Hybrid Coevolutionary Glowworm Swarm Optimization Algorithm with Simplex Search Method for System of Nonlinear Equations

Liangdong Qu\textsuperscript{a,b}, Dengxu He\textsuperscript{a,b,*}, Jinzhao Wu\textsuperscript{a,b,c}

\textsuperscript{a}College of Mathematics and Computer Science, Guangxi University for Nationalities, Nanning 530006, China
\textsuperscript{b}Guangxi Key Laboratory of Hybrid Computation and IC Design Analysis, Nanning 530006, China
\textsuperscript{c}School of Computer and Information Technology, Beijing Jiaotong University, Beijing 100044, China

Abstract

In this paper, the problem of solving system of nonlinear equations is equivalently transformed into the problem of function optimization, a hybrid coevolutionary glowworm swarm optimization algorithm with simplex search method for system of nonlinear equations (HCGSOSM) is proposed, where simplex search method is embedded to GSO, it is a local search operator, and combined with GSO to speed up the local search and then the more accurate solution can be obtained. The hybrid optimization algorithm sufficiently exerts the advantages such as group search, strong robustness and it satisfies the parallelism requests. For the problem that system of nonlinear equations has multi-solutions, it can also solve multi-solutions at a time. Several numerical simulation results show that the algorithm offers an effective way to solve system of nonlinear equations, high convergence rate, high accuracy and robustness.

Keywords: Glowworm Swarm Optimization Algorithm; Simplex Search Method; System of Nonlinear Equations; Swarm Intelligence; Parallelism

1 Introduction

Solving system of nonlinear equations is an important question in scientific calculations and engineering technology field. So far, people have made a lot of researches to solve nonlinear equations using theoretical and computational methods. But the problem of solving nonlinear equations is still not completely solved and lacks efficient and reliable algorithms for the strong nonlinear engineering calculation problems especially. Therefore, studying the efficient algorithms...
for strong nonlinear equations is a significant work. At present, some scholars calculate them with Newton’s method, genetic algorithm, evolutionary strategy, particle swarm algorithm and neural network algorithm and so on [1-8]. However, the initial value selection is difficult for most traditional algorithms, however, the solution accuracy need to be improved for intelligent algorithms.

In 2005, Krishnanand and Ghose proposed glowworm swarm optimization (GSO) algorithm [9]. GSO algorithm is a swarm intelligence bionic algorithm and it has good capacity to search for global extremum and more extremums of multimodal optimization problems. The GSO algorithm was applied to multimodal optimization, noise issues, theoretical foundations, addressing the problem of sensing hazards and pursuit of multiple mobile signal sources problems.

**Related Work:** The GSO does not depend on the initial points and derivatives to solve objective function and the problem of solving system of nonlinear equations can be transformed into a function optimization problem. So we can try to use GSO to solve system of nonlinear equations. But due to the dynamic change of decision domains in the process of glowworms moving, the algorithm slows convergence speed and has poor local search ability in the iteration. On the other hand, simplex search method, a kind of traditional algorithm, has strong local search ability, but the global convergence ability is poor, the optimization results depend on the initial selection and hasn’t parallelism. Analyzed the advantages and disadvantages of GSO and simplex search method, a hybrid GSO based simplex search method is proposed, where simplex search method is a local search operator, it is embedded to GSO and speed up the local search of GSO. Simulation results show that the hybrid optimization algorithm for solving system of nonlinear equations improves the convergence speed, convergence rate, high accuracy and robustness. It is a novel effective method for solving system of nonlinear equations.

The paper is organized as follows: Section 2 describes GSO algorithm and its principle. Section 3 presents simplex search method. Then the hybrid coevolutionary GSO algorithm with simplex search method for system of nonlinear equations is described in Section 4. Experiments and discussions are presented in Section 5. Finally, Section 6 provides some conclusions.

## 2 Glowworm Swarm Optimization Algorithm

GSO algorithm researches the behaviors of glowworms in the nature which glow to attract mates or prey. It is a swarm intelligence algorithm, and its basic principle is as follows: Luciferin induces glowworm to glow to attract mates or prey. The brighter the glow more is the attraction, meanwhile the higher of the luciferin, then glowworm moves towards the position having high luciferin. The luciferin value is corresponding to the fitness function value, so glowworm looks for the position having highest luciferin value to determine the optimal value of the fitness function in dynamic decision domains.

