Grasshopper Optimization Algorithm (GOA)

Seyed AmirHossein Adhami MirHosseini

سید امیرحسین ادهمی میرحسینی

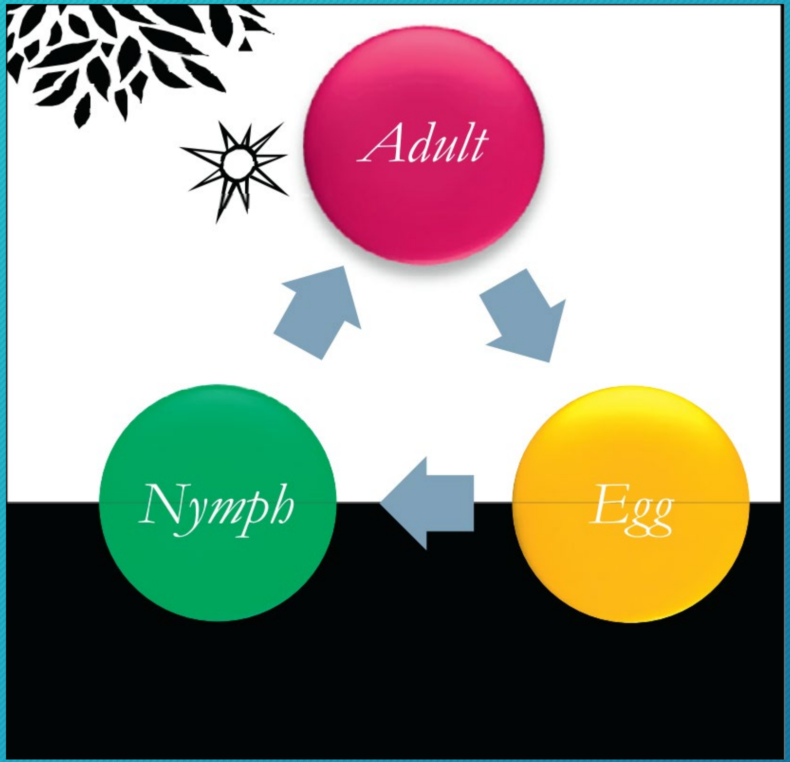
adhamiamirhossein@gmail.com

پاییز ۱۳۹۷

Introduction

- Steps to solve an optimization problem
  1. The parameters of the problem should be identified.
  2. The constrains that are applied to the parameters have to be recognized .
  3. The objectives of the given problem should be investigated.
  4. A suitable optimizer should be chosen to solve the problem.

Grasshopper Optimization Algorithm (GOA)



Grasshopper Optimization Algorithm (GOA)

- Mathematical modeling
  - Swarming behavior of grasshoppers
    - $X_i = S_i + G_i + A_i$
  - Random behavior
    - $X_i = r_1 S_i + r_2 G_i + r_3 A_i$ where r1,r2 and r3 are random numbers in [0,1]
    - $S_i = \sum_{\substack{i=1 \\ j\neq 1}}^{N} s(d_{ij})\widehat{d_{ij}}$
      - $d_{ij} = |x_j - x_i|$
      - $\widehat{d_{ij}} = \frac{x_j - x_i}{d_{ij}}$
      - $s(r) = fe^{\frac{-r}{l}} - e^{-r}$

Grasshopper Optimization Algorithm (GOA)

