   1a) Single-point crossover

   1b) Double-point crossover

   1c) Uniform crossover

2) **Blending**

3) **Extrapolation**

Single-point crossover:

$$parent_1 = [p_{m1}, p_{m2}, p_{m3}, p_{m4}, p_{m5}, p_{m6}, \cdots, p_{mN_{var}}]$$

$$parent_2 = [p_{d1}, p_{d2}, p_{d3}, p_{d4}, p_{d5}, p_{d6}, \cdots, p_{dN_{var}}]$$

$$offspring_1 = [p_{m1}, p_{m2}, \uparrow p_{d3}, p_{d4}, p_{d5}, p_{d6}, \cdots, p_{dN_{var}}]$$

$$offspring_2 = [p_{d1}, p_{d2}, \uparrow p_{m3}, p_{m4}, p_{m5}, p_{m6}, \cdots, p_{mN_{var}}]$$

Single-point crossover:

$$parent_1 = [p_{m1}, p_{m2}, p_{m3}, p_{m4}, p_{m5}, p_{m6}, \cdots, p_{mN_{var}}]$$

$$parent_2 = [p_{d1}, p_{d2}, p_{d3}, p_{d4}, p_{d5}, p_{d6}, \cdots, p_{dN_{var}}]$$

$$offspring_1 = [p_{m1}, p_{m2}, \uparrow p_{d3}, p_{d4}, p_{d5}, p_{d6}, \cdots, p_{dN_{var}}]$$

$$offspring_2 = [p_{d1}, p_{d2}, \uparrow p_{m3}, p_{m4}, p_{m5}, p_{m6}, \cdots, p_{mN_{var}}]$$

Double-point crossover:

$$offspring_1 = [p_{m1}, p_{m2}, \uparrow p_{d3}, p_{d4}, \uparrow p_{m5}, p_{m6}, \cdots, p_{mN_{var}}]$$

$$offspring_2 = [p_{d1}, p_{d2}, \uparrow p_{m3}, p_{m4}, \uparrow p_{d5}, p_{d6}, \cdots, p_{dN_{var}}]$$

Single-point crossover:

$$parent_1 = [p_{m1}, p_{m2}, p_{m3}, p_{m4}, p_{m5}, p_{m6}, \cdots, p_{mN_{var}}]$$

$$parent_2 = [p_{d1}, p_{d2}, p_{d3}, p_{d4}, p_{d5}, p_{d6}, \cdots, p_{dN_{var}}]$$

$$offspring_1 = [p_{m1}, p_{m2}, \uparrow p_{d3}, p_{d4}, p_{d5}, p_{d6}, \cdots, p_{dN_{var}}]$$

$$offspring_2 = [p_{d1}, p_{d2}, \uparrow p_{m3}, p_{m4}, p_{m5}, p_{m6}, \cdots, p_{mN_{var}}]$$

Double-point crossover:

$$offspring_1 = [p_{m1}, p_{m2}, \uparrow p_{d3}, p_{d4}, \uparrow p_{m5}, p_{m6}, \cdots, p_{mN_{var}}]$$

$$offspring_2 = [p_{d1}, p_{d2}, \uparrow p_{m3}, p_{m4}, \uparrow p_{d5}, p_{d6}, \cdots, p_{dN_{var}}]$$

Uniform crossover: randomly chooses whether or not to swap information between the two parents.

Disadvantage: Crossover by swapping does not introduce new information, just differet combinations. It totally relies on mutation to introduce new genetic material.

Crossover: Blending

- The new offspring comes from a combination of the two parents.

$$p_{new_1} = \beta p_{mn} + (1-\beta) p_{dn}$$

$$p_{new_2} = (1-\beta) p_{mn} + \beta p_{dn}$$

- p_{mn} is the n^{th} variable in the mother chromosome.
- p_{dn} is the n^{th} variable in the father chromosome.
- β is a random number in the range of 0 and 1.
- The same or different β can be used for each variable.
- Linear combination process is done for all variables to the right or to the left of some crossover point.
- Any number of points can be chosen to blend.
- **Disadvantage:** It does not allow the introduction of values beyond the extremes already represented in the population.

- Generate a random position n.