A set of \( n \) glowworms are randomly deployed in an \( m \)-dimensional dimensional function space. According to the similarity of luciferin value, each glowworm \( i \) selects a neighbour \( j \) with a probability \( p_{ij} \) and moves toward it within its decision domains range \( r_d \) \((0 < r_d < r_s)\), where \( r_s \) is a circular sensor above range of glowworm \( i \). The position of glowworm \( i \) is \( x_i(x_i \in \mathbb{R}^m, i = 1, 2, \cdots, n) \), which is a potential solution. Put \( x_i \) into the objective function and gain the fitness function value \( J(x_i) \) and luciferin value \( \ell_i \). Estimate the solution with luciferin value.

The algorithm can gain the optimal value of functions. The equations that modeled the
luciferin-update, probability distribution used to select a neighbour, movement update and local-decision range update are given below:

\[
\ell_i(t) = (1 - \rho_1)\ell_i(t - 1) + \gamma J(x_i(t)), \tag{1}
\]

\[
N_i(t) = \{ j : ||x_j(t) - x_i(t)|| < r^d_i(t), \ell_j(t) > \ell_i(t) \}, \tag{2}
\]

\[
p_{ij} = \frac{\ell_j(t) - \ell_i(t)}{\sum_{k \in N_i(t)} \ell_k(t) - \ell_i(t)}, \tag{3}
\]

\[
x_{i}(t + 1) = x_i(t) + s \frac{x_j(t) - x_i(t)}{||x_j(t) - x_i(t)||}, \tag{4}
\]

\[
r^d_i(t + 1) = \min \{ r_s, \max \{ 0, r^d_i(t) + \beta (n_t - |N_i(t)|) \} \}. \tag{5}
\]

Where \(N_i(t)\) is called neighbours of glowworm \(i\) consisting of those glowworms that have a relatively higher luciferin value and that are located within a dynamic decision domain. If the luciferin value of glowworm \(j\) is greater than \(i\)'s and the distance between the glowworm \(i\) and \(j\) is less than the dynamic decision domain, divide glowworm \(j\) into the neighbours of glowworm \(i\).

The \(\beta\) constant parameter affects the rate of change of the neighbourhood range. The constant parameter \(\rho_1\) decides whether algorithm has memory. A value \(\rho_1 = 0\) renders the algorithm memory less where the luciferin value of each glowworm depends only on the fitness value of its current position. Where \(\rho_1 \in (0, 1]\) leads to the reflection of the cumulative goodness of the path followed by the glowworms in their current luciferin values. The constant parameter \(\gamma\) can scale the function fitness values. The value of step-size \(s\) influences the range of objective function.

### 3 Simplex Search Method

Simplex search method or Nelder-Mead method, originally published in 1965, is one of the best known algorithms for multidimensional unconstrained optimization without derivatives [10]. The basic algorithm is quite simple to understand and very easy to use. The method does not require any derivative information, which makes it suitable for problems with non-smooth functions. For these reasons, it is very popular in many fields of science and technology. The general algorithm is given below:

1. **Initial simplex:** The initial simplex \(S\) is usually constructed by generating \(m + 1\) vertices \(x_0, x_1, \ldots, x_m\) around a given input point.

2. **Ordering:** Determine the indices of the worst, second worst and the best vertex, respectively, the vertices of the current working simplex \(S\) are ordered with respect to the function values, to satisfy \(f_0 \leq f_1 \leq \cdots \leq f_m\). Then \(l = 0, s = m - 1, \) and \(h = m\).

3. **Centroid:** Calculate the centroid \(c\) of the best side—this is the one opposite the worst vertex \(x_h\),

\[
c = \frac{1}{m} \sum_{j \neq h} x_j. \tag{6}
\]

4. **Centroid:** Compute the new working simplex from the current one. First, try to replace only the worst vertex \(x_h\) with a better point by using reflection, expansion or contraction with
respect to the best side. All test points lie on the line defined by \( x_h \) and \( c \), and at most two of them are computed in one iteration. If this succeeds, the accepted point becomes the new vertex of the working simplex. If this fails, shrink the simplex towards the best vertex \( x_l \). In this case, \( m \) new vertices are computed.

