

CUCKOO SEARCH

IMPLEMENTATION AND PRELIMINARY TESTS

Presented by
Jorge Mario Cruz-Duarte,
Ph.D. Student

jorge.cruz@ugto.mx



Cuckoo Search was proposed by Yang, X. S. and Deb, S. in 2009 [1].

Cuckoo Search via Lévy Flights

Xin-She Yang

Department of Engineering,
University of Cambridge
Trumpington Street
Cambridge CB2 1PZ, UK
Email: xy227@cam.ac.uk

Suash Deb

Department of Computer Science & Engineering
C. V. Raman College of Engineering
Bidyaganagar, Mahura, Janla
Bhubaneswar 752054, INDIA
Email: suashdeb@gmail.com

Abstract—In this paper, we intend to formulate a new meta-heuristic algorithm, called Cuckoo Search (CS), for solving optimization problems. This algorithm is based on the obligate brood parasitic behaviour of some cuckoo species in combination with the Lévy flight behaviour of some birds and fruit flies. We validate the proposed algorithm against test functions and then compare its performance with those of genetic algorithms and particle swarm optimization. Finally, we discuss the implication of the results and suggestion for further research.

Index Terms—algorithm; cuckoo search; Lévy flight; meta-heuristics; nature-inspired strategy; optimization;

II. CUCKOO BEHAVIOUR AND LÉVY FLIGHTS

A. Cuckoo Breeding Behaviour

Cuckoo are fascinating birds, not only because of the beautiful sounds they can make, but also because of their aggressive reproduction strategy. Some species such as the *ani* and *Guira* cuckoos lay their eggs in communal nests, though they may remove others' eggs to increase the hatching probability of their own eggs [12]. Quite a number of species engage the obligate brood parasitism by laying their eggs in the

[1] Yang, X.-S., & Suash Deb. (2009). Cuckoo Search via Levy flights. In *2009 World Congress on Nature & Biologically Inspired Computing (NaBIC)* (pp. 210–214). IEEE. <http://doi.org/10.1109/NABIC.2009.5393690>

Repositorio: <https://github.com/jcrvz/cimat-computo-evolutivo>

The screenshot shows the GitHub repository page for 'jcrvz / cimat-computo-evolutivo'. The top navigation bar includes links for Personal, Open source, Business, Explore, Pricing, Blog, Support, and options to Sign in or Sign up. The repository header shows 'jcrvz / cimat-computo-evolutivo' with 1 star and 0 forks. Below the header are tabs for Code, Issues (0), Pull requests (0), Projects (0), Pulse, and Graphs. A summary bar indicates 33 commits, 1 branch, 0 releases, 1 contributor, and a license of GPL-3.0. A red arrow points to the 'Clone or download' button, which is highlighted with a red box. The main content area displays a list of recent commits, including updates to README.md, CuckooSearch.m, Exercises.m, and other files, along with their commit messages and dates.

File	Commit Message	Date
README.md	Update README.md	Latest commit c689b19 4 days ago
CuckooSearch.m	Update CuckooSearch.m	4 days ago
Exercises.m	Add files via upload	4 days ago
Jamil, Yang - 2013 - A literature survey of benchmark functions fo...	Add files via upload	a month ago
LICENSE	Initial commit	a month ago
LevyFlight.m	Add files via upload	4 days ago
README.md	Update README.md	4 days ago
Rakhshani, Rahati - 2016 - Intelligent Multiple Search Strategy Cu...	Add files via upload	a month ago
Slides_CuckooSearch_ECECIMAT.pdf	Diapositivas sobre el curso	4 days ago



A Brief Introduction

Image taken from: <http://alpinebirding.zenfolio.com/p1056394909/h5AED9240#h5aed9240>

Cuckoo birds behaviour

The brood parasitism



Images taken from: Wikipedia contributors. Brood parasite. Wikipedia, The Free Encyclopedia. April 2, 2016, 09:39 UTC. Available at: https://en.wikipedia.org/w/index.php?title=Brood_parasite&oldid=713161956. Accessed April 15, 2016. Wikipedia contributors. Striped cuckoo. Wikipedia, The Free Encyclopedia. February 11, 2016, 21:19 UTC. Available at: https://en.wikipedia.org/w/index.php?title=Striped_cuckoo&oldid=704481047. Accessed April 15, 2016.

Foraging behaviour

Survival strategies



Images taken from: <http://i.kinja-img.com/gawker-media/image/upload/s--Y7LnrGO4--/18lonnm92lcl6jpg.jpg>,
<http://cdn.instructables.com/FI5/L9MT/GE05CW88/FI5L9MTGE05CW88.LARGE.jpg>



Looking for Nests via Lévy Flights

Lévy Flights have infinite mean and variance: Exploration is more efficient than performed by Gaussian processes.

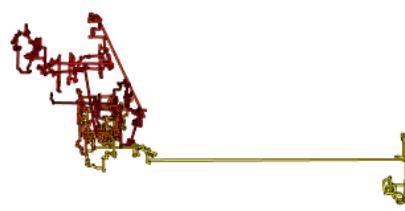
$$\vec{r}(t + 1) = \vec{r}(t) + \boxed{\vec{d}(t)}$$

“Independent, identically distributed (iid) random variables” in \mathbb{R}^d [2].

Gaussian distribution



Lévy distribution



$$f(x; \mu, c) = \sqrt{\frac{c}{2\pi}} \frac{e^{-\frac{c}{2(x-\mu)}}}{(x - \mu)^{3/2}}$$

(Lévy motion)

[1] X. Yang and S. Deb, “Cuckoo search: recent advances and applications,” *Neural Comput. Appl.*, pp. 1–9, 2014.

[2] THAOKAR, R.M. 2008. *Brownian motion of a torus*. S.l. ISBN 9780521760188.

Image taken from: <http://www.rspb.org.uk/discoverandenjynature/discoverandlearn/birdguide/name/c/cuckoo/>

The basic idea

Three idealised rules:

1. One egg is laid in a randomly chosen nest by each cuckoo at a time

An egg represents a solution

2. High quality eggs (best nests) shall carry out to the next generation

Best nest or egg is the best solution

3. There exists a probability (p_D) for the host bird discover the cuckoo's egg hidden in its nest

p_D could be simplified as a nests' proportion being replaced

[1] MARICHELVAM, M.K., PRABAHARAN, T. and YANG, X.S. 2014. Improved cuckoo search algorithm for hybrid flow shop scheduling problems to minimize makespan. *Applied Soft Computing*. Vol. 19, pp. 93–101. ISSN 15684946.

Highlights

1. “**CS is a population-based algorithm** (like GA and PSO) **powered by some sort of elitism and/or selection** (similar to HS).”
2. “**The randomisation is more efficient** as the step length is heavy-tailed, and any large step is possible.”
3. “**The number of parameters to be tuned is less than GA and PSO.**”
4. “**Local and global search are controlled by the switching/discovery probability.**”

[1] X.-S. Yang and Suash Deb, “Cuckoo Search via Levy flights,” in *2009 World Congress on Nature & Biologically Inspired Computing (NaBIC)*, 2009, pp. 210–214.



Before Implement It

Image taken from: <http://www.clipartqueen.com/image-files/bird-clip-art-swallow-nest.png>

Definitions

Definition 1: Let $\mathfrak{X}^t = \{\vec{x}_1^t, \vec{x}_2^t, \dots, \vec{x}_{N_N}^t\}$ be a finite set of candidate solutions (or nests) of any optimisation problem in \mathbb{R}^{N_D} , with a cost function given by $f: \mathbb{R}^{N_D} \rightarrow \mathbb{R}$, N_N is called number of available nests and $\vec{x}_i^t \in \mathbb{R}^{N_D}$ denotes the i -th solution (or nest with an unique egg) at the time t of an iterative procedure.

Definition 2: Let $\vec{x}_*^t \in \mathfrak{X}^t$ be the best solution (or the best camouflaged cuckoo's egg in a nest) found at the t -th iteration, and determined by $\vec{x}_*^t = \operatorname{argmin}(\{f(\mathfrak{X}^t)\}, f(\vec{x}^*))$, with $\{f(\mathfrak{X}^t)\} = f(\vec{x}_1^t), f(\vec{x}_2^t), \dots, f(\vec{x}_{N_N}^t)$.

Definition 3: $\mathfrak{X}^{t+1} = \{\vec{x}_1^{t+1}, \vec{x}_2^{t+1}, \dots, \vec{x}_{N_N}^{t+1}\}$ represents the finite set of new eggs (or candidate solutions) to be located in a different set of nests. Each new nest position \vec{x}_i^{t+1} can be obtained through two stochastic strategies.

Definitions

Definition 4: Let $\lambda_{L,i}$ be a random number with a symmetric Lévy stable distribution. It can be obtained from **Mantegna's algorithm**, as follows,

$$\lambda_{L,i} = n_{0,\sigma} |n_{0,1}|^{-1/\xi}, \quad \sigma^\xi = \frac{\sin\left(\frac{\pi\xi}{2}\right) \Gamma(\xi + 1)}{\xi 2^{\frac{\xi-1}{2}} \Gamma\left(\frac{\xi+1}{2}\right)}$$

where $\xi \in [0.3, 2)$ is a distribution's parameter, $n_{0,\sigma}$ is a random number with normal distribution $\mathcal{N}(0, \sigma)$, and $\Gamma(\square)$ is the well-known Gamma function.

$$\Gamma(z) = \int_0^\infty x^{z-1} e^{-x} dx$$

Definitions

Definition 5: **Strategy 1**—Let $\vec{x}_i^{t+1} \in \mathbb{R}^{N_D}$ a new nest position from a previous poorer solution \vec{x}_i^t , $f(\vec{x}_i^t) > f(\vec{x}_*^t)$, which is determined by

$$\vec{x}_i^{t+1} = \vec{x}_i^t + \delta_x \vec{\lambda}_L \odot (\vec{x}_i^t - \vec{x}_*^t),$$

where δ_x is the step size, which is usually set as 0.1 for $N_D \leq 3$ and 0.01 for $N_D > 3$; $\vec{\lambda}_L = (\lambda_{L,1}, \lambda_{L,2}, \dots, \lambda_{L,N_D})^T$ is a vector of i.i.d. symmetric Lévy stable random variables; and \odot is the Hadamard-Schur product.

Definitions

Definition 6: **Strategy 2**—Let $\vec{x}_i^{t+1} \in \mathbb{R}^{N_D}$ a new nest position with an associated probability of $p_D \in [0,1]$, which is determined as

$$\vec{x}_i^{t+1} = \vec{x}_i^t + \vec{u} \odot \mathcal{H}(\vec{u} - p_D) \odot (\vec{x}_j^t - \vec{x}_k^t),$$

since $\vec{u} \in \mathbb{R}^{N_D}$ is a vector of i.i.d. uniform random variables between 0 and 1; $\mathcal{H}: \mathbb{R}^{N_D} \rightarrow \mathbb{R}^{N_D}$ is the extended Heaviside function, such as $\mathcal{H}(\vec{y}) = (\mathcal{h}(y_1), \mathcal{h}(y_2), \dots, \mathcal{h}(y_{N_D}))^T$, $\mathcal{h}: \mathbb{R} \rightarrow \mathbb{R}$, $\mathcal{h}(y) := \frac{d}{dy} \max\{y, 0\}$ is the Heaviside function; and $\vec{x}_j^t, \vec{x}_k^t \in \mathfrak{X}^t$ with $j \neq k$.

