Finding all solutions of systems of nonlinear equations using spiral dynamics inspired optimization with clustering

Article *in* Journal of Advanced Computational Intelligence and Intelligent Informatics · September 2015

CITATIONS
8

READS
650

2 authors:

Kuntjoro Adji Sidarto
Bandung Institute of Technology
48 PUBLICATIONS **198** CITATIONS

SEE PROFILE

Adhe Kania
Bandung Institute of Technology
5 PUBLICATIONS **11** CITATIONS

SEE PROFILE

Paper:

Finding All Solutions of Systems of Nonlinear Equations Using Spiral Dynamics Inspired Optimization with Clustering

Kuntjoro Adji Sidarto and Adhe Kania

Department of Mathematics, Institut Teknologi Bandung
Bandung 40132, Indonesia
E-mail: {sidarto,adhe.kania}@math.itb.ac.id
[Received April 29, 2015; accepted August 6, 2015]

Nowadays the root finding problem for nonlinear system equations is still one of the difficult problems in computational sciences. Many attempts using deterministic and meta-heuristic methods have been done with their advantages and disadvantages, but many of them have fail to converge to all possible roots. In this paper, a novel method of locating and finding all of the real roots from the system of nonlinear equations is proposed mainly using the spiral dynamics inspired optimization by Tamura and Yasuda [1]. The method is improved by the usage of the Sobol sequence of points for generating initial candidates of roots which are uniformly distributed than of pseudo-random generated points. Using clustering technique, the method localizes all potential roots so the optimization is conducted in those points simultaneously. A set of problems as the benchmarks from the literature is given. Having only a single run for each problem, the proposed method has successfully found all possible roots within a bounded domain.

Keywords: systems of nonlinear equations, root finding problem, spiral dynamics inspired optimization, clustering, Sobol sequence of points

1. Introduction

The problem of solving systems of nonlinear equations in several variables arises frequently in a wide range of engineering and practical applications. The Newton and quasi-Newton methods, which are widely used, have advantage on their high speed of convergence once the initial candidate of the root which is sufficiently accurate is provided. The weakness of these methods is that accurate initial candidate must be given beforehand to ensure convergence. To overcome this weakness, several meta-heuristic optimization algorithms are recently proposed. These algorithms firstly convert the problem of finding the roots of system of nonlinear equations into an optimization problem, then they look for the desired root as a point that solve the optimization problem. Luo et al. [2] proposed a hybrid approach using chaos optimization algorithm and quasi-Newton method. Burden and Faires [3] made the combination of steepest descent method and Newton-type methods for solving systems of nonlinear equations. For each approach, chaos optimization algorithm and steepest descent method are used to obtain an initial good candidate of the root that will be used by Newton-type method to obtain the solution. However, at single run of the algorithm these approaches are only able to find a single root of a system of nonlinear equations. If we want to find all possible roots, the programs should run repeatedly for an adequate time.

The problem of finding all roots based on meta-heuristic optimizations have been proposed on some recent articles. Generally the first step in the methods in those articles is to transform the problem into an optimization problem. In Tsoulos and Stavrakoudis [4], the authors implement the global optimization methods, such as Multistart and Minfinder, in order to discover all local minima of the optimization problem. Sacco and Henderson [5] who were searching multiple roots, used the algorithm of Luus-Jakola random search to explore the search space and used Fuzzy Clustering Means for clustering the best solutions previously found, for finding more than one root. Subsequently, the searching using the best solution as the starting points was done based on the algorithm of multiple Nelder-Mead simplex instances. Grosan and Abraham [6] transformed the system of nonlinear equations into a multi-objective optimization problem and solved it by employing an evolutionary computation technique. Recently Song et al. [7] proposed a transformation technique from the system of nonlinear equations into a bi-objective optimization problem, and solve it by employing an evolutionary computation technique.

Having transforming the nonlinear equation system into an optimization problem, the approach in this paper focuses on locating all possible real roots of the nonlinear equation system using a combination of a certain clustering technique and the algorithm of Spiral Dynamics Inspired Optimization of Tamura and Yasuda [1]. The algorithm is implemented on each resulting clusters, which are the neighborhoods of the roots in order to find the accurate position of the roots. In the next section, the problem of solving system of nonlinear equations is formulated as an optimization problem. Section 3 presents a review of Spiral Dynamics Optimization Algorithm, and Section 4 presents the proposed clustering technique for locating the

roots of system of nonlinear equations. In Section 5, a method for generating initial population of points based on Sobol sequence is discussed. In Section 6, a problems set which can be the benchmark of the particular method is given, and their solutions resulting from the implementation of the proposed method is presented. Finally some conclusions are given in Section 7.