Simplex transformations in the Nelder-Mead method are controlled by four parameters: \( \alpha \) for reflection, \( \beta_2 \) for contraction, \( \gamma_2 \) for expansion and \( \delta \) for shrinkage. They should satisfy the following constraint \( \alpha > 0, 0 < \beta_2 < 1, \gamma_2 > 1, \gamma_2 > \alpha, 0 < \delta < 1 \). The standard values, used in most implementations, are \( \alpha = 1, \beta_2 = 0.5, \gamma_2 = 2, \delta = 0.5 \).

a. Reflect: Compute the reflection point \( x_r = c + \alpha (c - x_h) \) and \( f_r = f(x_r) \). If \( f_l \leq f_r \leq f_s \), accept \( x_r \) and terminate the iteration.

b. Expand: If \( f_r < f_l \), compute the expansion point \( x_e = c + \gamma_2 (x_r - c) \) and \( f_e = f(x_e) \). If \( f_e < f_r \), accept \( x_e \) and terminate the iteration. Otherwise (if \( f_e \geq f_r \)), accept \( x_r \) and terminate the iteration.

c. Contract: If \( f_r \geq f_s \), compute the contraction point \( x_c \) by using the better of the two points \( x_h \) and \( x_r \).

Outside: If \( f_s \leq f_r \leq f_h \), compute \( x_c = c + \beta_2 (x_r - c) \) and \( f_c = f(x_c) \). If \( f_c \leq f_r \), accept \( x_c \) and terminate the iteration. Otherwise, perform a shrink transformation.

Inside: If \( f_r \geq f_h \), compute \( x_c = c + \beta_2 (x_h - c) \) and \( f_c = f(x_c) \). If \( f_c < f_h \), accept \( x_c \) and terminate the iteration. Otherwise, perform a shrink transformation.

d. Shrink: Compute \( m \) new vertices \( x_j = x_l + \delta (x_j - x_l) \) and \( f_j = f(x_j) \), for \( j = 0, 1, \ldots, m \), with \( j \neq l \).

(5) The stop criteria: Iteration number \( T \) or \( \left\{ \frac{1}{m+1} \sum_{j=0}^{m} [f(x_j) - f(c)]^2 \right\}^{1/2} \).

4 Hybrid Coevolutionary Glowworm Swarm Optimization Algorithm with Simplex Search Method for System of Nonlinear Equations (HCGSOSSM)

4.1 Transformation of the Problem

Suppose that system of nonlinear equations as follows:

\[
\begin{align*}
  f_1(x_1, x_2, \ldots, x_m) &= 0, \\
  f_2(x_1, x_2, \ldots, x_m) &= 0, \\
  \vdots \\
  f_K(x_1, x_2, \ldots, x_m) &= 0.
\end{align*}
\]

(7)

Solving Eq. (7) is equivalent to solve a function optimization problem as follows:
\[
\min P(X) = \sum_{i=1}^{K} f_i^2(x_1, x_2, \cdots, x_m),
\]

Hence solving Eq.(7) is transformed into solving a set of values \( x^* = (x_1, x_2, \cdots, x_m) \), which makes equation \( P(x^*) = 0 \).

### 4.2 The Process of HCGSOSSM

Now our task is to minimize Eq.(8). Because the efficiency and accuracy of simplex search method have a great relationship with the initial point, it hasn’t parallelism and only get a solution one time, while, GSO can overcome its disadvantages, however, the efficiency and accuracy of GSO are low, convergence speed slows. In Literature [11], after implemented GSO, Hooke-Jeeves pattern search is implemented, for the next iteration of GSO, it doesn’t utilize the results of Hooke-Jeeves pattern search, so its efficiency is also low. Simplex search method is embedded to GSO in this paper. In the new algorithm, if a glowworm has neighbours, it moves to its one neighbour with a probability, if it hasn’t neighbours, this shows it is local optimum. Its quality is better and can be used as the initial point of simplex search method. Then, the information of two algorithms are utilized each other. We can define the fitness function as follows:

\[
J(x) = \frac{1}{1 + P(x)}.
\]

The flow chart of HCGSOSSM is the following Fig. 1.