CS' pseudocode

Input: Cost function $\mathbf{f}(\vec{y})$, $\vec{y} = (y_1, y_2, \dots, y_{N_D})^T$,

Parameters such as number of nests or population ($N_N > 2$), step size (δ_x), switch probability (p_D), maximum number of generations (N_G) and stop criteria (if they exist).

Output: Best solution found, $\vec{x}_*^t \approx \operatorname{argmin}_{\vec{y}} \{\mathbf{f}(\vec{y})\}$.

Step 0. Set $t = 0$, initialise \mathfrak{X}^t and identify \vec{x}^* .

Step 1. Obtain \mathfrak{X}^{t+1} via $\vec{x}_i^{t+1} = \vec{x}_i^t + \delta_x \vec{\lambda}_L \odot (\vec{x}_i^t - \vec{x}_*^t)$,

Step 2. For each element of \mathfrak{X}^{t+1} , if $\mathbf{f}(\vec{x}_i^{t+1}) > \mathbf{f}(\vec{x}_i^t)$, make $\vec{x}_i^{t+1} = \vec{x}_i^t$, and update \vec{x}^* .

Step 3. Make $\mathfrak{X}^t = \mathfrak{X}^{t+1}$, choose randomly \vec{x}_j^t and \vec{x}_k^t from \mathfrak{X}^t , renew \mathfrak{X}^{t+1} via
$$\vec{x}_i^{t+1} = \vec{x}_i^t + \vec{u} \odot \mathcal{H}(\vec{u} - p_D) \odot (\vec{x}_j^t - \vec{x}_k^t)$$

Step 4. For each element of \mathfrak{X}^{t+1} , if $\mathbf{f}(\vec{x}_i^{t+1}) > \mathbf{f}(\vec{x}_i^t)$, make $\vec{x}_i^{t+1} = \vec{x}_i^t$, and update \vec{x}^* .

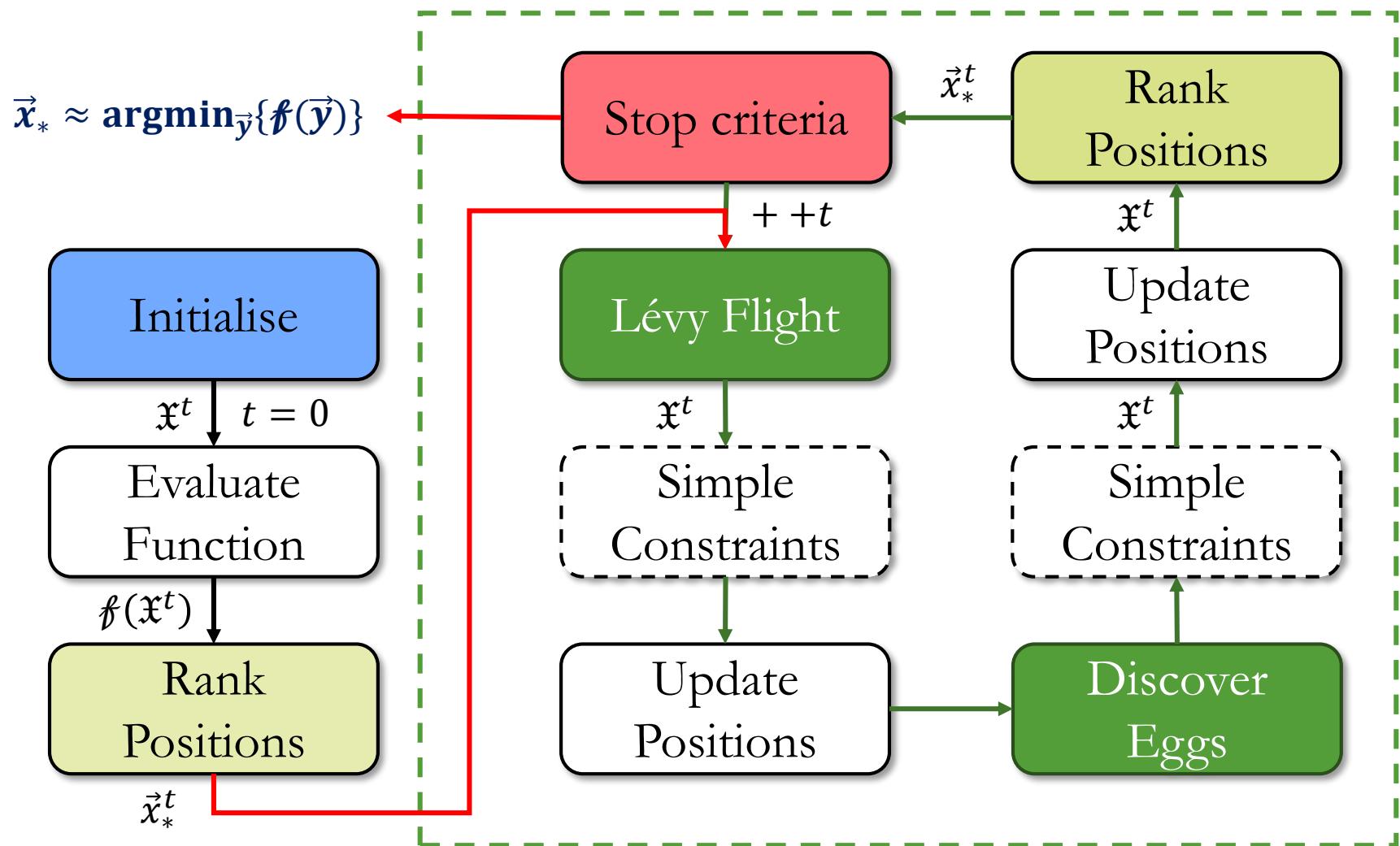
Step 5. Increment $t = t + 1$ and verify if $t \leq N_G$ and stop criteria are not reached, go to Step 1. Otherwise, print \vec{x}_*^t .

Implementing Cuckoo Search

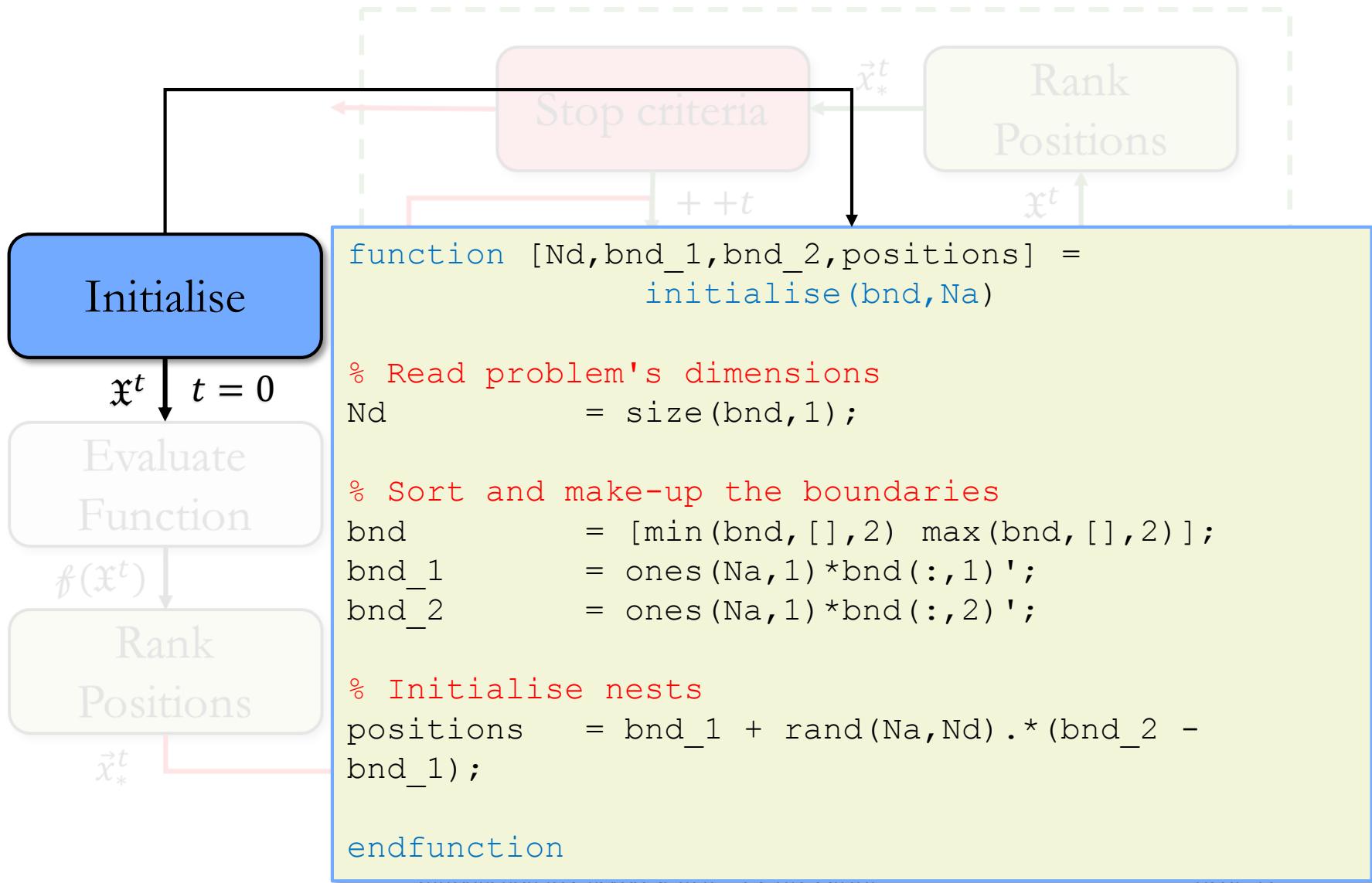


Image taken from: <http://www.planetofbirds.com/cuculiformes-cuculidae-black-billed-cuckoo-coccyzus-erythrophthalmus>

