Performance Evaluation of Bat Algorithm to Solve Deterministic and Stochastic Optimization Problems

Ratnaji Vanga, Manu K. Gupta and J. Venkateswaran

Industrial Engineering and Operations Research
IIT Bombay

ISCI - 2013

Outline

- Objective
- Bat Algorithm
- Different Optimization Problems
- Implementation Details
- Results and Discussion
- Conclusions and Future work

Objective

- There are many approaches to solve deterministic optimization problems while very few methods were implemented in stochastic settings.
- Propose a new simulation based optimization approach to solve the non-linear stochastic optimization problems.
- Interface a meta-heuristic in simulation based optimization.
- Bat algorithm is one such newly developed meta-heuristic.
- Study the effect of dimensionality vs stochasticity.

Bat Algorithm

- Nature inspired meta-heuristic optimization algorithm.
- First proposed by *Yang (2010)*.
- Based on echolocation behaviour of bats.
- Applied on continuous optimization problem by *Parpinenli and Lopes (2011)*.
- Solved numerical optimization problems by *Tsai et al. (2011)*.
- Used to solve multi objective optimization problems by *Yang (2011)*.
- Applied for multi-stage, multi-machine, multi-product scheduling by *Musikapun and Pongcharoen (2012)*.

Bat Algorithm Contd...

Three Rules of Bat Algorithm

- All bats use echolocation to sense distance, and they also know the difference between food prey and background barriers in some magical way.
- Bats fly randomly with velocity v_i at position x_i with a fixed frequency f_{min}, varying wavelength λ and loudness A_0 to search for prey. They can automatically adjust the wavelength (or frequency) of their emitted pulses and adjust the rate of pulse emission $r_i \in [0, 1]$, depending on the proximity of their target.
- Although the loudness can vary in many ways, assume that the loudness varies from a large (positive) A_0 to a minimum constant value A_{min}.

Bat Algorithm Contd...

- Movement of virtual bat is simulated by following equations:

$$f_i = f_{min} + (f_{max} - f_{min})\beta$$
$$v_i^t = v_i^{t-1} + (\vec{x}^{t-1} - \vec{x}_*)f_i$$
$$\vec{x}_i^t = \vec{x}_i^{t-1} + v_i^t$$

Modified Bat Algorithm

- To explore locally in both the directions:

$$\vec{x}^t = \vec{x}_i^{t-1} + \vec{v}_i^t * e, \;\; e \in [-1, 1]$$

- Keep track of only one best solution.
- Elimination of random generation of new solution.
- Modifications on conditions for updating r_i and A_i.

Bat Algorithm Mechanism

Step 1. Initialization: Randomly spread the bats into the solution space.

Step 2. Move the bats by predefined rules. Generate a random number. If it is greater than the fixed pulse emission rate, move the bat by the random walk process.

Step 3. Evaluate the fitness of the bats and update the global near best solution.

Step 4. Check the termination condition to decide whether go back to step 2 or terminate the program and output the near best result.

Modified Bat Algorithm

Require: parameters n, α, γ, Number of iterations (N), lb, ub

Initialize the bats population $\vec{x_i}$ randomly, $t = 0$, f_i, r_i^0, v_i^0 and A_i.
Compute fitness of each bat $F(\vec{x_i})$, $\forall i = 1, 2, \ldots n$ and find the current best $\vec{x}_*$
while (t < N) **do**
    Itebest $\leftarrow$ large value
    for i = 1 **to** n **do**
        Generate new solutions by adjusting frequency, velocity and location
        if $(rand > r_i)$ **then**
            Generate a local solution around the selected best solution:$\vec{x_i}^t = \vec{x}_* + \epsilon \bar{A}$, $\epsilon \in unif[-1, 1]$
        end if
        if $(\vec{x_i}^t \notin [lb, ub])$ **then**
            Generate a random solution in the range [lb, ub]
        end if
        if $rand < A_i$ & $F(\vec{x}_i^t) < F(\vec{x}_i^{t-1})$ **then**
            Increase r_i: $r_i = r_i^0(1 - exp(-\gamma t))$ and Decrease A_i: $A_i = \alpha A_i$
        end if
        Update iteration best (itebest)
    end for
    Find the current best $\vec{x}_*$, $t = t + 1$
end while
Post process results and visualisation

Optimization problems

Rosenbrock: D dimensional Rosenbrock function is defined by

$$f(x_1, x_2, \cdots, x_D) = \sum_{d=1}^{D-1} 100(x_{d+1} - x_d^2)^2 + (x_d - 1)^2 \qquad (1)$$

The above function is for deterministic case and stochastic version of given function is as follows:

$$f(x_1, x_2, \cdots, x_D) = \sum_{d=1}^{D-1} 100r(x_{d+1} - x_d^2)^2 + (x_d - 1)^2 \qquad (2)$$

where r is a random variable.