![Flow chart of HCGSOSSM](#)
Its specific implementation steps are illustrated as follows:

**Step 1** Set the initial luciferin $\ell_0$, initial decision domains range $r_0^d$, circular sensor range $r_s$, neighbour number $n_t$, moving step of glowworm $s$ and iteration number of simplex search $T$. Set the stop criteria of HCGSOSSM, for example $\max_iter$ as the maximum iteration.

**Step 2** Randomly initialize a population and the population has $n$ glowworms.

**Step 3** According to Eq.(1), (2) and (9), calculate fitness value and current location of each glowworm $i$, gain a corresponding luciferin value and its neighbours set $N_i(t)$.

**Step 4** if $N_i(t)$ isn’t empty, according to Eq.(3) and (4), calculate $p_{ij}$ and $x_i(t+1)$; otherwise, $x_i(t)$ is used as the initial point of Cauchy iteration method and implement Cauchy iteration method.

**Step 5** According to Eq.(5), update $r_i^d$.

**Step 6** (Judge the stop criteria) If the stop criteria are satisfied, the algorithm ends and outputs the results; otherwise, go to Step 3.

This new algorithm has the following characteristics:

1. Structure complementarity, enriches the structures and enhances parallelism. Simplex search method’s structure is simple, is the serial structure; GSO is the parallel structure. After mix, the structures are rich. Those enhance parallelism.

2. Behavior complementarity, raises the convergence speed. Simplex search method iterates quickly, but it hasn’t population information, easy to fall into the local minimum; the speed of GSO is slow, but it has certain overall situation ability. After the mix, Simplex search method enhance the convergence speed and rate of GSO.

3. Weakens the stringency of parameters. The two algorithms have some parameters and the below experiments show that the dependence on the parameters of the new algorithm is not strong.

4. Weakens the influence of the initial points.

5. Because GSO can solve multimodal optimization problems, for multi-solutions of system of nonlinear equations, HCGSOSSM can get all multi-solutions at a time.

## 5 Experiments

In order to verify the feasibility and validity of the new algorithm, several experiments have been conducted and the experimental conditions are: Intel(R) Core (TM) 2 Duo 2.20GHz CPU, 1G Memory and Windows XP operation system. The programs are realized in MATLAB 7.

In the experiments of HCGSOSSM, the initial simplex $S$ is constructed by normal distribution $N(0, 0.01^2)$ around a given input point. The main parameters of HCGSOSSM are set as follows:

<table>
<thead>
<tr>
<th>$n$</th>
<th>$\rho$</th>
<th>$\gamma$</th>
<th>$\beta$</th>
<th>$n_t$</th>
<th>$s$</th>
<th>$\ell_0$</th>
<th>$r_0^d$</th>
<th>$r_s$</th>
<th>$T$</th>
<th>$\max_iter$</th>
</tr>
</thead>
<tbody>
<tr>
<td>50</td>
<td>0.4</td>
<td>0.6</td>
<td>0.08</td>
<td>5</td>
<td>0.01</td>
<td>5</td>
<td>5</td>
<td>3</td>
<td>5</td>
<td>100</td>
</tr>
</tbody>
</table>
Example 1[4]: \[
\begin{align*}
  f_1(x) &= x_1^2 - x_2 + 1 = 0, \\
  f_2(x) &= x_1 - \cos(0.5\pi x_2) = 0.
\end{align*}
\]
Where \(x \in [-2, 2]\), the exact values are \(x^* = (-1/\sqrt{2}, 1.5), x^* = (0, 1)\) and \(x^* = (-1, 2)\).

Example 2[4,5]: \[
\begin{align*}
  f_1(x) &= x_1^2 - x_2 - 1 = 0, \\
  f_2(x) &= (x_1 - 2)^2 + (x_2 - 0.5)^2 - 1 = 0.
\end{align*}
\]
Where \(x \in [0, 2]\), the exact values are \(x^* = (1.546342883319945, 1.391176312794241)\) and \(x^* = (1.067346085806690, 0.139227668868681)\).