2. Problem Description

Consider a system of nonlinear equations of the form

$$\begin{aligned} f_1(x_1,x_2,\ldots,x_n) &= 0,\\ f_2(x_1,x_2,\ldots,x_n) &= 0,\\ \vdots \qquad & \quad \vdots \qquad \cdots \cdots \cdots \cdots \\ f_n(x_1,x_2,\ldots,x_n) &= 0, \end{aligned} \tag{1}$$

where $(x_1,x_2,\ldots,x_n) \in D = [a_1,b_1] \times [a_2,b_2] \times \cdots \times [a_n,b_n] \subset \mathbb{R}^n$ and $f_i : D \to \mathbb{R} \quad i = 1,2,\ldots,n$ being continuous functions with at least one of them is nonlinear.

The above system can be written in vector form as follows:

$$\mathbf{f}(\mathbf{x}) = 0$$

where $\mathbf{f} = (f_1,f_2,\ldots,f_n)^T$ and $\mathbf{x} = (x_1,x_2,\ldots,x_n)^T$. A vector $\mathbf{x}^* = (x_1^*,x_2^*,\ldots,x_n^*)^T \in D$ is called a solution or a root of the system if $\mathbf{f}(\mathbf{x}^*) = 0$.

The relationship between the optimization problem of a function from $\mathbb{R}^n$ to $\mathbb{R}$ and the problem of solving system of nonlinear equations is as follows.

System $\mathbf{f}(\mathbf{x}) = 0$ has a solution $\mathbf{x} = (x_1,x_2,\ldots,x_n)^T$ if the objective function F of the optimization problem, which is defined by

$$\begin{aligned} F(\mathbf{x}) &= F(x_1,x_2,\ldots,x_n)\\ &= \frac{1}{1+\sum_{i=1}^{n} |f_i(x_1,x_2,\ldots,x_n)|}\\ &= \frac{1}{1+\sum_{i=1}^{n} |f_i(\mathbf{x})|}, \qquad \cdots \cdots \cdots \end{aligned} \tag{2}$$

has the maximum value of 1. This would suggest the possible usage of global optimization methods for discovering the solution of nonlinear system Eq. (1). If there exist $\mathbf{x}^*$ such that $F(\mathbf{x}^*) = 1$, then $\mathbf{x}^*$ is a global optimum of $F(\mathbf{x})$ in Eq. (2) and $f_i(\mathbf{x}^*) = 0 \quad i = 1,2,\ldots,n$. Hence $\mathbf{x}^*$ is the root of the system of equations. The determination of formula $F(\mathbf{x})$ (Eq. (2)) is motivated by the formula for fitness function $F(x) = \dfrac{1}{1+\mathrm{abs}(f(x))}$ in Aggarwal [8] for solving transcendental equations $f(x) = 0$ using the Genetic Algorithm.

Many optimization problems are of highly nonlinear involving many variables under some constraints. Such nonlinearity often results in the multimodal objective function. Hence, local search algorithm such as hill-climbing or steepest descent methods are not suitable. Only global search algorithms are suitable for obtaining optimal solutions. Recently several metaheuristic methods have been developed to perform global search. These methods does not need the demanding condition such as the differentiability of the objective functions, so they could be used on both continuous and discrete problems. One of the recently proposed methods is the Spiral Dynamics Inspired Optimization developed by Tamura and Yasuda [1]. This optimization method is applied to find the global maximum of $F(\mathbf{x})$ (Eq. (2)) in this paper. Having possibility of several different global maximum points of $F(\mathbf{x})$, which corresponds to several different roots of the system $\mathbf{f}(\mathbf{x}) = 0$, a certain clustering technique is proposed in order to be able to locate the different positions of the roots. In Sections 3 and 4, the Spiral Dynamics Optimization of Tamura and Yasuda and the proposed clustering technique are respectively presented.