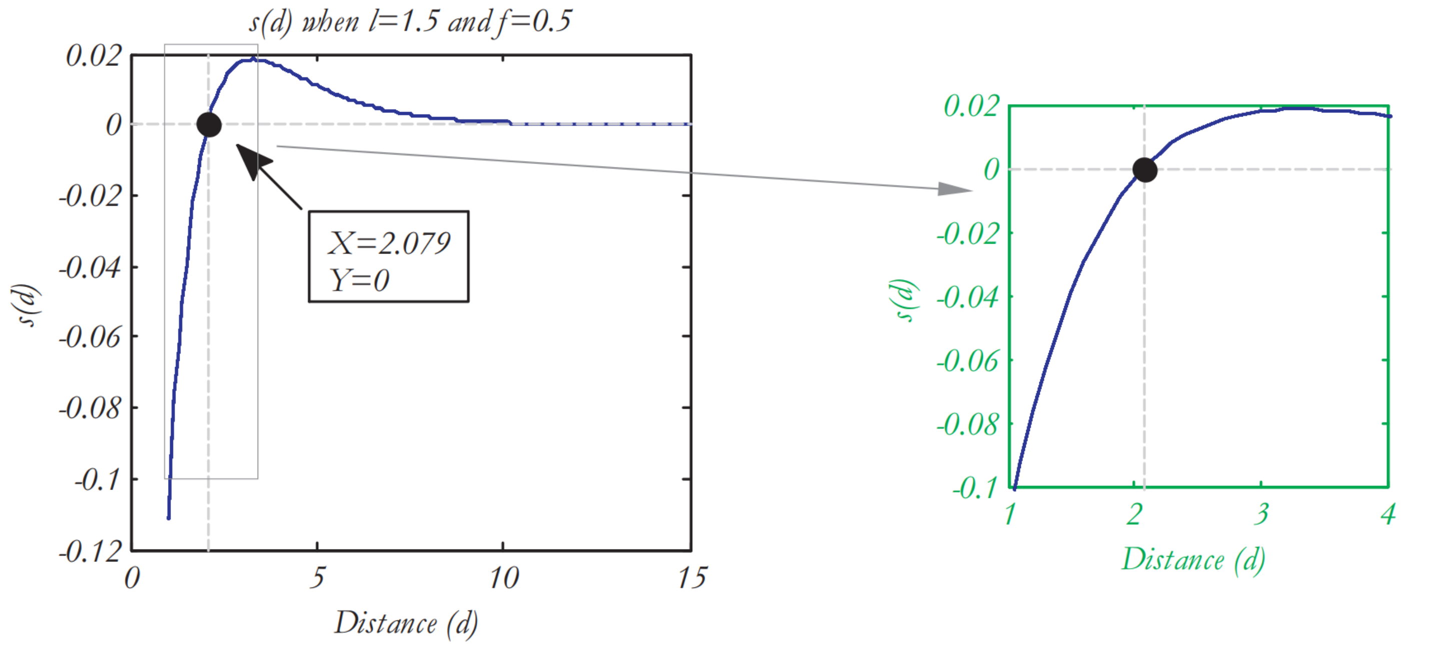
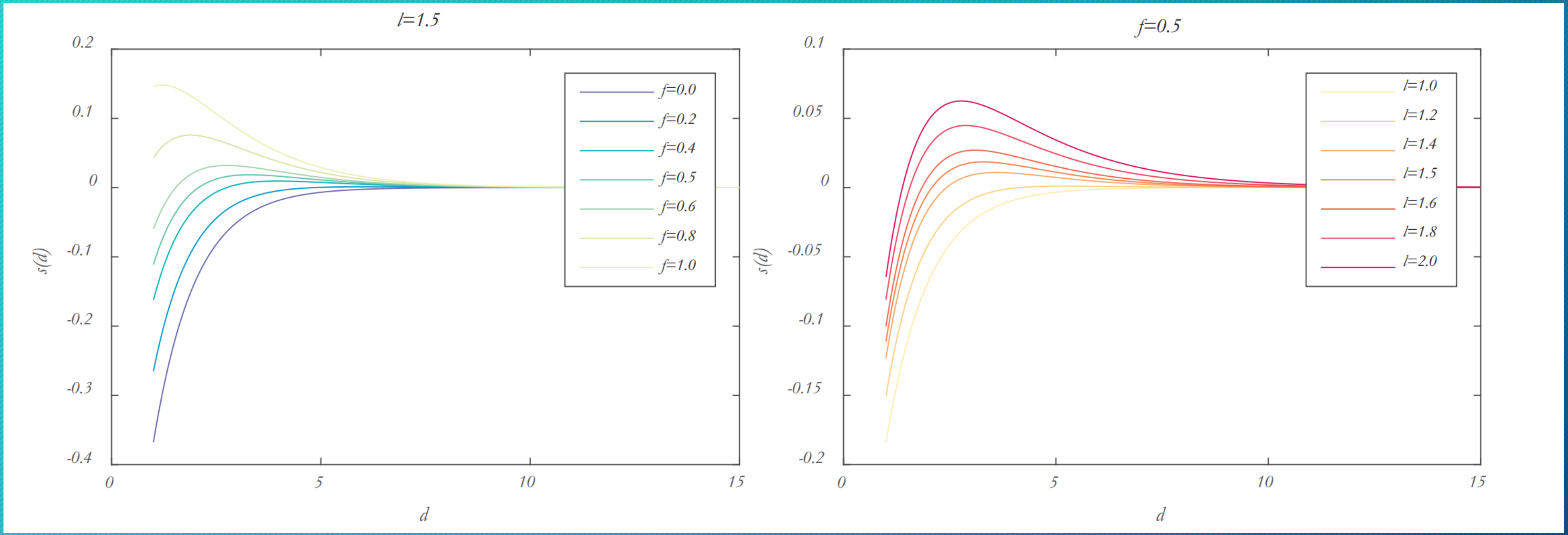


Fig. 2. (left) Function s when l=1.5 and f=0.5 (right) range of function s when x is in [1,4].