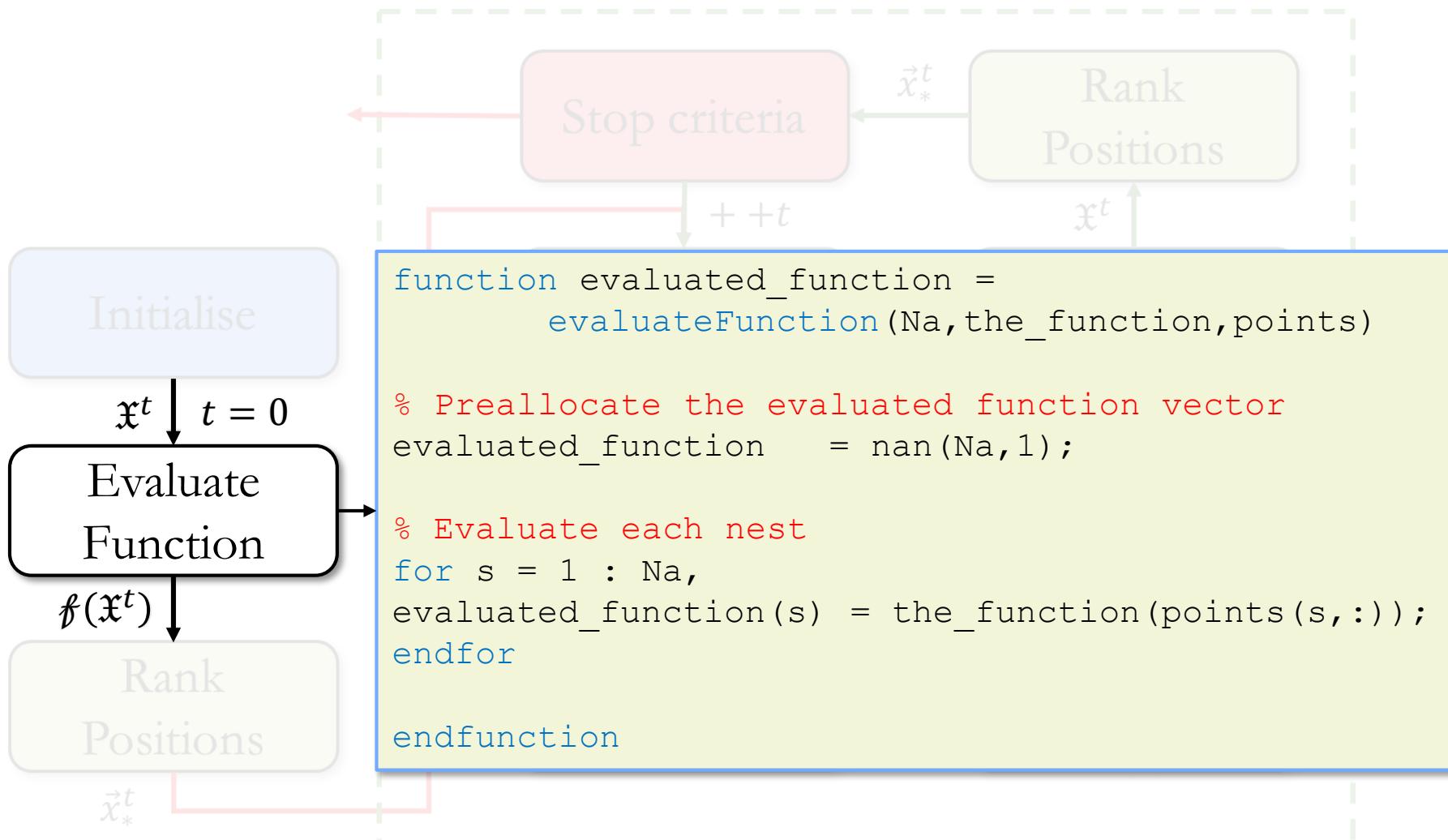
CS as a block diagram...



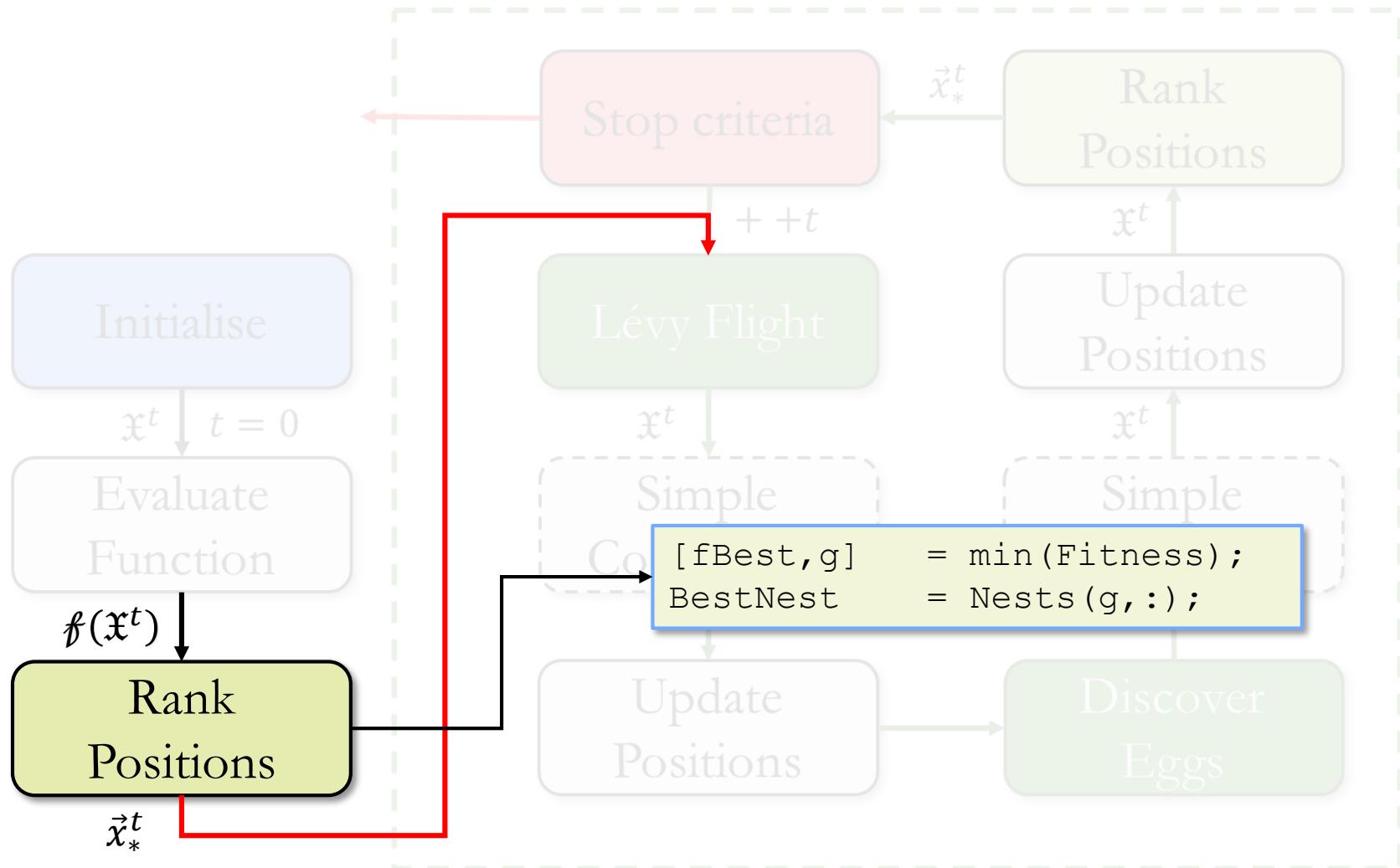
Initialise Function Block



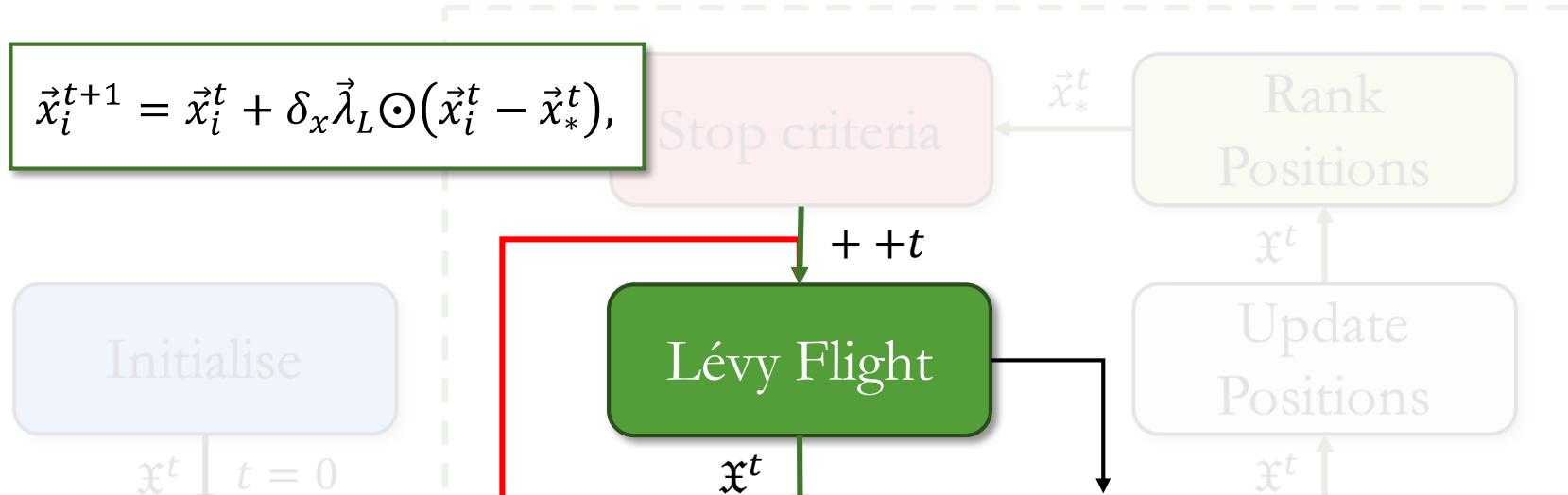
Evaluate Function Block



Rank Positions Block



Lévy Flight Block



```

function newNests = LevyFlight(Nests,Delta,Beta,BestNest)
% Read sizes
Nests    = size(Nests);

% Calculate std dev for N(0,Sigma) (Mantegna's algorithm)
Sigma    = (gamma(1 + Beta)*sin(pi*Beta/2)/(gamma((1 + Beta)/2)* ...
            Beta*2^((Beta - 1)/2)))^(1/Beta);

% Obtain Levy random variable: Lambda
Lambda    = (randn(sNests)*Sigma)./(abs(randn(sNests)).^(1/Beta));

% Update Nests' positions
newNests = Nests + Delta*randn(sNests).*Lambda.* (Nests-ones(sNests(1),1)*BestNest);
endfunction