Method 1: blending at the n^{th} point and swap genes from $n+1$ to N_{var}

$$offspring_1 = \left[p_{m1}, p_{m2}, \cdots, \underbrace{p_{new_1}}_{n^{th}gene}, p_{d,n+1}, \cdots, p_{dN_{var}} \right]$$

$$offspring_2 = \left[p_{d1}, p_{d2}, \cdots, \underbrace{p_{new_2}}_{n^{th}gene}, p_{m,n+1}, \cdots, p_{mN_{var}} \right]$$

Method 2: blending genes from the point n to point N_{var}

$$offspring_1 = \left[p_{m1}, p_{m2}, \cdots, \underbrace{p_{new_1,n}}_{n^{th}\,gene}, p_{new_1,n+1}, \cdots, p_{new_1,N_{var}} \right]$$

$$offspring_2 = \left[p_{d1}, p_{d2}, \cdots, \underbrace{p_{new_2,n}}_{n^{th}\,gene}, p_{new_2,n+1}, \cdots, p_{new_2,N_{var}} \right]$$

- The new offspring comes from a combination of the two parents.

$$p_{new_1} = p_{mn} - \beta(p_{mn} - p_{dn})$$

$$p_{new_2} = p_{dn} + \beta(p_{mn} - p_{dn})$$

where $\beta \geq 0$ is a random number.

- It allows offspring generation outside of the two parent variable values.
- The offspring is discarded if outside of the allowed range, and run again.

- Variations include choosing any number of variables to modify and generating different β for each variable.

Method 1: blending at the n^{th} point and swap genes from $n+1$ to N_{var}

$$offspring_1 = \left[p_{m1}, p_{m2}, \cdots, \underbrace{p_{new_1}}_{n^{th}\, gene}, p_{d,n+1}, \cdots, p_{dN_{var}} \right], offspring_2 = \left[p_{d1}, p_{d2}, \cdots, \underbrace{p_{new_2}}_{n^{th}\, gene}, p_{m,n+1}, \cdots, p_{mN_{var}} \right]$$

Method 2: blending genes from the point n to point N_{var}

$$offspring_1 = \left[p_{m1}, p_{m2}, \cdots, \underbrace{p_{new_1,n}}_{n^{th}\, gene}, p_{new_1,n+1}, \cdots, p_{new_1,N_{var}} \right], offspring_2 = \left[p_{d1}, p_{d2}, \cdots, \underbrace{p_{new_2,n}}_{n^{th}\, gene}, p_{new_2,n+1}, \cdots, p_{new_2,N_{var}} \right]$$

Example (method 1): Consider $\beta = 0.0272$

$$chromosome_2 = [0.1876, \quad 8.9371]$$

$$chromosome_3 = [2.6974, \quad 6.2647]$$

$$offspring_1 = [0.1876 - 0.0272 \times (0.1876 - 2.6974), 6.2647] = [0.2559, 6.2647]$$

$$offspring_2 = [2.6974 + 0.0272 \times (0.1876 - 2.6974), 8.9371] = [2.6291, 8.9371]$$

Mutations:

- Choose a mutation rate μ.
- Total number of variables that can be mutated in the population: $\mu(N_{pop}-1)N_{var}$ (Elitism is implemented).

- Mutate the chosen variable as: $p_n' = p_n + \sigma N_n(0,1)$
  - σ is a chosen constant.
  - $N_n(0,1)$: standard normal distribution (mean = 0 and standard deviation = 1).

- Replace the chosen variable by p_n'.
- If bounds exceed, discard and generate again.
- Generally not allowed on the best solution (elitism).

Example: $\mu = 20\%$.

Number of variables to be mutated: $0.2 \times 7 \times 2 = 2.8 \approx 3$

Randomly generate row and column numbers.

$Row = [4 \quad 4 \quad 7]; Column = [1 \quad 2 \quad 1]$

Population after crossover		Population after Mutation		
x	y	x	y	Cost
7.7246	5.5655	7.7246	5.5655	-9.8884
0.18758	8.9371	0.18758	8.9371	-8.0108
2.6974	6.2647	2.6974	6.2647	-2.8957
5.6130	0.1239	9.819	7.1315	17.601
0.2558	6.2647	0.2558	6.2647	-0.03688
2.6292	8.9371	2.6292	8.9371	-10.472
6.6676	5.5655	9.1602	5.5655	$-\mathbf{14.05}$
3.7544	6.2647	3.7544	6.2647	2.1359

Table 3: Mutating Population.

Example: Consider a function $f(x,y,z) = x - 2xy + 3z$ to be minimised, where $2 \leq x, y, z \leq 5$.

