Lioness Algorithm for Finding Optimal Design of Experiments

Hongzhi Wang, Qian Xiao and Abhyuday Mandal

University of Georgia, Athens, GA, USA

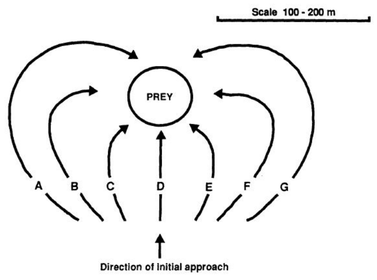
Introduction

- Design of experiments has wide applications in scientific and industrial fields. Optimal designs are highly demanded, but identifying them are not easy as theoretical results only exist for special cases.
- Optimization algorithms are widely used to optimize certain optimality criteria to identify optimal designs, but one algorithm usually only works for one design space (either continuous or discrete design space).
- We propose a brand new population-based meta-heuristic algorithm that was inspired by the hunting behaviors of lionesses pride, and we named it lioness algorithm (LA).
- LA is compatible with both continuous and discrete design space and it requires very little memory usage and CPU time.

Lioness Algorithm for Optimal Designs

Inspiration and General Framework

- In a lion pride, male lions are responsible for protecting their territory, while lionesses take care of hunting. Lionesses usually hunt in a group manner. The mechanics of lioness algorithm (LA) was inspired by the hunting process of lionesses.
- Stander (1992) introduced three terminologies to describe the relative positions between lionesses and prey: centre, left wing, and right wing. The centre lionesses (C, D, and E in the figure) are at the closest position toward to prey, and they charge prey at the first place. Left wing (A and B) and right wing (F and G) lionesses encircle prey and remain stealth to prevent prey from running away.



- We use the following notations to demonstrate the algorithm: $\mathbf{X}$ denotes a design matrix, k denotes the number of factors, and n denotes the number of runs for each factor. LA starts with generating a certain (user-defined) number of random n by k design candidates.
- The second step of LA is to evaluate all the design candidates according to an optimality criterion. We denote the global best as centre solution (CS), the second global best as left wing (LW), and third global best as right wing (RW).
- The third step of LA is to create a new collection of design candidates via CS, LW, and RW. Let $X_1^{new}, \ldots, X_m^{new}$ denote the new collection of design candidates, where $m \equiv 6k + 3$. The first three new candidates will always be CS, LW, and RW, i.e. $X_1^{new} = \text{CS}$, $X_2^{new} = \text{LW}$, and $X_3^{new} = \text{RW}$.
- Let $C = 4$, for each column j of X^{new}, where $j = 1, \ldots, k$: $X_C^{new} = \text{CS}$ and replace the j^{th} column of X_C^{new} with the j^{th} column of LW then $C = C + 1$; $X_C^{new} = \text{CS}$ and replace the j^{th} column of X_C^{new} with the j^{th} column of RW then $C = C + 1$.
- Similarly, for each column j of X^{new}: $X_C^{new} = \text{LW}$ and replace the j^{th} column of X_C^{new} with the j^{th} column of CS then $C = C + 1$; $X_C^{new} = \text{LW}$ and replace the j^{th} column of X_C^{new} with the j^{th} column of RW then $C = C + 1$. After that, repeat exactly the same procedure when $X_C^{new} = \text{RW}$.
- The last step is to enforce a mechanism for avoiding local optima. This step varies between designs with continuous and discrete space. When a stopping condition is met, evaluate all the new design candidates and return the CS as the optimal design.

Designs with Discrete Space

- Here we illustrate how to apply the general framework of LA to identify optimal designs over discrete space, by taking Latin hypercube designs (LHDs, McKay et al. (1979)) as an example.
- LHDs are the most popular designs for computer experiments, and an LHD can be viewed as an n by k design matrix with each column being a random permutation of $1, \ldots, n$. To identify optimal LHDs, LA starts with generating random n by k LHDs.
- The second step of LA is to evaluate all the design candidates according to an optimality criterion, which can be maximin distance criterion, maximum projection criterion, or maximum absolute correlation criterion (Morris & Mitchell (1995), Joseph et al. (2015), Georgiou, (2009)), and then determine CS, LW, and RW.
- The third step of LA should be exactly the same as that in the general framework.
- We enforce the following mechanism for avoiding local optima in LHD case: for all the new design candidates except X_1^{new}, for every column from each candidate, draw a random number z from Uniform$(0, 1)$ and if $z < p$, where p is an user-defined probability, two random elements within current column will be switched. When a stopping condition is met, evaluate all the new design candidates and return the CS as the optimal design.

Designs with Continuous Space

- Now all the factors are assumed to have continuous domain. Let $[LB_j, UB_j]$ denote the design space of the j^{th} variable of a design candidate, where $j = 1, \ldots, k$, the n observations in the j^{th} column of a random design candidate are generated through n random draws from Uniform(LB_j, UB_j).
- LA starts with generating an user-defined number of random designs candidates according to above procedure, and then evaluate them according to an optimality criterion, i.e. the D-optimality criterion. After CS, LW, and RW are determined, the third step should be exactly the same as that in the general framework.
- We enforce the following mechanism for avoiding local optima in designs with continuous space: for all the new design candidates except X_1^{new}, for every element from each candidate, draw a random number z from Uniform$(0, 1)$ and if $z < p$, where p is given by