```

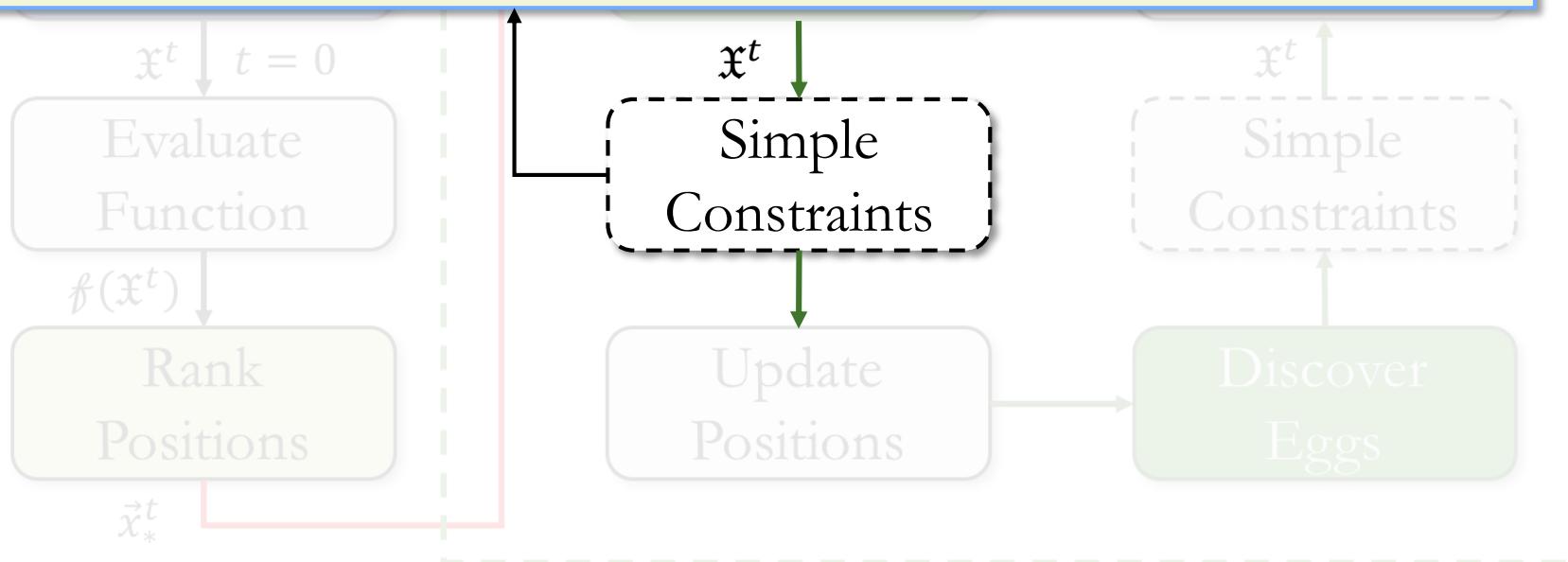
Simple Constraints Block

```
function Nests = simpleConstraints(Nests,lowerBoundaries,upperBoundaries)

% Check if solutions are inside the search space (feasible region)
check = Nests < lowerBoundaries;
Nests = ~check.*Nests + check.*lowerBoundaries;

check = Nests > upperBoundaries;
Nests = ~check.*Nests + check.*upperBoundaries;

endfunction
```



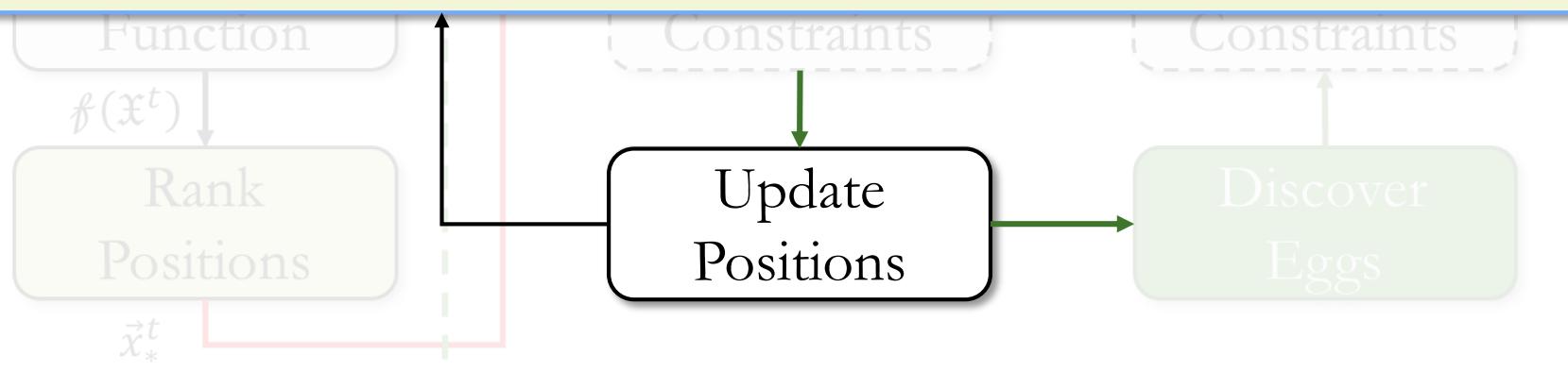
Update Positions Block

```
function [new_positions,new_fitness] =
    updatePositions(Na,Nd,the_function,fitness,positions,best_positions)

% Evaluate Function Block
evaluated_function      = evaluateFunction(Na,the_function,positions);

% Update best positions
condition      = evaluated_function < fitness;
new_positions   = (condition*ones(1,Nd)).*positions + ...
                  (!condition*ones(1,Nd)).*best_positions;
new_fitness     = min(evaluated_function,fitness);

endfunction
```



Discovering Eggs Block

$$\vec{x}_i^{t+1} = \vec{x}_i^t + \vec{u} \odot \mathcal{H}(\vec{u} - p_D) \odot (\vec{x}_j^t - \vec{x}_k^t),$$

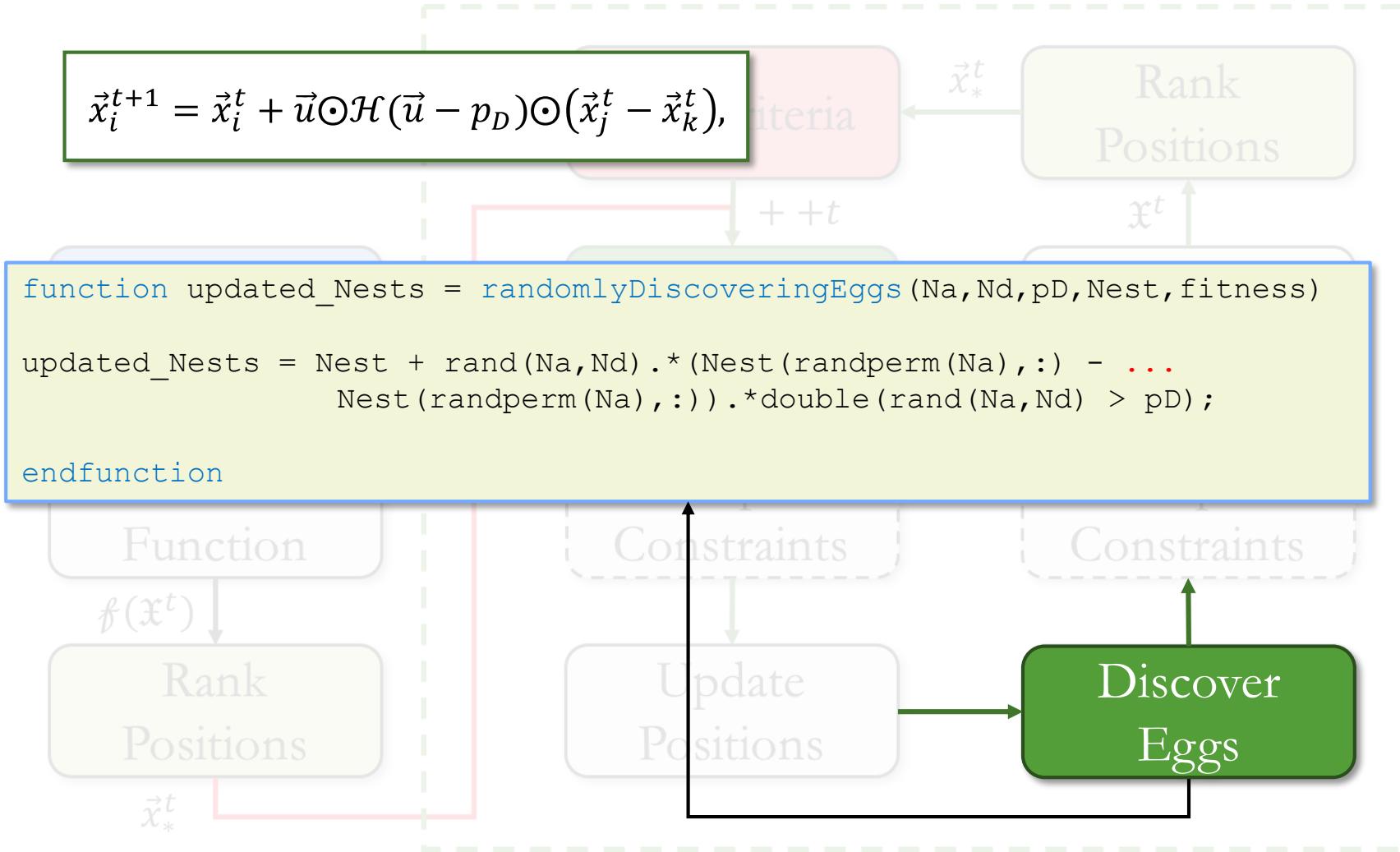
```

function updated_Nests = randomlyDiscoveringEggs(Na,Nd,pD,Nest,fitness)

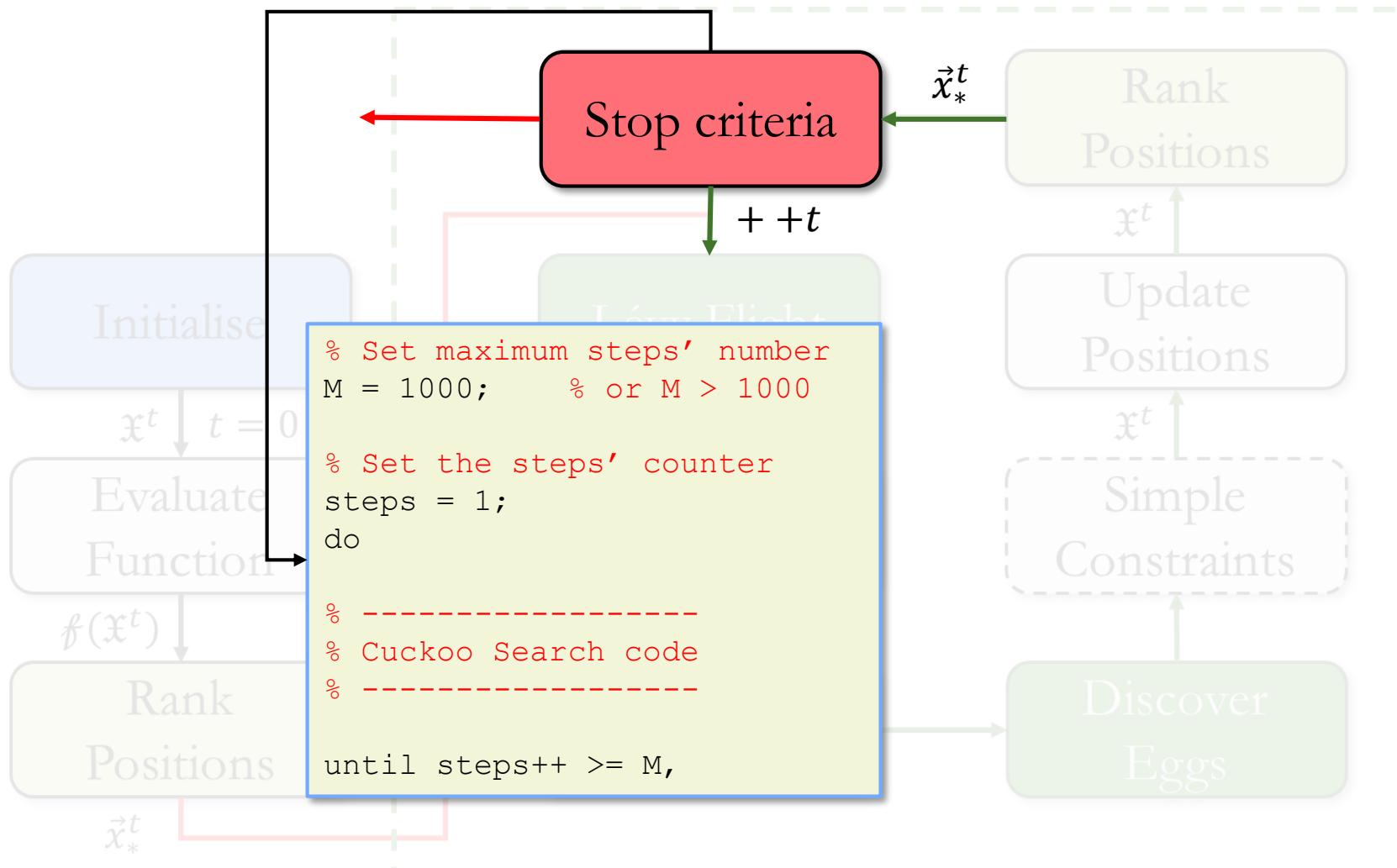
updated_Nests = Nest + rand(Na,Nd).* (Nest(randperm(Na),:)) - ...
                Nest(randperm(Na),:)).*double(rand(Na,Nd) > pD);

endfunction