Grasshopper Optimization Algorithm (GOA)



Grasshopper Optimization Algorithm (GOA)

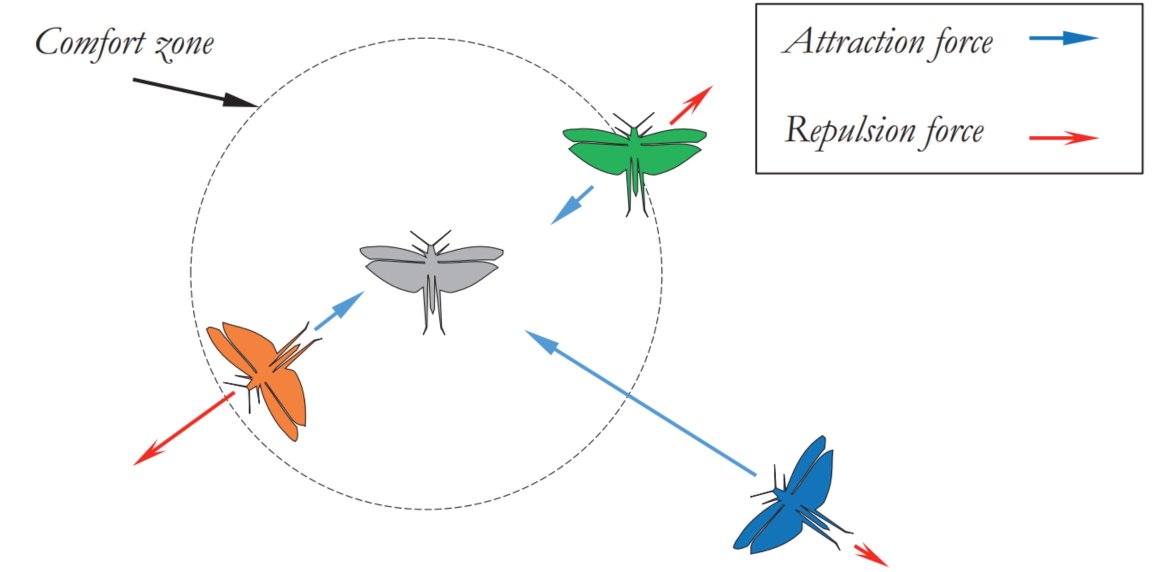


Fig. 4. Primitive corrective patterns between individuals in a swarm of grasshoppers.

Grasshopper Optimization Algorithm (GOA)

- $G_i = -g\widehat{e_g}$
- $A_i = u\widehat{e_w}$

- $X_i = \sum_{\substack{i=1 \\ j \neq 1}}^{N} s(|x_j - x_i|)\frac{x_j - x_i}{d_{ij}} - g\,\widehat{e_g} + u\widehat{e_w}$
  - Where $s(r) = fe^{\frac{-r}{l}} - e^{-r}$ and N in the number of grasshoppers

Grasshopper Optimization Algorithm (GOA)

- $X_i^d = c\left(\sum_{\substack{i=1 \\ j \neq 1}}^{N} c\frac{ub_d - lb_d}{2} s\left(\left|x_j^d - x_i^d\right|\right)\frac{x_j - x_i}{d_{ij}}\right) + \widehat{T_d}$
    - $c = cmax - l\frac{cmax - cmin}{L}$

Grasshopper Optimization Algorithm (GOA)

Initialize the swarm X_i *(i = 1, 2, ..., n)*
Initialize cmax, cmin, and maximum number of iterations
Calculate the fitness of each search agent
T=the best search agent
while *(l < Max number of iterations)*
    Update c using Eq. (2.8)
    for *each search agent*
        Normalize the distances between grasshoppers in [1,4]
        Update the position of the current search agent by the equation (2.7)
        Bring the current search agent back if it goes outside the boundaries
    end for
    Update T if there is a better solution
    l=l+1
end while
Return T

End