Global Optima

It has a global minimum at $(x_1, x_2, \cdots, x_D) = (1, 1, \cdots, 1)$ where $f(x_1, x_2, \cdots, x_D) = 0$.

Rosenbrock

The global minimum is inside a long, narrow, parabolic shaped flat valley. To find the valley is trivial. To converge to the global minimum, however, is difficult.

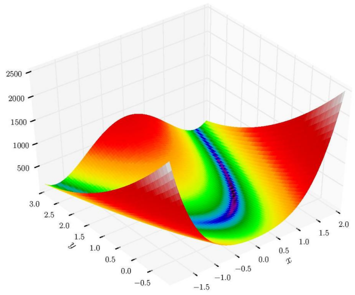

Figure: Rosenbrock Function 3-D plot

Rastrigin

Rastrigin D dimensional Rastrigin function is defined by

$$f(x_1, x_2, \cdots, x_D) = 10D + \sum_{i=1}^{D} \left[x_i^2 - 10\cos 2\pi x_i\right] \tag{3}$$

The above function is for deterministic case and stochastic version of given function is as follows:

$$f(x_1, x_2, \cdots, x_D) = 10D + \sum_{i=1}^{D} r\left[x_i^2 - 10\cos 2\pi x_i\right] \tag{4}$$

where r is a random variable.

Global Optima

It has a global minimum at $(x_1, x_2, \cdots, x_D) = (0, 0, \cdots, 0)$ where $f(x_1, x_2, \cdots, x_D) = 0$.

Rastrigin

Rastrigin function has many local minima i.e. the "valleys" in the plot. However, the function has just one global minimum

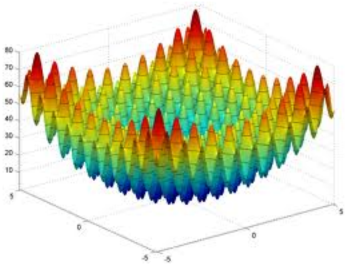

Figure: Rastrigin Function 3-D plot

Simulation based optimization algorithm

```
Initial_experiment_setup
Call initializeAlgo()
while stopping condition not met, say, iterations ≤ 10*M do
    for current_replication = 1 ... Max_replication do
        Obtain the new solution by calling setVariable() {Before
        simulation run}
        Update the solution in the Anylogic Model
        Simulate the model
        Record the objective value obtained from simulation
        model{After_simulation_run}
        After_Iteration
    end for
    Send the mean objective value using functionEval(objective)
end while
```

Experimentation details

Experiments:

We studied following 12 cases.

$$\left\{ \begin{array}{l} Rosenbrock \\ Rastrigin \end{array} \right\} \times \left\{ \begin{array}{l} Deterministic \\ Stochastic - r \in U[0,1] \\ Stochastic - r \in U[0,10] \end{array} \right\} \times \left\{ \begin{array}{l} 2-D \\ 6-D \end{array} \right\}$$

Each function is tested with 10 different seeds for 100000 evaluations.

Measure of Performance

- Quality of solution
- Execution time
- Number of Iterations

Parameter Setting

- Static factors: $\alpha = 0.9$ and $\gamma = 0.9$
- Dynamic factors: n, f_{max}, $\bar{v}_i^0$, r_i^0

	n	f_{max}	$\bar{v}_i^0$	r_i^0
Rosenbrock - Deterministic	1	0.01	0.4	0.4
Rastrigin - Deterministic	2	0.01	0.5	0.5
Rosenbrock - Stochastic	1	0.01	0.4	0.4
Rastrigin - Stochastic	2	0.01	0.5	0.5

Table: Parameter setting based on initial experiments

Rosenbrock Deterministic - 2D

						Z≤0.05			Z≤0.1		
	Iteration	Time(ms)	Zbest	X	Y	Iteration	Time	Z	Iteration	Time	Z
Min	37401	152	0.0004	0.9686	0.9379	28	0	0.0056	28	0	0.028
Max	99982	377	0.0058	1.0762	1.1588	9845	42	0.0464	1059	14	0.0993
Avg	72610	289.6	0.0024	1.00504	1.0187	1213.5	12	0.02951	252.2	8.5	0.06402