```



Stop Criteria Block





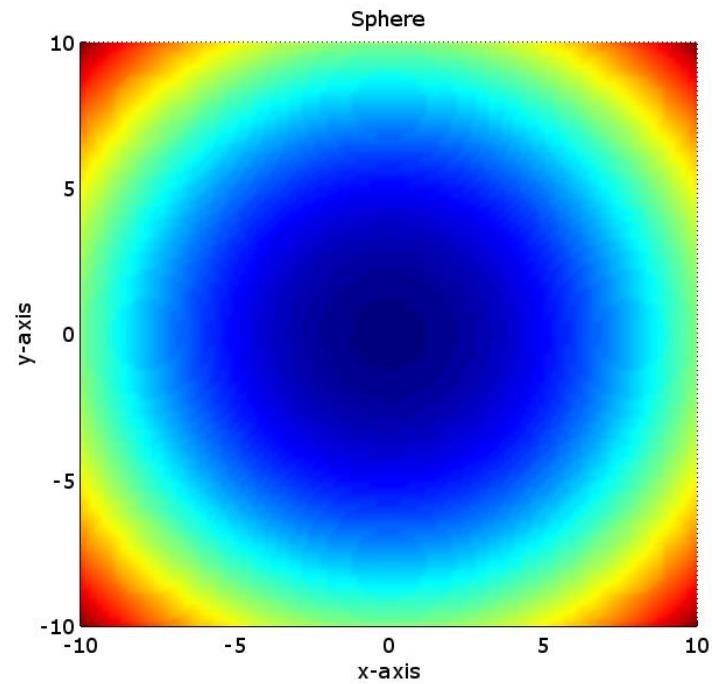
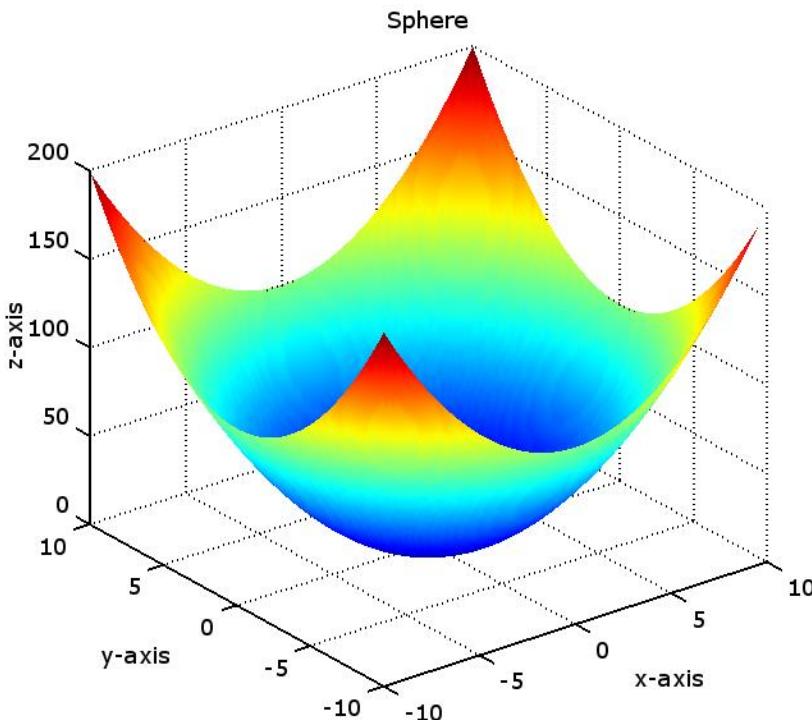
Testing Cuckoo Search

Image taken from: <http://thegraphicsfairy.com/wp-content/uploads/blogger/-X-9JNfjFPJs/TkR4zWBWYMI/AAAAAAAANuU/Bv2WRiYTok/s1600/nest%2Bvictorian%2Bimage%2Bgraphicsfairy009c.jpg>

Benchmark Functions: Sphere

$$f(\vec{x}) = \sum_{i=1}^{N_D} x_i^2$$

$$-10 \leq \vec{x} \leq 10 \quad \vec{x} = (x_1, x_2, \dots, x_{N_D})^T$$



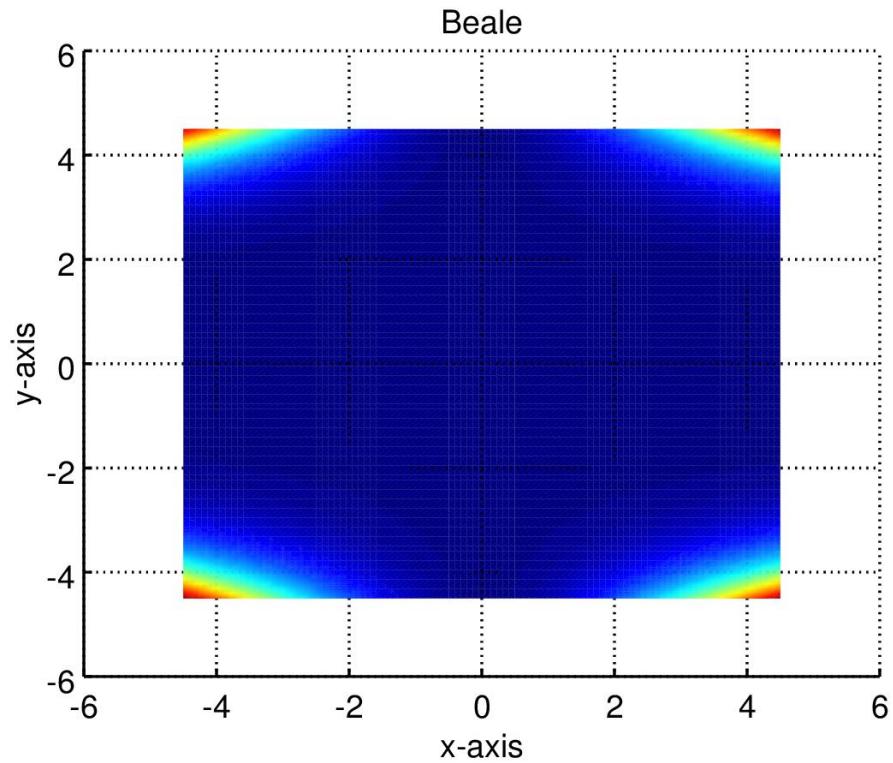
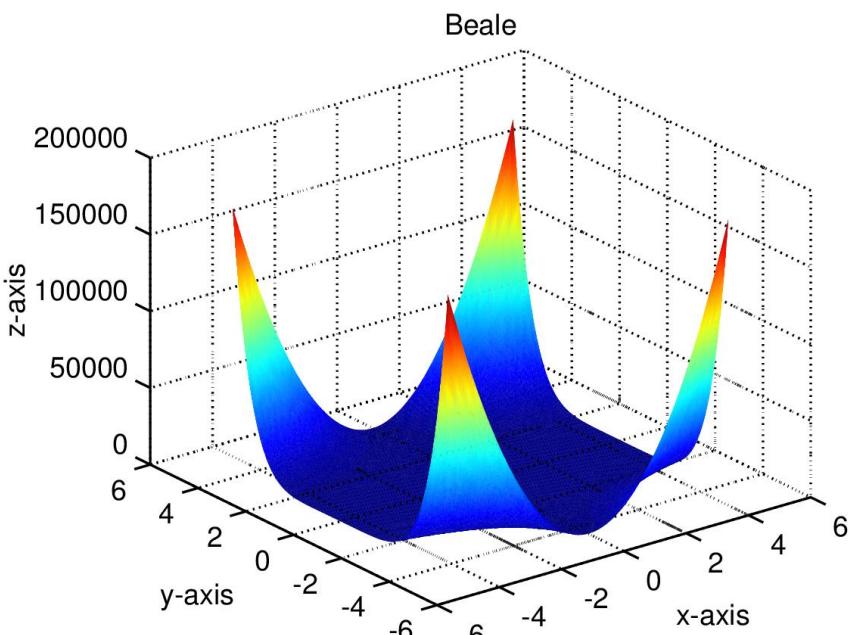
$$f_{min} = \min f(\vec{x}^*) = 0$$

$$\vec{x}^* = \arg \min f(\vec{x}) = (0, 0, \dots, 0)^T$$

Soneji, H., & Sanghvi, R. C. (2014). Towards the improvement of Cuckoo search algorithm. *International Journal of Computer Information Systems and Industrial Management Applications*, 6, 77–88. article.