3. Spiral Dynamics Optimization

At first, the 2-Dimensional spiral model and n-Dimensional spiral model are reviewed. Let $R^{(2)} = \begin{pmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{pmatrix}$ and $\mathbf{x}$ be any vector in $\mathbb{R}^2$. The multiplication of $\mathbf{x}$ by $R^{(2)}$ gives a counterclockwise rotation of $\mathbf{x}$. The matrix $R^{(2)}$ represents the rotation in the 2D-plane by an angle $\theta \quad (0 < \theta < 2\pi)$. The multiplication of $\mathbf{x}$ by the diagonal matrix $\mathrm{diag}_2(r,r) = \begin{pmatrix} r & 0 \\ 0 & r \end{pmatrix}$ with $0 < r < 1$ gives shortening length of $\mathbf{x}$ in $\mathbb{R}^2$ by the factor of r. Consequently, the multiplication of a vector $\mathbf{x}$ in $\mathbb{R}^2$ by matrix $S_2(r,\theta) = \mathrm{diag}_2(r,r)$. $R^{(2)}$ gives the geometric effect of anticlockwise rotation of $\mathbf{x}$ through the angle θ, then shorten the vector $R^{(2)}\mathbf{x}$ by a factor of r. Consequently, the equation

$$\begin{aligned} &\mathbf{x}_{k+1} = S_2(r,\theta)\,\mathbf{x}_k,\\ &\quad \text{where } \mathbf{x}_k = (x_1(k),x_2(k))^T \quad k = 0,1,2,\ldots \quad . \end{aligned} \tag{3}$$

describes the transformation of an initial point $\mathbf{x}_0$ in $\mathbb{R}^2$ repeatedly by the operator $\mathbf{x} \mapsto S_2(r,\theta)\mathbf{x}$. The graph of the sequence $\mathbf{x}_0,\mathbf{x}_1,\mathbf{x}_2,\ldots$ in the 2D-plane will produce a trajectory of spiral form, so it seems the point $\mathbf{x}_0$ goes spiral inward and toward the origin, in the case of $0 < r < 1$. Having translated the initial point toward an arbitrary point $\mathbf{x}^*$ in Eq. (3) gives

$$\mathbf{x}_{k+1} - \mathbf{x}^* = S_2(r,\theta)(\mathbf{x}_k - \mathbf{x}^*) \quad \cdots \cdots \cdots \tag{4a}$$

where $\mathbf{x}_k = (x_1(k),x_2(k))^T \quad k = 0,1,2,\ldots$, so

$$\mathbf{x}_{k+1} = S_2(r,\theta)\,\mathbf{x}_k - (S_2(r,\theta) - I_2)\,\mathbf{x}^* \quad \cdots \tag{4b}$$

The iterated point $\mathbf{u}_k = \mathbf{u}(k) = \mathbf{x}_k - \mathbf{x}^* \quad k = 0,1,2,\ldots$ in Eq. (4a) will move spirally inward and toward the origin. As a result, the iterated point $\mathbf{x}_0$ will move spirally inward and toward $\mathbf{x}^*$ in the plane.

Consider the optimization problem as follows,

$$\underset{\mathbf{x}\in\mathbb{R}^2}{\text{maximize}}\, F(\mathbf{x}),\ \mathbf{x}=(x_1,x_2)^T\in\mathbb{R}^2.$$

Based on the previous spiral model, the Spiral Dynamics Optimization Algorithm (SDOA) of Tamura and Yasuda [1] is developed below.

Input :

$m(\geq 2)$ the number of search points

$\theta\,(0<\theta<2\pi)$, $r\,(0<r<1)$

$k_{\max}$ maximum number of iteration

Process :

Step 1. Randomly generate initial points $\mathbf{x}_i(0)\in\mathbb{R}^2$ $i=1,2,\ldots,m$ in the feasible region.

Step 2. Set $k=0$.

Step 3. Find $\mathbf{x}^*$ as $\mathbf{x}^*=\mathbf{x}_{i_g}(0)$ with $i_g=\arg\max_i F(\mathbf{x}_i(0))\quad i=1,2,\ldots,m$

Step 4. Update $\mathbf{x}_i$:
$\mathbf{x}_i(k+1)=S_2(r,\theta)\,\mathbf{x}_i(k)-(S_2(r,\theta)-I_2)\,\mathbf{x}^*$
$i=1,2,\ldots,m$

Step 5. Update $\mathbf{x}^*$:
$\mathbf{x}^*=\mathbf{x}_{i_g}(k+1),\quad i_g=\arg\max_i F(\mathbf{x}_i(k+1)),$
$i=1,2,\ldots,m$

Step 6. If $k=k_{\max}$ then stop.
Otherwise, set $k=k+1$ and return to step 4.

Output :

$\mathbf{x}^*$ as a maximum point of $F(\mathbf{x})$.

