**SCA: A Sine Cosine Algorithm for solving optimization problems**

R09546014工工一洪翊倫

**Section1: Introduction**

首先，作者提及隨機優化演算法有效避免陷入局部最佳，比傳統的優化演算法更具優勢。動機根據N.F.L. Thm.,作者提出SCA algorithm,用簡單的數學函數就可以找到嘗試找到最佳解，何樂不為；並且透過許多標竿問題與機翼的升力問題證明此演算法是有用的。

**Section2: Methods and Problems**

與PSO algorithm非常類似，但是不考慮個別particle走過的最佳解，所以只考慮群體的移動。依據sin(x) , cos(x)的值域可以達到exploitation、exploration的效果。作者的迭代是等比率縮小移動步伐，我把SN ratio考慮進去，實驗數據觀察到略優於作者演算法的效果。

**SN Ratios:**

= 10

where is the square of mean of objective value of all particles

and is the square of sample variance of objective value of all particles

根據隨機最佳化的精神，應該忽略sample variance，因為面對許多有區域最佳解的問題，如果考慮解的分布則很可能全部陷進區域最佳解；因此，對於Minimization problem(Smaller-the-better):

= -10

Maximization problem(Larger-the-better):

= -10

Where is the objective value of each particle, n is the number of particles

**Section3: Requirements and Functionality Implementation of the System**

* Data structure

Inherited from PSO:

class SCASolver

{

internal Random randomizer = new Random();

//SCA settings

internal int numberOfParticles = 30;

public int IterationLimit { get; set; } = 500;

//with respect to bench mark problem

internal int numberOfVariables;

internal OptimizationType optimizationmode;

internal double[] lowerBounds;

internal double[] upperBounds;

//data fields

internal double iterationBestSolution;

internal double[] soFarTheBestSolution;

internal double soFarTheBestObjectiveValue;

internal double[][] localBestSolution;

internal double[] localBestObjectiveValue;

internal ObjectiveFunction objfunction;

internal double iterationAverage;

internal double iterationBest;

internal int iterationCount = 0;

//for UI

internal Series average = new Series("Iteration Average");

internal Series itrBest = new Series("Iteration Best");

internal Series sofarTheBest = new Series("So Far The Best");

//properties

[Category("Execution")]

public int IterationLimit { get; set; } = 500;

* Algorithm

**Initialize** a set of search agents(solutions)(X)

**Do**

**Evaluate** each of the search agents by the objective function

**Update** the best solution obtained so far(P = X\*)

**Update** 、、、and

**Update** the position of search agents

**While**(t< maximum number of iterations)

**Return** the best solution obtained so far as the global optimum

The following position updating equations are prosed by:

, < 0.5

=

, 0.5

Where

,where t:iteration count T:maximum iteration

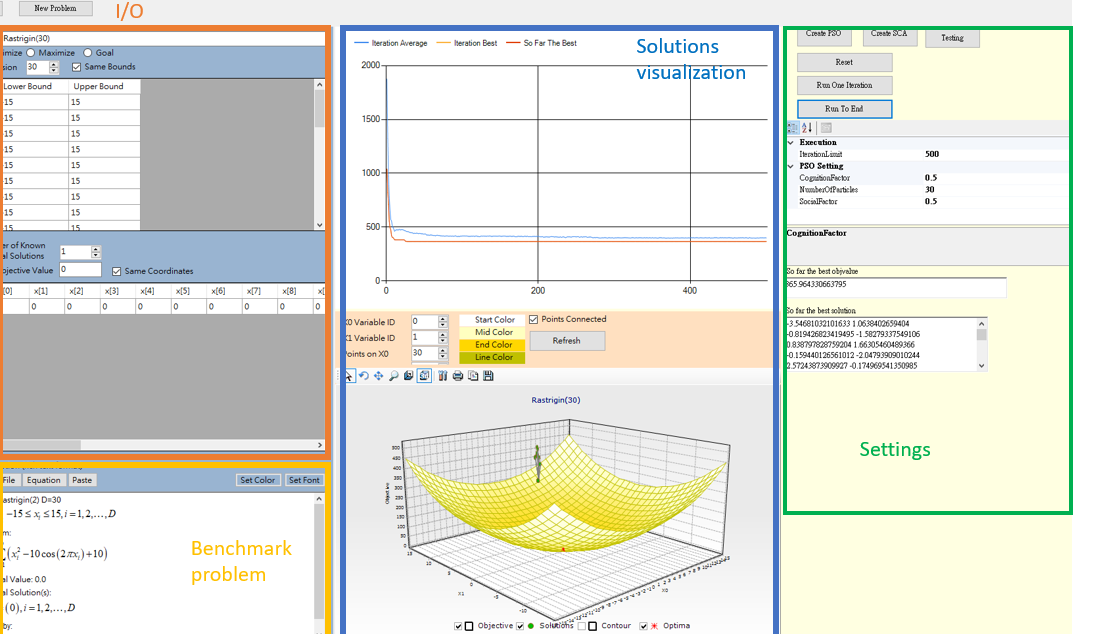
= 2\*

My adaptive way to update r1:

if SN ratio <1 :

else :

* Interface design



* Functionalities

Please look up the source code, which is exactly same as PSO Algorithm.

**Section4: Numerical or Example Tests**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Algorithm**  **Benchmark problems** | | PSO | | SCA | | AdaSCA | |
| mean | std | mean | std | mean | std |
| Ackley(2) |  | 0.216999 | 0.221189 | 0 | 0 | 6.61E-01 | 0.202042 |
| Ackley(10) | | 9.250488 | 0.130398 | 0.019952 | 0.181994 | 8.82E-01 | 0.250153 |
| Ackley(20) | | 9.333706 | 0.135248 | 2.00E-02 | 0.273571 | 9.32E-01 | 0.325761 |
| Ackley(30) | | 9.658840 | 0.137987 | 8.24E-01 | 0.237533 | 5.73E-01 | 0.237168 |
| Bohachevesky(2) | | 9.247171 | 0.136021 | 0 | 0 | 0 | 0 |
| Girewank(2) | | 9.500719 | 0.165093 | 1 | 0 | 0 | 0 |
| Girewank(10) | | 9.117605 | 0.132147 | 1 | 0 | 1 | 1.92E-11 |
| Girewank(20) | | 9.131721 | 0.185763 | 1 | 6.99E-14 | 1 | 7.77E-14 |
| Girewank(30) | | 8.675168 | 0.141451 | 1.09E+00 | 0.081789 | 1.17 | 0.132213 |
| Schwefel(2) | | 262.2758 | 0.199403 | 7.50E-01 | 0.210669 | 1.2 | 0.266159 |
| Schwefel(10) | | 2861.309 | 0.082148 | 2.11E+03 | 0.062197 | 1984.369 | 0.094831 |
| Schwefel(20) | | 6438.257 | 0.037028 | 5.32E+03 | 0.050256 | 5296.562 | 0.038368 |
| Schwefel(30) | | 10142.75 | 0.042652 | 8.79E+03 | 0.030715 | 8809.239 | 0.029506 |

**Section5: Conclusion and Discussion**

經過實驗驗證與觀察，在只有一個全局最佳解區域且無區域最佳解時的標竿問題，表現:SCA> AdaSCA>PSO；當只有一個全局最佳解並有多個區域最佳解時，表現AdaSCA>SCA>PSO。因此，面對多個區域最佳解問題時，把SN-ratio納入解的更新是有用的。另外關於老師針對r1如何調整?根據實驗觀察與標竿問題的困難度有關；在測試時當r1>4就很難收斂了，因為迭代前期r1越大exploitation的時間越久，困難的問題才需要較大的r1以避免快速的收斂到區域最佳解。此外，近一步探討SN-ratio，遇到較容易的問題，考慮原始的SN-ratio公式理論上可以幫助群體往全局最佳方向，更快速收斂到全局最佳解。

Reference:

MIRJALILI. “SCA: A Sine Cosine Algorithm for Solving Optimization Problems.” *Knowledge-based systems* (2016): 120–133. Web.