Benchmark Functions: Beale

$$f(\vec{x}) = (1.5 - x_1 + x_1 x_2)^2 + (2.25 - x_1 + x_1 x_2^2)^2 + (2.625 - x_1 + x_1 x_2^3)^2 \quad \vec{x} = (x_1, x_2)^T$$
$$-4.5 \leq \vec{x} \leq 4.5$$



$$\vec{x}^* = \arg \min f(\vec{x}) = (3, 0.5)^T$$

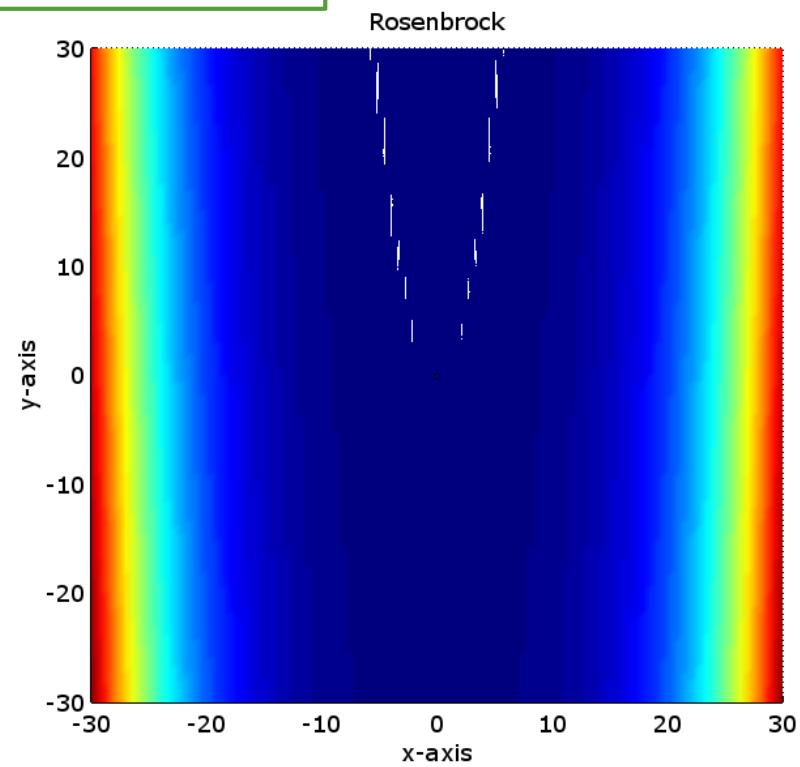
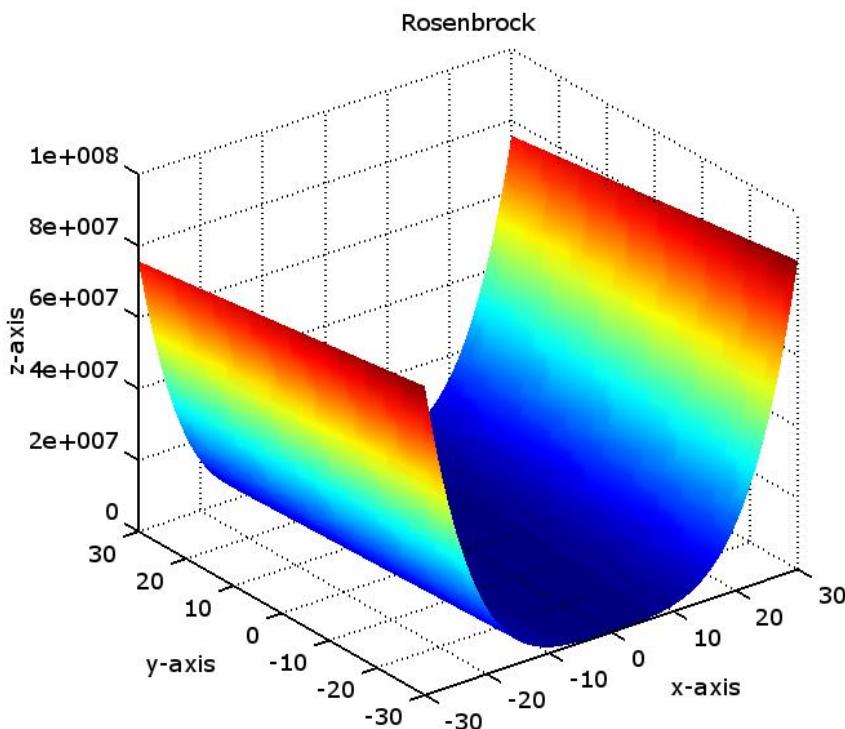
$$f_{\min} = \min f(\vec{x}^*) = 0$$

Jamil, M., & Yang, X. S. (2013). A literature survey of benchmark functions for global optimisation problems. *International Journal of Mathematical Modelling and Numerical Optimisation*, 4(2), 150. <http://doi.org/10.1504/IJMMNO.2013.055204>

Benchmark Functions: Rosenbrock

$$f(\vec{x}) = \sum_{i=1}^{N_D-1} \left[100(x_{i+1} - x_i^2)^2 + (x_i - 1)^2 \right]$$

$$\vec{x} = (x_1, x_2, \dots, x_{N_D})^T$$
$$-30 \leq \vec{x} \leq 30$$



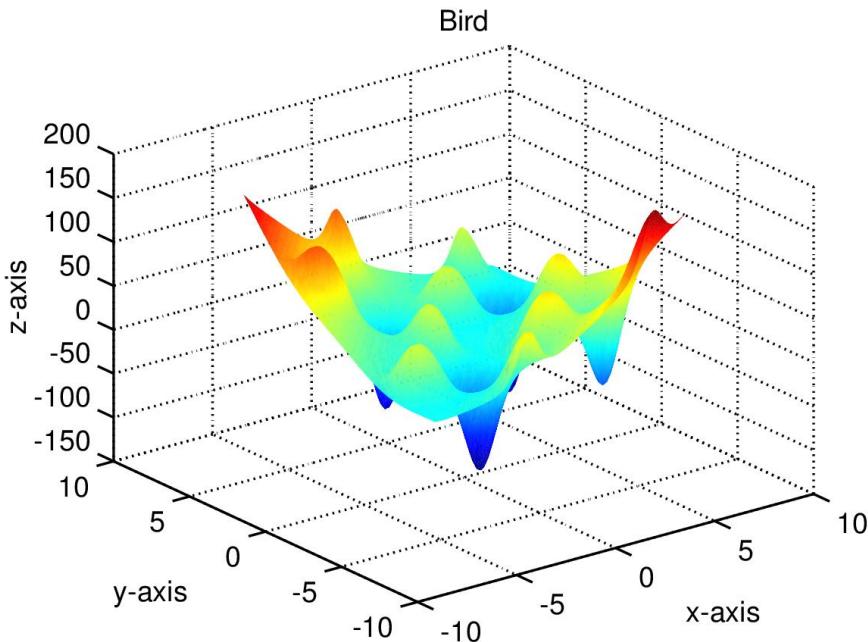
$$\vec{x}^* = \arg \min f(\vec{x}) = (1, 1, \dots, 1)^T$$

$$f_{\min} = \min f(\vec{x}^*) = 0$$

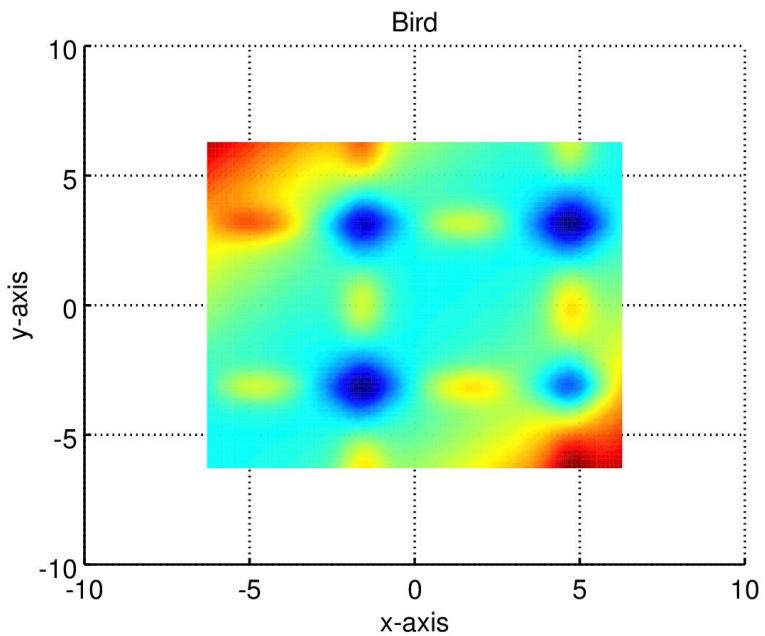
Jamil, M., & Yang, X. S. (2013). A literature survey of benchmark functions for global optimisation problems. *International Journal of Mathematical Modelling and Numerical Optimisation*, 4(2), 150. <http://doi.org/10.1504/IJMMNO.2013.055204>

Benchmark Functions: Bird

$$f(\vec{x}) = \sin(x_1) e^{(1-\cos(x_2))^2} + \cos(x_2) e^{(1-\sin(x_1))^2} + (x_1 - x_2)^2$$



$$-2\pi \leq \vec{x} \leq 2\pi \quad \vec{x} = (x_1, x_2)^T$$



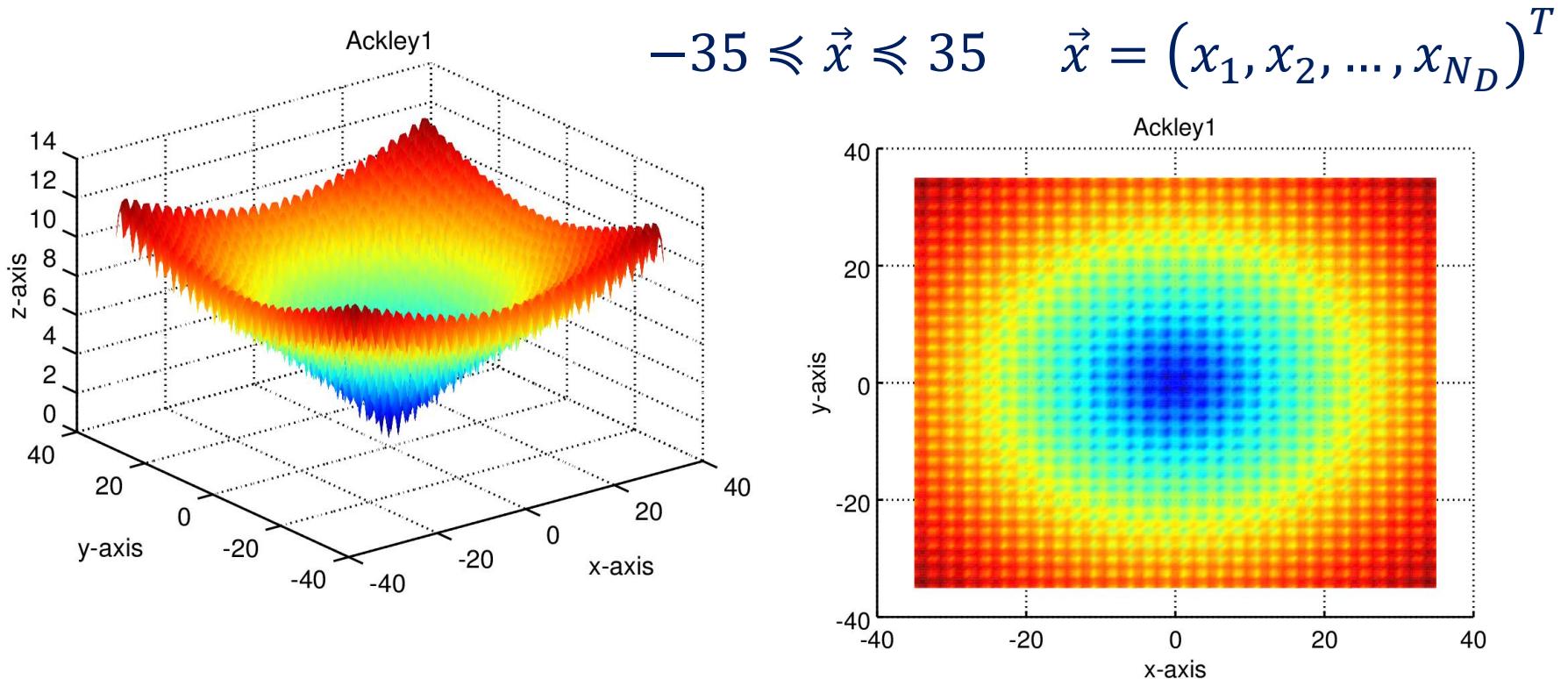
$$\begin{aligned}\vec{x}^* &= \arg \min f(\vec{x}) \\ &= (4.70104, 3.15294)^T \\ \text{or } &(-1.58214, -3.13024)^T\end{aligned}$$