The extension of 2-D spiral model to n-Dimensional spiral model may be done with the help of rotation matrix in n-dimensional space. In this case, rotation in n-dimension is performed in the similar way as in 2-dimensional rotation. Let $R_{i,j}^{(n)}$ be $n\times n$ matrix whose entries are follows.

$$r_{ii}=r_{jj}=\cos\theta,\ r_{ji}=\sin\theta,\ r_{ij}=-\sin\theta$$
$$\text{and } r_{st}=\delta_{st} \text{ for all other entries of } R_{i,j}^{(n)}.$$
$$(\text{where } \delta_{st}=1 \text{ if } s=t \text{ and } \delta_{st}=0 \text{ if } s\neq t)$$

Here $R_{i,j}^{(n)}$ resemble the identity matrix except for the ii, ij, jj and ji positions. Multiplication of a vector by rotation matrix $R_{i,j}^{(n)}$ only alter the i^{th} and j^{th} components of the vector. It has no effect on other components. Thus $R_{i,j}^{(n)}$ acts as the plane rotation. In total we will have $\binom{n}{2}=\frac{n!}{2!(n-2)!}=\frac{1}{2}n(n-1)$ number of plane rotation matrices. Let $R^{(n)}$ be $n\times n$ matrix defined as follows:

$$R^{(n)}=\prod_{i=1}^{n-1}\left(\prod_{j=1}^{i}R_{n-i,n+1-j}^{(n)}\right)$$

$R^{(n)}$ is a composition of plane rotations matrix $R_{i,j}^{(n)}$. So the n-Dimensional spiral model is formulated as below.

$$\mathbf{x}(k+1)=S_n(r,\theta)\,\mathbf{x}(k)$$
$$\text{where } \mathbf{x}(k)=(x_1(k),x_2(k),\ldots,x_n(k))^T$$
$$\text{and } S_n(r,\theta)=\text{diag}_n(r,r,\ldots,r)\,.\,R^{(n)}$$

Having translated the initial point toward an arbitrary point $\mathbf{x}^*$, the n-Dimensional spiral model with center at $\mathbf{x}^*$ is defined as follows.

$$\mathbf{x}(k+1)=S_n(r,\theta)\,\mathbf{x}(k)-(S_n(r,\theta)-I_n)\,\mathbf{x}^*$$
$$\text{where } \mathbf{x}(k)=(x_1(k),x_2(k),\ldots,x_n(k))^T$$
$$\text{and } S_n(r,\theta)=\text{diag}_n(r,r,\ldots,r)\,.\,R^{(n)}$$

Now consider the optimization problem

$$\underset{\mathbf{x}\in\mathbb{R}^n}{\text{maximize}}\, F(\mathbf{x}),\ \mathbf{x}=(x_1,x_2,\ldots,x_n)^T\in\mathbb{R}^n.$$

The SDOA method for n-dimensional spiral model is written below.

Input :

$m(\geq 2)$ the number of search points

$\theta\,(0<\theta<2\pi),\quad r\,(0<r<1)$

$k_{\max}$ maximum number of iteration

Process :

Step 1. Randomly generate initial points $\mathbf{x}_i(0)\in\mathbb{R}^n$ $i=1,2,\ldots,m$, in the feasible region.

Step 2. Set $k=0$.

Step 3. Find $\mathbf{x}^*$ as $\mathbf{x}^*=\mathbf{x}_{i_g}(0)$ with $i_g=\arg\max_i F(\mathbf{x}_i(0))\quad i=1,2,\ldots,m$

Step 4. Update $\mathbf{x}_i$:
$\mathbf{x}_i(k+1)=S_n(r,\theta)\,\mathbf{x}_i(k)-(S_n(r,\theta)-I_n)\,\mathbf{x}^*$
$i=1,2,\ldots,m$

Step 5. Update $\mathbf{x}^*$:
$\mathbf{x}^*=\mathbf{x}_{i_g}(k+1),\quad i_g=\arg\max_i F(\mathbf{x}_i(k+1)),$
$i=1,2,\ldots,m$

Step 6. If $k=k_{\max}$ then stop.
Otherwise, set $k=k+1$ and return to step 4.

Output :

$\mathbf{x}^*$ as a maximum point of $F(\mathbf{x})$.

4. A Clustering Technique for Locating the Roots of System of Nonlinear Equations

Up to this time, we have optimization problem with the objective function $F(\mathbf{x})=\frac{1}{1+\sum_{i=1}^{n}|f_i(\mathbf{x})|}$ and the SDOA in Section 3 for solving the problem. In general, the nonlinear system (1) may have more than one root in the given

domain, which means there exits more than one of global maximum point of $F(\mathbf{x})$. A single run of SDOA can only obtain a single maximum point. Multiple runs of SDOA must be done in order to obtain the all distinct roots. We propose the idea of finding all possible roots in a single run of algorithm. A certain clustering technique is needed so there are a number clusters potentially containing the roots. Simultaneously, the SDOA will be applied to each cluster. Note that a cluster with center at $\mathbf{x}$ and radius ρ is the set of all points $\mathbf{y}$ satisfying $\|\mathbf{x}-\mathbf{y}\| < \rho$.