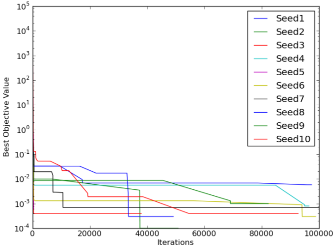

Figure: Iterations-vs-best objective

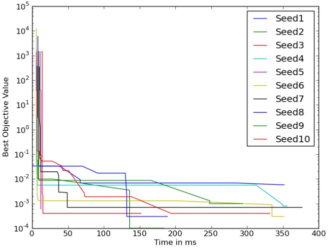

Figure: Time-vs-best objective

Summary tables

2D Case

Function model	Iteration	Time(ms)	Zbest	X	Y
Rastrigin-Determinstic	68868.9	275.2	0.07365	-0.00233	0.00233
Rastrigin-UNIF(0,1)	76105.8	2513.3	0.08878	-0.00511	0.10189
Rastrigin-UNIF(0,10)	63672.9	2841.7	0.05187	-0.10462	0.403
Rosenbrock-Deteminstic	72610	289.6	0.0024	1.00504	1.0187
Rosenbrock-Unif(0,1)	71314.9	2167.3	0.00057	1.00153	1.0036
Rosenbrock-Unif(0,10)	84844.4	3439.7	0.00213	1.00126	1.00797

6D Case

Function	Iteration	Time	Zbest	X1	X2	X3	X4	X5	X6
Ras-D	69539.8	2775.8	16.8201	-0.43278	0.75327	-0.11037	0.01637	0.27124	0.19124
Ras-U(0,1)	84743.7	1837.1	4.54323	0.49579	-1.17359	-0.0986	0.61771	0.19573	1.78118
Ras-U(0,10)	71676.7	1414.1	10.78911	0.00788	-0.89448	1.30638	1.77143	2.10919	-1.7045
Ros-D	50995.8	1048	35.20167	-0.20311	0.45405	0.43338	0.38025	0.21873	0.3118
Ros-U(0,1)	54793.8	1144.1	7.52775	0.81628	0.68104	0.45017	0.71196	0.06243	0.4085
Ros-U(0,10)	53361.8	674.4	15.31221	0.43724	0.3053	-0.19337	0.50185	0.17153	0.9007

Observations

- In 2-D case, solution improves when small variance is present for Rosenbrock function while it deteriorates for Rastrigin function.
- In 2-D case with higher variance, the performance of BA for Rosenbrock function deteriorates while it improves for Rastrigin function.
- In 6-D case, solution improves when variance is small and deteriorates when variance is high.
- It implies that there should be some optimal level of stochasticity in the system to get the optimal solutions.
- Best solution obtained with 6-D case is 4.51 while in 2-D case we always obtained the convergence level of 0.1.
- BA algorithm performs well for less number of bats.

Summary

- A simulation based optimization approach for non-linear stochastic problems is proposed.
- Modified BA is proposed and interfaced with anylogic model.
- Tested dimensionality vs stochasticity for rosenbrock and rastrigin functions.
- Results shows that BA initially converges very fast for all tested conditions.

Future work

- This approach can be applied to other optimization problems like queueing models and (s, S) inventory system.
- Other meta-heuristics (genetic algorithm, particle swarm optimization etc.) can be implemented in this setting and results can be compared.

P. Musikapun and P. Pongcharoen.
Solving multi-stage multi-machine multi-product scheduling problem using bat algorithm.
In *International Conference on Management and Artificial Intelligence*. IACSIT press, 2012.

R. Parpinelli and H. Lopes.
New inspirations in swarm intelligence: a survey.
Int. J. Bio-Inspired Computation, 2011.

P. W. Tsai, J. S. Pan, B. Y. Liao, M. J. Tsai, and V. Istanda.
Bat algorithm inspired algorithm for solving numerical optimization problems.
Applied Mechanics and Materials, 148-149:134–137, 2011.

X. S. Yang.
A new mataheuristic bat inspired algorithm.
Studies in Computational Intelligence, 2010.

X. S. Yang.
Bat algorithm for multiobjective optimization.
international journal of Bio-Inspired Computation, 3(5):267–274, 2011.

Thank you!!!