$$\begin{aligned}f_{\min} &= \min f(\vec{x}^*) \\ &= -106.764537\end{aligned}$$

Jamil, M., & Yang, X. S. (2013). A literature survey of benchmark functions for global optimisation problems. *International Journal of Mathematical Modelling and Numerical Optimisation*, 4(2), 150. <http://doi.org/10.1504/IJMMNO.2013.055204>

Benchmark Functions: Ackley1

$$f(\vec{x}) = -20e^{-0.02\sqrt{\frac{1}{N_D}\sum_{i=1}^{N_D} x_i^2}} - e^{\frac{1}{N_D}\sum_{i=1}^{N_D} \cos(2\pi x_i)} + 20 + e$$



$$f_{min} = \min f(\vec{x}^*) = 0$$

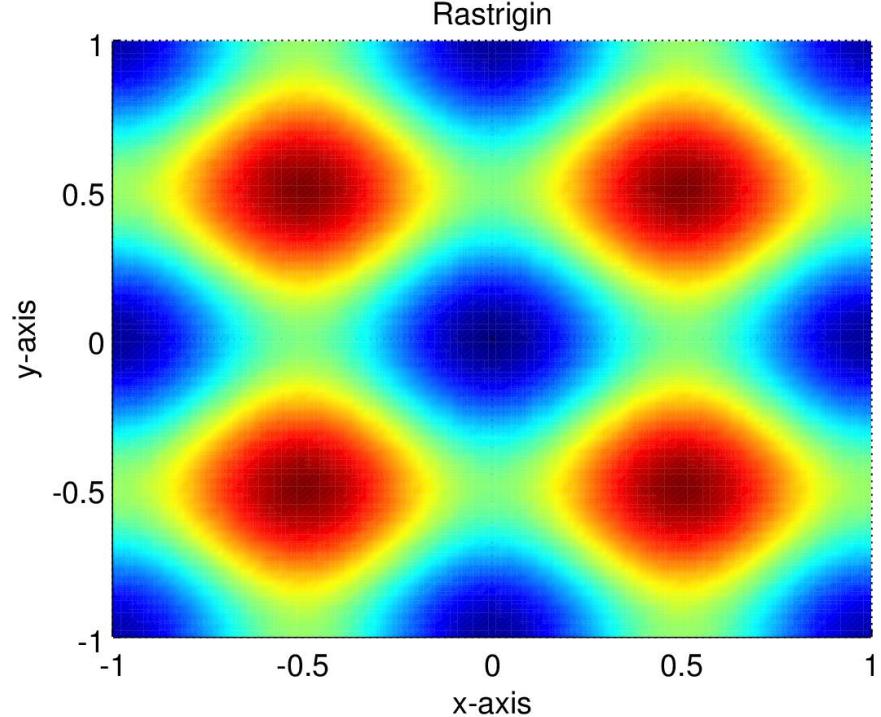
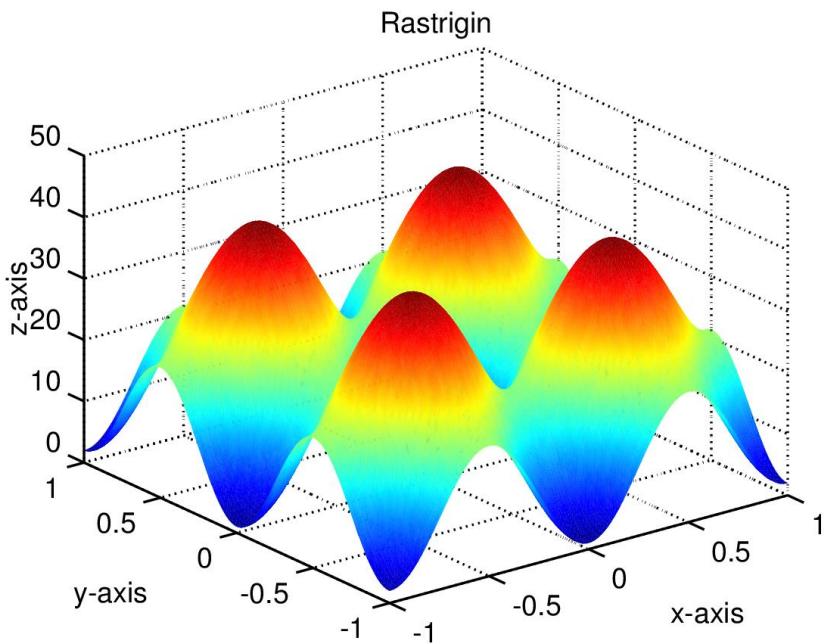
$$\vec{x}^* = \arg \min f(\vec{x}) = (0, 0, \dots, 0)^T$$

Jamil, M., & Yang, X. S. (2013). A literature survey of benchmark functions for global optimisation problems. *International Journal of Mathematical Modelling and Numerical Optimisation*, 4(2), 150. <http://doi.org/10.1504/IJMMNO.2013.055204>

Benchmark Functions: Rastrigin

$$f(\vec{x}) = 10N_D + \sum_{i=1}^{N_D} (x_i^2 - 10 \cos(2\pi x_i))$$

$$\begin{aligned} -0.9 \leq \vec{x} \leq 0.9 \\ \vec{x} = (x_1, x_2, \dots, x_{N_D})^T \end{aligned}$$



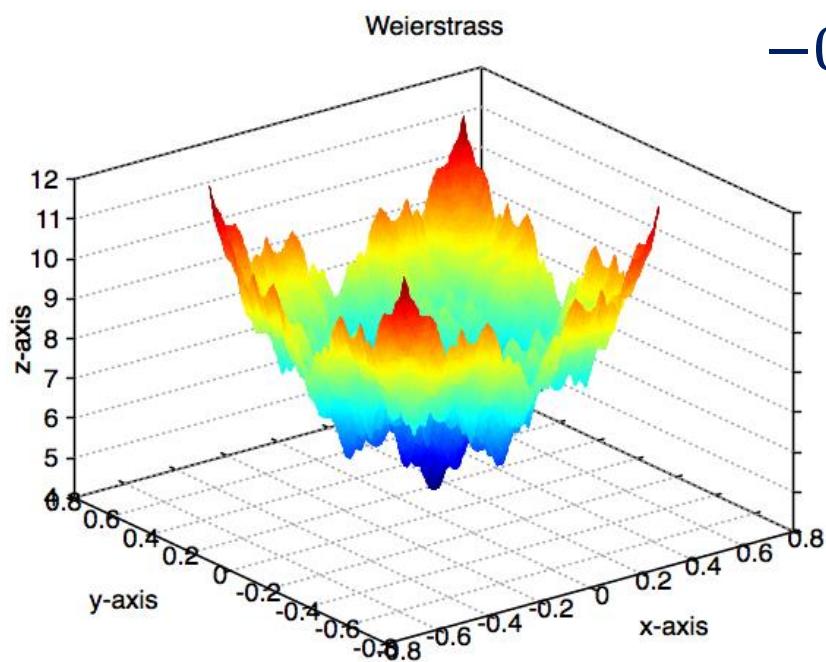
$$f_{min} = \min f(\vec{x}^*) = 0$$

$$\vec{x}^* = \arg \min f(\vec{x}) = (0, 0, \dots, 0)^T$$

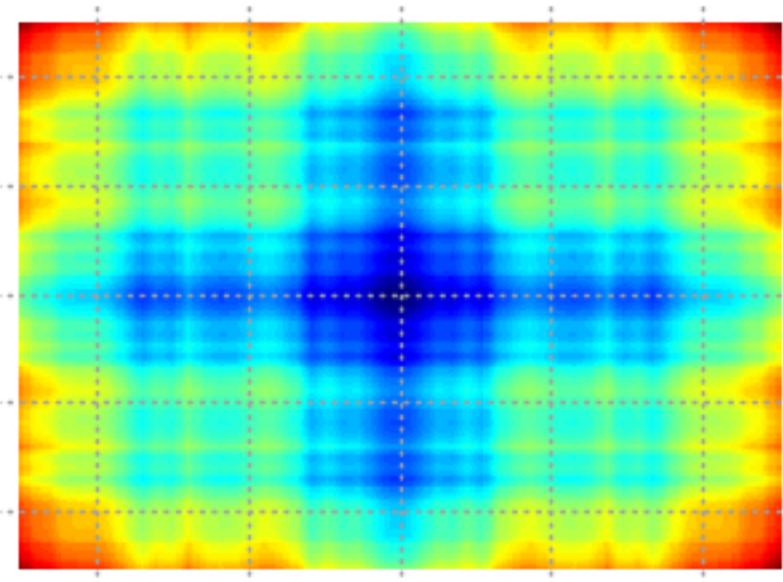
Soneji, H., & Sanghvi, R. C. (2014). Towards the improvement of Cuckoo search algorithm. *International Journal of Computer Information Systems and Industrial Management Applications*, 6, 77–88. article.

Benchmark Functions: Weierstrass

$$f(\vec{x}) = \sum_{i=1}^{N_D} \left[\sum_{k=0}^{k_{max}} a^k \cos(2\pi b^k(x_i + 0.5)) - N_D \sum_{k=0}^{k_{max}} a^k \cos(\pi b^k) \right]$$



$$-0.5 \leq \vec{x} \leq 0.5 \quad \vec{x} = (x_1, x_2, \dots, x_{N_D})^T$$



$$f_{min} = \min f(\vec{x}^*) = 4$$

$$\vec{x}^* = \arg \min f(\vec{x}) = (0, 0, \dots, 0)^T$$

Soneji, H., & Sanghvi, R. C. (2014). Towards the improvement of Cuckoo search algorithm. *International Journal of Computer Information Systems and Industrial Management Applications*, 6, 77–88. article.