The clustering start by evaluating the function $F(\mathbf{x})$ in all initial points obtained by step 1 of SDOA. The point $\mathbf{x}$, which is in the vicinity to the root of $\mathbf{f}(\mathbf{x}) = 0$, will have the value of $F(\mathbf{x})$ close to 1. Hence if $\mathbf{x}_i$ and $\mathbf{x}_j$ are two distinct points where their values of $F(\mathbf{x})$ greater than a prescribed number $\gamma > 0$, with $0 < \gamma < 1$, then these two points will be taken as a consideration. Now let $\mathbf{x}_t = (\mathbf{x}_i + \mathbf{x}_j)/2$ be the mid-point between $\mathbf{x}_i$ and $\mathbf{x}_j$. If $F(\mathbf{x}_t) < F(\mathbf{x}_i)$ and $F(\mathbf{x}_t) < F(\mathbf{x}_j)$, then $\mathbf{x}_i$ and $\mathbf{x}_j$ will be considered as the potential candidates of the roots. Then we construct two separate clusters with $\mathbf{x}_i$ and $\mathbf{x}_j$ respectively as the center. If $F(\mathbf{x}_i) < F(\mathbf{x}_t) < F(\mathbf{x}_j)$ then only the point $\mathbf{x}_j$ will be considered as the potential candidate of the root, so we construct a cluster with center at $\mathbf{x}_j$. On the other hand if $F(\mathbf{x}_t) > F(\mathbf{x}_i)$ and $F(\mathbf{x}_t) > F(\mathbf{x}_j)$, then not only $\mathbf{x}_i$ and $\mathbf{x}_j$ are considered as the potential candidates of the roots, but point $\mathbf{x}_t$ is also considered as the potential candidate of the root. Hence after assigning two separate clusters with each center at $\mathbf{x}_i$ and $\mathbf{x}_j$, this clustering technique will be conducted again for the mid-point of $\mathbf{x}_i$ and $\mathbf{x}_t$. More detail of the preceding description is written in the following steps.

Input

$m_{\text{cluster}} (> 1)$:
the number of search points at the clustering phase

$\gamma (0 < \gamma < 1)$:
'cut-off' parameter for function value $F(\mathbf{x})$

$\varepsilon\,(0 < \varepsilon < 1)$:
parameter for roots acceptance

$\delta\,(0 < \delta < 1)$:
parameter to distinguish between one candidate root and another one in case they are very close each other

$k_{\text{cluster}} (> 1)$:
maximum iteration number at the clustering phase

m, r, θ, $k_{\max}$:
input parameters for SDOA phase

Process

Clustering phase

1. Randomly generate initial points $\mathbf{x}_i(0) \in \mathbb{R}^n \;\; i = 1,2,\ldots,m_{\text{cluster}}$ in the feasible region D, where $D = [a_1,b_1] \times [a_2,b_2] \times \cdots \times [a_n,b_n] \subset \mathbb{R}^n$.
2. Set $k = 0$.
3. Set $\mathbf{x}'$ as $\mathbf{x}' = \mathbf{x}_{ig}(0)$, $i_g = \arg\max_i F(\mathbf{x}_i(0))$ $i = 1,2,\ldots,m_{\text{cluster}}$
4. Store $\mathbf{x}'$ as the center of the first cluster with its radius equals to $\frac{1}{2}(\min_l |b_l - a_l|)$ $l = 1,2,\ldots,n$.
5. For $i = 1,2,\ldots,m_{\text{cluster}}$ do
   If $F(\mathbf{x}_i) > \gamma$ and $\mathbf{x}_i$ is not the center of the existing cluster, then $\mathbf{x}_i$ may have a possibility to become a cluster center, and then do the following functions cluster: *Function Cluster* (input: $\mathbf{y}$)
   a. Find a cluster with its center nearest to $\mathbf{y}$.
   b. Let C be that cluster, with center at $\mathbf{x}_C$.
   c. Set $\mathbf{x}_t$ as mid-point between $\mathbf{y}$ and $\mathbf{x}_C$.
   d. Evaluate $F(\mathbf{y})$, $F(\mathbf{x}_C)$, and $F(\mathbf{x}_t)$:
      - If $F(\mathbf{x}_t) < F(\mathbf{y})$ and $F(\mathbf{x}_t) < F(\mathbf{x}_C)$
        Set a new cluster with center at $\mathbf{y}$ and its radius equals the distance between $\mathbf{y}$ and $\mathbf{x}_t$
      - Else, if $F(\mathbf{x}_t) > F(\mathbf{y})$ and $F(\mathbf{x}_t) > F(\mathbf{x}_C)$
        Set a new cluster with $\mathbf{y}$ as its center and radius equals the distance between $\mathbf{y}$ and $\mathbf{x}_t$. Redo *Function Cluster* with $\mathbf{x}_t$ as its input.
      - Else, if $F(\mathbf{y}) > F(\mathbf{x}_C)$, set $\mathbf{y}$ as the center of C.
   e. Change the radius of C to the distance between $\mathbf{y}$ and $\mathbf{x}_t$.
6. Set $\mathbf{x}_p = \mathbf{x}_{i_g}$ where $i_g = \arg\max_i F(\mathbf{x}_i(k))$ $i = 1,2,\ldots,m_{\text{cluster}}$
7. Update $\mathbf{x}_i$
   $\mathbf{x}_i(k+1) = S_n(r,\theta)\,\mathbf{x}_i(k) - (S_n(r,\theta) - I_n)\,\mathbf{x}_p$
   $i = 1,2,\ldots,m_{\text{cluster}}$
8. Do steps 5 to 7 for k_{cluster} times.