Example 3[6]: \[
\begin{align*}
  f_1(x) &= 3x_1 + x_1^3 + x_2 + 1 = 0, \\
  f_2(x) &= x_1 + 2x_2 + e^{x_3} - 2 = 0.
\end{align*}
\]
\[
\begin{align*}
  f_1(x) &= x_1^3 + x_2^3 + 5x_1x_2x_3 - 85 = 0, \\
  f_2(x) &= x_1^3 - x_2^3 + 60 = 0, \\
  f_3(x) &= x_3^3 - x_2 - 2 = 0, \\
  D &= \{(x_1, x_2, x_3)|3 \leq x_1 \leq 5, 2 \leq x_2 \leq 4, 0.5 \leq x_3 \leq 2\}
\end{align*}
\]
The exact value is \(x^* = (4, 3, 1)\).

Example 4[7]: \[
\begin{align*}
  f_1(x) &= x_1^2 x_2 + x_2^2 - 5x_1x_2x_3 - 85 = 0, \\
  f_2(x) &= x_3^3 - x_2^3 - x_2 - 2 = 0, \\
  f_3(x) &= x_3^3 - x_2 - 2 = 0, \\
  D &= \{(x_1, x_2, x_3)|3 \leq x_1 \leq 5, 2 \leq x_2 \leq 4, 0.5 \leq x_3 \leq 2\}
\end{align*}
\]

The exact value is \(x^* = (4, 3, 1)\).

Table 2: The experimental results for Example 1-4

<table>
<thead>
<tr>
<th>Ex.</th>
<th>Method</th>
<th>Average approximate roots (x)</th>
<th>(P(x))</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Literature[4]</td>
<td>(-0.707724,1.500668)</td>
<td>5.7595e-008</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(0.000114,0.999817)</td>
<td>6.3581e-008</td>
</tr>
<tr>
<td></td>
<td>Literature[11]</td>
<td>(-0.730839333990662,1.542360512171837)</td>
<td>5.3950e-004</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(1.429129869894717e-006 , 1.000113980653129)</td>
<td>4.5561e-008</td>
</tr>
<tr>
<td></td>
<td>HCGSOSSM</td>
<td>(-0.707106781186548,1.5000000000000000)</td>
<td>7.5188e-031</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(0.0000000000000000, 1.0000000000000000)</td>
<td>3.7494e-033</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(-1.0000000000000000, 2.0000000000000000)</td>
<td>0</td>
</tr>
<tr>
<td>2</td>
<td>Literature[4]</td>
<td>(1.546314, 1.391152)</td>
<td>4.5201e-009</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(1.067307, 0.139012)</td>
<td>6.9729e-008</td>
</tr>
<tr>
<td></td>
<td>Literature[11]</td>
<td>(1.546443398564858, 1.390631993359393)</td>
<td>1.8572e-006</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(1.066851999307000, 0.139425973209928)</td>
<td>2.1760e-006</td>
</tr>
<tr>
<td></td>
<td>HCGSOSSM</td>
<td>(1.546342883319945, 1.391176312794241)</td>
<td>3.0815e-031</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(1.067346085806690, 0.13922766886861)</td>
<td>1.4298e-030</td>
</tr>
<tr>
<td>3</td>
<td>Literature[6]</td>
<td>(-0.451123, 0.445178)</td>
<td>9.8118e-013</td>
</tr>
<tr>
<td></td>
<td>Literature[11]</td>
<td>(-0.631192941509398, 0.494703417321859)</td>
<td>0.4230</td>
</tr>
<tr>
<td></td>
<td>HCGSOSSM</td>
<td>(-0.451122938776610, 0.445177705391158)</td>
<td>5.2385e-030</td>
</tr>
<tr>
<td>4</td>
<td>Literature[7]</td>
<td>(3.9940, 3.0079, 1.0079)</td>
<td>0.1589</td>
</tr>
<tr>
<td></td>
<td>Literature[11]</td>
<td>(4.001406721890630, 3.011350221428261, 0.999234835117343)</td>
<td>5.0315</td>
</tr>
<tr>
<td></td>
<td>HCGSOSSM</td>
<td>(4.0000000000000000, 3.0000000000000000, 1.0000000000000000)</td>
<td>0</td>
</tr>
</tbody>
</table>
From Table 2, we see that HCGSOSSM can obtain the approximate roots of the systems of equations. Compared with the results of the references, the approximate roots of Example 1-4 have small errors. Especially, for Example 1, HCGSOSSM obtained the approximate root of the other root (-1, 2).