Welded Beam Design Problem

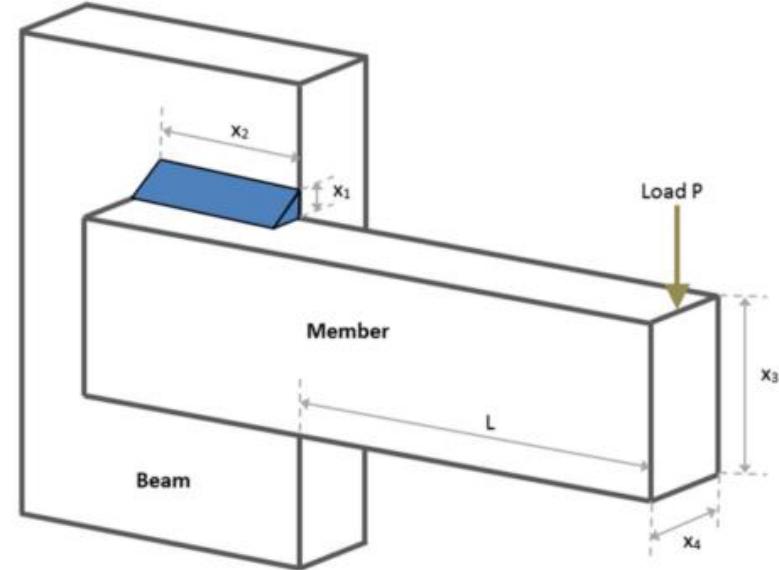
Find the minimum fabrication cost considering four design variables: the width ($w = x_1$) and length ($L = x_2$) of the welded area, the depth ($h = x_3$) and thickness ($d = x_4$) of the main beam [1,2].

Shear stress (τ), bending stress in the beam (σ), buckling load on the bar (P_c) and deflection of the beam (δ).

$$0.125 \leq x_1 \leq 5$$

$$0.1 \leq x_2, x_3, x_4 \leq 10$$

x_1 and x_2 are discrete variables
and integer multiples of 0.0065 in.



$$x^* = (0.205730, 3.470489, 9.036624, 0.205729)$$

$$\text{where } f(x^*) = 1.724852.$$

[1] YANG, X.-S. and DEB, S. 2010. Engineering Optimisation by Cuckoo Search. *Int. J. Mathematical Modelling and Numerical Optimisation*. Vol. 1, no. 4, pp. 330–343. ISSN 2040-3607.

[2] Rakhshani, H., & Rahati, A. (2016). Intelligent Multiple Search Strategy Cuckoo Algorithm for Numerical and Engineering Optimization Problems. *Arabian Journal for Science and Engineering*, <http://doi.org/10.1007/s13369-016-2270-8>

Welded Beam Design Problem

$$\min_x f(x) = 1.10471x_1^2x_2 + 0.04811x_3x_4(14.0 + x_2)$$

Subject to:

$$g_1(x) : \tau(x) - \tau_{\max} \leq 0,$$

$$g_2(x) : \sigma(x) - \sigma_{\max} \leq 0,$$

$$g_3(x) : x_1 - x_4 \leq 0,$$

$$g_4(x) : 0.10471x_1^2 + 0.04811x_3x_4(14.0 + x_2) - 5.0 \leq 0,$$

$$g_5(x) : 0.125 - x_1 \leq 0,$$

$$g_6(x) : \delta(x) - \delta_{\max} \leq 0,$$

$$g_7(x) : P - P_c(x) \leq 0,$$

$$P = 6000 \text{lb}, L = 14 \text{ in}, E = 30 \times 10^6 \text{ psi}, G = 12 \times 10^6$$

$$\text{psi}, \tau_{\max} = 13,600 \text{ psi}, \sigma_{\max} = 30,000 \text{ psi},$$

$$\delta_{\max} = 0.25 \text{ in.}$$

$$\tau(x) = \sqrt{(\tau')^2 + 2\tau'\tau'' \frac{x_2}{2R} + (\tau'')^2},$$

$$\tau' = \frac{P}{\sqrt{2}x_1x_2},$$

$$\tau'' = \frac{MR}{J},$$

$$M = P \left(l + \frac{x_2}{2} \right),$$

$$R = \sqrt{\frac{x_2^2}{4} + \left(\frac{x_1 + x_3}{2} \right)^2},$$

$$J = 2 \left\{ \sqrt{2}x_1x_2 \left[\frac{x_2^2}{12} + \left(\frac{x_1 + x_3}{2} \right)^2 \right] \right\}$$

$$\sigma(x) = \frac{6PL}{x_4x_3^2},$$

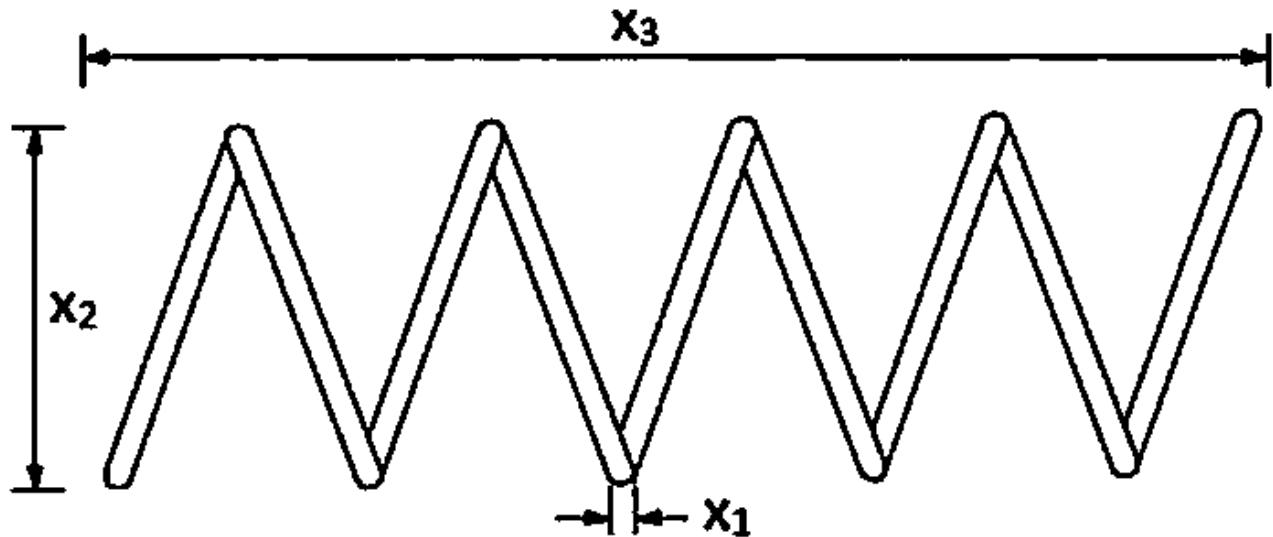
$$\delta(x) = \frac{4PL^3}{Ex_3^3x_4},$$

$$P_c(x) = \frac{4.013E\sqrt{\frac{x_3^2x_4^6}{36}}}{L^2} \left(1 - \frac{x_3}{2L}\sqrt{\frac{E}{4G}} \right),$$

[1] YANG, X.-S. and DEB, S. 2010. Engineering Optimisation by Cuckoo Search. *Int. J. Mathematical Modelling and Numerical Optimisation*. Vol. 1, no. 4, pp. 330–343. ISSN 2040-3607.

Spring Design Problem

Find the minimum weight of a spring. Its design variables are: the wire diameter (x_1), the mean coil diameter (x_2) and the number of active coils (x_3) [1].



$$0.05 \leq x_1 \leq 1$$

$$0.25 \leq x_2 \leq 1.3$$

$$2 \leq x_3 \leq 15$$

Best solution:

$$x^* = (0.051690, 0.356750, 11.287126)$$

$$\text{where } f(x^*) = 0.012665.$$

- [1] Rakhshani, H., & Rahati, A. (2016). Intelligent Multiple Search Strategy Cuckoo Algorithm for Numerical and Engineering Optimization Problems. *Arabian Journal for Science and Engineering*. <http://doi.org/10.1007/s13369-016-2270-8>
- [2] Cagnina, L. C., Esquivel, S. C. & Coello Coello, C. A. Solving engineering optimization problems with the simple constrained particle swarm optimizer. *Inform.* **32**, 319–326 (2008).

Spring Design Problem

$$\min_x f(x) = (x_3 + 2)x_2x_1^2$$

Subject to:

$$g_1(x) : 1 - \frac{x_2^3 x_3}{71,785 x_1^4} \leq 0,$$

$$g_2(x) : \frac{4x_2^2 - x_1 x_2}{12,566 (x_2 x_1^3 - x_1^4)} + \frac{1}{5108 x_1^2} - 1 \leq 0,$$

$$g_3(x) : 1 - \frac{140.45 x_1}{x_2^2 x_3} \leq 0,$$

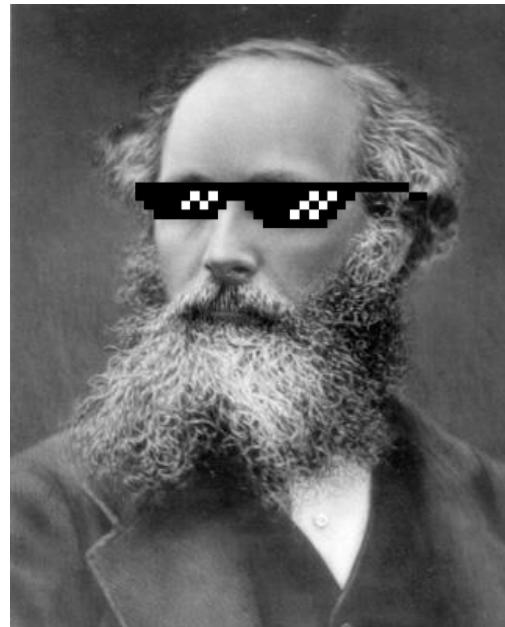
$$g_4(x) : \frac{x_1 + x_2}{1.5} - 1 \leq 0,$$

$$0.05 \leq x_1 \leq 1$$

$$0.25 \leq x_2 \leq 1.3$$

$$2 \leq x_3 \leq 15$$

- [1] YANG, X.-S. and DEB, S. 2010. Engineering Optimisation by Cuckoo Search. *Int. J. Mathematical Modelling and Numerical Optimisation*. Vol. 1, no. 4, pp. 330–343. ISSN 2040-3607.
- [2] Cagnina, L. C., Esquivel, S. C. & Coello Coello, C. A. Solving engineering optimization problems with the simple constrained particle swarm optimizer. *Inform.* **32**, 319–326 (2008).



Thanks!