Spiral Optimization phase

9. Having done steps 1 to 8, we obtain a number of clusters: $C_1, C_2, \ldots, C_{n_c}$. Each cluster has center at $\mathbf{x}_{C_i}$ and radius of $\rho_i\,(i = 1,2,\ldots,n_c)$. To each cluster, perform SDOA as described in Section 3, in order to obtain a candidate of root in each cluster. Use m, r, θ, $k_{\max}$ as SDOA input with $D_i = [x_{1,i} - \rho_i, x_{1,i} + \rho_i] \times [x_{2,i} - \rho_i, x_{2,i} + \rho_i] \times \cdots \times [x_{n,i} - \rho_i, x_{n,i} + \rho_i] \subset \mathbb{R}^n$ as domain of the search region where $\mathbf{x}_{C_i} = (x_{1,i}, x_{2,i}, \ldots, x_{n,i})^T$ $i = 1,2,\ldots,n_c$.

Roots selection

10. Keep only candidate roots which satisfy condition $1 - F(\mathbf{x}) < \varepsilon$.
11. Suppose from step 10 there result n_g candidate roots. From these n_g candidate roots, select only those which satisfy $\|\mathbf{x}_i - \mathbf{x}_j\| > \delta$ for $i, j = 1,2,\ldots,n_g$ where $\|\mathbf{x}_i - \mathbf{x}_j\|$ is the distance

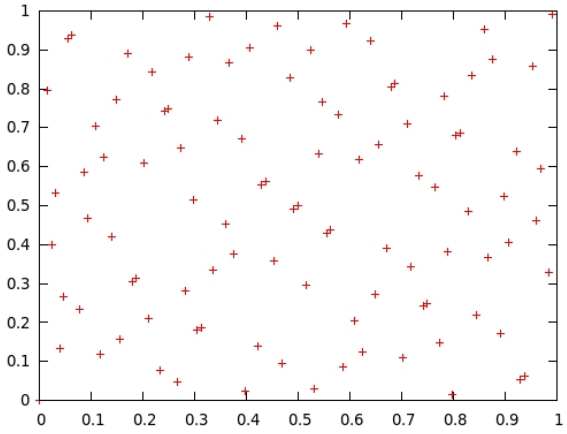

Fig. 1. Scatter plot of the first 100 points of Sobol sequence of points.

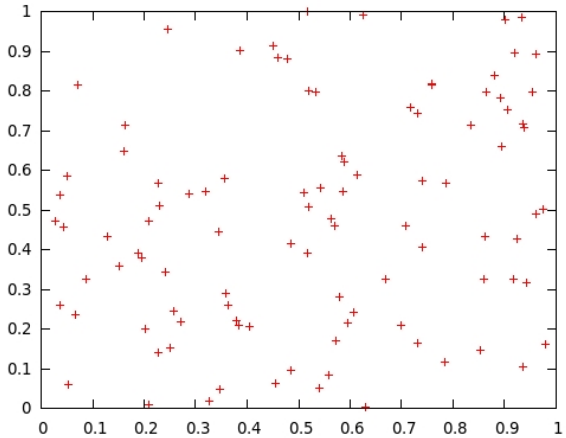

Fig. 2. A scatter plot of 100 points of pseudo-random points.

between the candidate roots $\mathbf{x}_i$ and $\mathbf{x}_j$. In case where $\left\|\mathbf{x}_i - \mathbf{x}_j\right\| \leq \delta$, select only $\mathbf{x}_i$ as a root if $F(\mathbf{x}_i) \geq F(\mathbf{x}_j)$, otherwise select $\mathbf{x}_j$ as a root.