Example 5[8]:
\[
\begin{align*}
x^2 + y^2 + z^2 - 3 &= 0, \\
x^2 + y^2 + xy + x + y - 5 &= 0, \\
x + y + z - 3 &= 0,
\end{align*}
\]
\(x, y, z \in [-1.732, 1.732]\).

The theoretical value \(X^*\) is \(x = y = z = 1\).

Example 6[8]:
\[
\begin{align*}
x + y - 2z &= 0, \\
xy - 1 &= 0, \\
x^2 + y^2 - 2 &= 0,
\end{align*}
\]
\(x, y, z \in [-2, 2]\).

The theoretical value \(X^*\) are \(x = y = z = 1\) and \(x = y = z = -1\).

In the experiments, average approximate roots of HCGSOSSM is denoted by \(HCGSOSSMR\), average errors of HCGSOSSM is denoted by \(HCGSOSSME\), average errors of AGSO in Lecture[11] is denoted by \(AGSOE\), average errors of PSO-LM[8] is denoted by \(PSO-LME\), average errors of basic PSO[8] is denoted by \(PSOE\).

<table>
<thead>
<tr>
<th>Ex.</th>
<th>Variable</th>
<th>(HCGSOSSMR)</th>
<th>(HCGSOSSME)</th>
<th>(AGSOE)</th>
<th>(PSO-LME)</th>
<th>(PSOE)</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>(x)</td>
<td>1.0000000027489102</td>
<td>2.7489e-008</td>
<td>0.0174</td>
<td>0.0773</td>
<td>0.0877</td>
</tr>
<tr>
<td></td>
<td>(y)</td>
<td>0.9999999972510898</td>
<td>2.7489e-008</td>
<td>0.0148</td>
<td>0.0773</td>
<td>0.0877</td>
</tr>
<tr>
<td></td>
<td>(z)</td>
<td>1.0000000000000000</td>
<td>3.3307e-016</td>
<td>0.0047</td>
<td>0.0773</td>
<td>0.0877</td>
</tr>
<tr>
<td>6</td>
<td>(x)</td>
<td>-1.000000006490788</td>
<td>6.4907e-009</td>
<td>—</td>
<td>—</td>
<td>—</td>
</tr>
<tr>
<td></td>
<td>(y)</td>
<td>-0.999999993509212</td>
<td>6.4907e-009</td>
<td>—</td>
<td>—</td>
<td>—</td>
</tr>
<tr>
<td></td>
<td>(z)</td>
<td>-1.0000000000000000</td>
<td>0</td>
<td>—</td>
<td>—</td>
<td>—</td>
</tr>
</tbody>
</table>

From Table 3, it is observed that HCGSOSSM can obtain the approximate roots of systems of equations and have smaller errors compared with the results in references. Especially, for Example 6, it also solved the approximate roots of the other root (-1, -1, -1). The experimental results show that HCGSOSSM is effective.

### 6 Conclusions

In this paper, we propose a hybrid coevolutionary glowworm swarm optimization algorithm with simplex search method for system of nonlinear equations. The hybrid algorithm does not need
to restrict the forms of equations; also equations do not need to be continuous and differentiable. The hybrid algorithm is able to overcome the problem that it’s difficult for traditional algorithms to select initial values and has parallelism. For the problem that system of nonlinear equations has multi-solutions, it can also solve multi-solutions at a time. The experimental results show that the algorithm is effective, and it has high accuracy, high convergence rate and parallelism.

Acknowledgement

This work is supported by the National Natural Science Foundation of China(No.60873118 and 60973147), the Doctoral Fund of Ministry of Education of China(20090009110006), the Key Natural Science Foundation of Guangxi University for Nationalities (No.2010ZD008) and the Natural Science Foundation of Guangxi of China (No.2010GXNSFB013052).

References