Example: Consider a function $f(x,y,z) = x - 2xy + 3z$ to be minimised, where $2 \leq x,y,z \leq 5$.

- Chromosome: $[x,y,z]$

Example: Consider a function $f(x,y,z) = x - 2xy + 3z$ to be minimised, where $2 \le x, y, z \le 5$.

- Chromosome: $[x, y, z]$

Step 1: Population initialised with population size = 4

n	Chromosome	Cost
1	[4.7401, 3.8971, 2.2926]	-25.3274
2	[2.8355, 3.6406, 4.8725]	-3.1928
3	[4.4442, 4.7174, 2.3810]	-30.3429
4	[4.8947, 2.4728, 4.9118]	-4.5771

Example: Consider a function $f(x, y, z) = x - 2xy + 3z$ to be minimised, where $2 \leq x, y, z \leq 5$.

Step 2: Ranked population and natural selection with $N_{keep} = 3$

n	Chromosome	Cost
1	[4.4442, 4.7174, 2.3810]	-30.3429
2	[4.7401, 3.8971, 2.2926]	-25.3274
3	[4.8947, 2.4728, 4.9118]	-4.5771
4	[2.8355, 3.6406, 4.8725]	-3.1928

KING'S College LONDON
University of London

Example: Consider a function $f(x,y,z) = x - 2xy + 3z$ to be minimised, where $2 \le x, y, z \le 5$.

Step 3: Selection with rank weighting (roulette wheel weighting)

n	Chromosome	Cost	$P_n = \frac{N_{keep} - n + 1}{1+2+3}$	$\sum_{i=1}^{n} P_n$
1	[4.4442, 4.7174, 2.3810]	-30.3429	0.5	0.5
2	[4.7401, 3.8971, 2.2926]	-25.3274	0.3333	0.8333
3	[4.8947, 2.4728, 4.9118]	-4.5711	0.1667	1.0000
4	[2.8355, 3.6406, 4.8725]	-3.1928		

- Generate two random numbers: 0.0975, 0.6324

Example: Consider a function $f(x,y,z) = x - 2xy + 3z$ to be minimised, where $2 \leq x,y,z \leq 5$.

Step 4: Crossover with single-point crossover technique

p_1: [4.4442, 4.7174, 2.3810]

p_2: [4.7401, 3.8971, 2.2926]

Example: Consider a function $f(x,y,z) = x - 2xy + 3z$ to be minimised, where $2 \le x, y, z \le 5$.

Step 4: Crossover with single-point crossover technique

p_1: [4.4442, 4.7174, 2.3810]

p_2: [4.7401, 3.8971, 2.2926]

• Generate randomly the crossover point: 2

$offspring_1$: [4.4442, 3.8971, 2.2926] $\Rightarrow$ Cost: -23.3170

$offspring_2$: [4.7401, 4.7174, 2.3810] $\Rightarrow$ Cost: -32.8388

Example: Consider a function $f(x,y,z) = x - 2xy + 3z$ to be minimised, where

$2 \leq x,y,z \leq 5.$

Step 5: Mutation

$\mu = 0.2; \#mutation = 0.2(4-1)3 = 1.8 \approx 2$

Row = [2 3]; Column = [1 2]

n	Chromosome	Chromosome after mutation	Cost
1	[4.4442, 4.7174, 2.3810]	[4.4442, 4.7174, 2.3810]	-30.3429
2	[4.7401, 3.8971, 2.2926]	[4.5570, 3.8971, 2.2926]	-24.0834
3	[4.8947, 2.4728, 4.9118]	[4.8947, 3.3312, 4.9118]	-12.9803
4	[4.7401, 4.7174, 2.3810]	[4.7401, 4.7174, 2.3810]	-32.8388

- $\sigma = 1$
- 4.7401 + $\sigma \times rand$ = 4.7401 $-$ 0.1831 = 4.5570
- 2.4728 + $\sigma \times rand$ = 2.4728 + 0.8584 = 3.3312

KING'S College LONDON
University of London

Example: Consider a function $f(x, y, z) = x - 2xy + 3z$ to be minimised, where $2 \leq x, y, z \leq 5$.

Ranked population at the next iteration:

n	Chromosome	Cost
1	[4.7401, 4.7174, 2.3810]	-32.8388
2	[4.4442, 4.7174, 2.3810]	-30.3429
3	[4.5570, 3.8971, 2.2926]	-24.0834
4	[4.8947, 3.3312, 4.9118]	-12.9803