Output

Roots $\mathbf{x}$ of system of nonlinear equations $\mathbf{f}(\mathbf{x}) = 0$.

5. Generating Initial Population of Points Using Sobol Sequence

In step 1 of SDOA and Clustering phase in the previous section, an initial population of points is generated randomly. One problem with random numbers is that the obtaining numbers may not distribute uniformly. As the consequence, the generated initial population of points may not uniformly distribute in the feasible region of the problem. The uniform distribution of initial population of points is much desired in order to obtain all candidate solutions in the clustering phase. It will be helpful if the generated population of points in the search region has minimum the deviation from uniformity. This deviation called discrepancy is measured as follow (see e.g. Seydel [9]).

Let $Q \subseteq [0,1]^n$ be an arbitrary axially parallel n-dimensional rectangle in the unit cube $[0,1]^n \subset \mathbb{R}^n$. It means Q is a product of n intervals. Consider a set of points $\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N \in [0,1]^n$. The idea of discrepancy is that, for evenly distributed point set, the fraction of the points lying within the rectangle Q should correspond to the volume of the rectangle. Let # denote the number of points, then we expect that

$$\frac{\#\text{ of } \mathbf{x}_i \in Q}{\#\text{ of all points}} \approx \frac{vol(Q)}{vol([0,1]^n)}$$

for as many rectangles as possible. This leads to the following definition of discrepancy of point set:

The discrepancy of the point set $\{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N\}$ is $D_N = \sup_Q \left| \frac{\#\text{ of } \mathbf{x}_i \in Q}{N} - vol(Q) \right|$.

If the set of rectangle is restricted to Q^* with $Q^* = \prod_{i=1}^n [0, y_i)$, the corresponding discrepancy is denoted by D_n^*. The more evenly the points are distributed, the closer discrepancy D_N is to zero. In this case D_N refers to the first N points of a sequence of points $\mathbf{x}_1, \mathbf{x}_2, \ldots$. The discrepancies D_N and D_N^* satisfy $D_N^* \leq D_N \leq 2^n D_N^*$.

Next, a sequence of points $\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N, \cdots \in \mathbb{R}^n$ is called low-discrepancy sequence if there is a constant C_n such that for all N we have $D_N \leq C_n (\ln N)^n / N$. The sequences with low discrepancy are also called quasi-random numbers. Various low discrepancy sequences have been constructed and Sobol sequence is one of them. The result of constructing Sobol sequence of 100 points in two dimensions, using algorithm developed by Joe and Kuo [10], is shown in **Fig. 1**. **Fig. 2** shows the result of constructing pseudo-random sequence of 100 points in two dimensions. Visual inspection of **Figs. 1** and **2** show that pseudo-random sequence of points could give some regions with no points at all, whereas Sobol sequence of points appears to cover the region more uniformly.

Considering the advantage of having low discrepancy sequence of points rather than the pseudo-random sequence of points, we propose the usage of Sobol sequence of points as initial points $\mathbf{x}_i(0) \in \mathbb{R}^n$ both in the Step 1 of SDOA and of Clustering phase.

Since the construction of Sobol sequence of points does not involve generation of pseudo-random points, the proposed algorithm does not involve random process. Thus for a given set of parameter values, the proposed algorithm will always produce the same results each time we execute the algorithm.

6. Numerical Experiments

In order to assess the efficacy of the proposed technique, a set of problems from various benchmark problems have been examined. In this study, all the numerical experiments were performed on a Notebook equipped with processor Intel Core™i5 with 4 GB ram and 1.6 GHz CPU running Ubuntu Linux 12.04. The code

was written in C++ and compiled using g++.

6.1. Test Problems

Problem 1:

From Chen et al. [11], the system of equations and our chosen domain are as follows:

$$\begin{aligned} f_1(x_1,x_2) &= e^{x_1-x_2} - \sin(x_1+x_2) = 0 \\ f_2(x_1,x_2) &= x_1^2 x_2^2 - \cos(x_1+x_2) = 0 \\ \text{with } D &= \{(x_1,x_2) : -10 \le x_1 \le 10, \\ &\qquad -10 \le x_2 \le 10\} \end{aligned}$$

The total number of roots is unspecified in the literature.