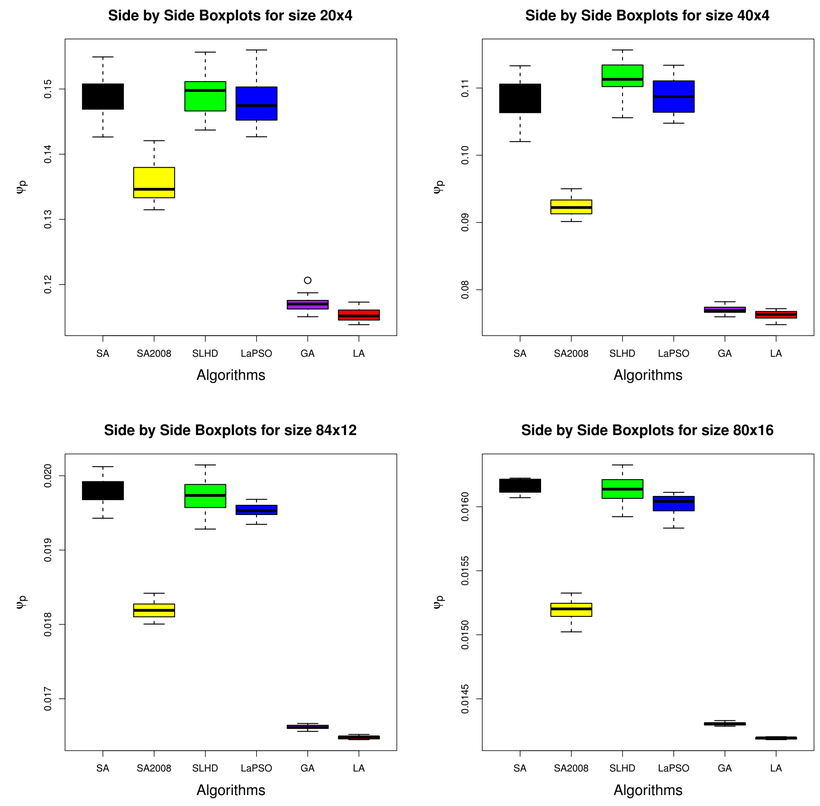
$$\frac{(\text{total number of iterations} - \text{current iteration} + 1)}{\text{total number of iterations}},$$

each element will be updated with the following equation $X_{(ij)}^{new} = X_{(ij)}^{new} \pm \alpha \times X_{(ij)}^{new}$, where $\pm$ is a random draw from $+$ and $-$ with 50% chance for each and α is turning parameter for controlling how big the change would be.

Performance of the Proposed Algorithm

Numerical Results on Latin Hypercube Designs

Maximin Distance LHDs Results



- Among all the six algorithms, LA has the smallest ϕ_p values.
- When design size increases, the advantage become more obvious.

Orthogonal and Nearly Orthogonal LHDs Results

$n \times k$	Criteria	SA	LaPSO	GA	LA
20×2	ave$(\lvert q \rvert)$	0	0	0	0
	max$\lvert q \rvert$	0	0	0	0
20×4	ave$(\lvert q \rvert)$	0.0198	0.0035	0.0020	**0.0008**
	max$\lvert q \rvert$	0.0406	0.0090	0.0030	**0.0015**
40×4	ave$(\lvert q \rvert)$	0.00891	0.00147	0.00041	**0.00038**
	max$\lvert q \rvert$	0.02045	0.00338	0.00094	**0.00056**
40×8	ave$(\lvert q \rvert)$	0.0669	0.0247	0.0063	**0.0046**
	max$\lvert q \rvert$	0.1811	0.0456	0.0114	**0.0090**
80×8	ave$(\lvert q \rvert)$	0.0492	0.0138	0.0020	**0.0014**
	max$\lvert q \rvert$	0.1102	0.0240	0.0038	**0.0033**
84×12	ave$(\lvert q \rvert)$	0.0660	0.0645	0.0151	**0.0104**
	max$\lvert q \rvert$	0.1560	0.1594	0.0283	**0.0208**

- Among all the six algorithms, LA has the smallest average absolute correlation criterion (ave$(\lvert q \rvert)$) and the maximum absolute correlation criterion (max$\lvert q \rvert$).
- For the case of 20×2, all the algorithms identified orthogonal LHDs (all correlations are 0). For the rest cases, LA identified nearly orthogonal LHDs with small column-wise correlations.

Numerical Results on Designs with Continuous Input

- The Arrhenius equation is widely used in chemical experiments and the model is define as $Er = A\exp(-B/T)$, where Er denotes the expectation of the reaction time, A is the Arrhenius frequency parameter, B is the activation temperature, and T is the design matrix to estimate A and B.
- Stokes et al. (2020) proposed a two-points locally D-optimal design for the Arrhenius equation with the following D-optimality criterion: $-\log|\sum_{i=1}^{2} p_i(\nabla Er(T_i))(\nabla Er(T_i))^T|$, where p_i is weight and ∇Er is the gradient of the mean function, which is given by $\nabla Er = (\exp(-B/T), A\exp(-B/T)/T)^T$. A design T is the locally D-optimal design if it minimizes the criterion.
- We ran LA and other popular optimization algorithms such as PSO (Kennedy & Eberhart, 1995), GA (Holland, 1975), and GWO (Mirjalili et al., 2014) to search for the D-optimal design. All the algorithms found the D-optimal design, while LA has the lowest CPU time.

	PSO	GA	GWO	LA
D-efficiency	100%	100%	100%	100%
CPU	0.2971	0.4808	0.6063	0.1746

R Package

We developed the R package `LA` available on the Comprehensive R Archive Network (`https://cran.r-project.org/web/packages/LA/index.html`), which contains the LA and its contents in this work.

Reference

Wang, H.; Xiao, Q. and Mandal, A. (2021), "Lioness Algorithm for Finding Optimal Designs", to be submitted.