Problem 2:

From Tsoulos and Stavrakoudis [4] and also Sacco and Henderson [5], the system of equations and our chosen domain are following:

$$\begin{aligned} f_1(x_1,x_2) &= 0.5\sin(x_1x_2) - 0.25\frac{x_2}{\pi} - 0.5x_1 \\ &= 0 \\ f_2(x_1,x_2) &= \left(1 - \frac{0.25}{\pi}\right)\left(e^{2x_1} - e\right) + e\frac{x_2}{\pi} - 2ex_1 \\ &= 0 \\ \text{with } D &= \{(x_1,x_2) : -1 \le x_1 \le 3, \\ &\qquad -17 \le x_2 \le 4\} \end{aligned}$$

Problem 3:

From Luo et al. [2] and also Krzyworzcka [12], the system of equations and our chosen domain are below.

$$\begin{aligned} f_1(\mathbf{x}) &= x_1 + \frac{x_2^2 x_4 x_6}{4} + 0.75 = 0 \\ f_2(\mathbf{x}) &= x_2 + 0.405e^{1+x_1x_2} - 1.405 = 0 \\ f_3(\mathbf{x}) &= x_3 - \frac{x_4x_6}{2} + 1.5 = 0 \\ f_4(\mathbf{x}) &= x_4 - 0.605e^{1-x_3^2} - 0.395 = 0 \\ f_5(\mathbf{x}) &= x_5 - \frac{x_2x_6}{2} + 1.5 = 0 \\ f_6(\mathbf{x}) &= x_6 - x_1x_5 = 0 \\ \text{with } \mathbf{x} &= (x_1,x_2,\ldots,x_6)^T \in \mathbb{R}^6 \\ \text{and } D &= \{\mathbf{x} : -5 \le x_i \le 5 \;\; i = 1,2,\ldots,6\} \end{aligned}$$

Problem 4:

From Luo et al. [2], the system of equations is defined as follow:

$$\begin{aligned} f_1(x_1,x_2,x_3) &= x_1x_2 - (x_1-2x_3)(x_2-2x_3) - 165 \\ &= 0 \\ f_2(x_1,x_2,x_3) &= \frac{x_1x_2^3}{12} - \frac{(x_1-2x_3)(x_2-2x_3)^3}{12} - 9369 \\ &= 0 \\ f_3(x_1,x_2,x_3) &= \frac{2(x_2-x_3)^2(x_1-x_3)^2x_3}{x_2+x_1-2x_3} - 6835 \\ &= 0 \\ \text{with } D &= \{(x_1,x_2,x_3) : -40 \le x_i \le 40 \\ &\qquad i = 1,2,3\} \end{aligned}$$

Problem 5:

From Sacco and Henderson [5], the system of equations is defined as follow:

$$\begin{aligned} f_1(\mathbf{x}) &= 2x_1 + x_2 + x_3 + x_4 + x_5 - 6 = 0 \\ f_2(\mathbf{x}) &= x_1 + 2x_2 + x_3 + x_4 + x_5 - 6 = 0 \\ f_3(\mathbf{x}) &= x_1 + x_2 + 2x_3 + x_4 + x_5 - 6 = 0 \\ f_4(\mathbf{x}) &= x_1 + x_2 + x_3 + 2x_4 + x_5 - 6 = 0 \\ f_5(\mathbf{x}) &= x_1x_2x_3x_4x_5 - 1 = 0 \\ \text{with } \mathbf{x} &= (x_1,x_2,\ldots,x_5)^T \in \mathbb{R}^5 \\ \text{and } D &= \{\mathbf{x} : -10 \le x_i \le 10 \, i = 1,2,\ldots,5\} \end{aligned}$$

Problem 6:

From Sacco and Henderson [5] and also Kearfott [13], the system is as follows:

$$\begin{aligned} f_1(\mathbf{x}) &= 4.731 \times 10^{-3}x_1x_3 - 0.3578x_2x_3 - 0.1238x_1 \\ &\quad + x_7 - 1.637 \times 10^{-3}x_2 - 0.9338x_4 - 0.3571 \\ &= 0 \\ f_2(\mathbf{x}) &= 0.2238x_1x_3 + 0.7623x_2x_3 + 0.2638x_1 - x_7 \\ &\quad - 0.07745x_2 - 0.6734x_4 - 0.6022 = 0 \\ f_3(\mathbf{x}) &= x_6x_8 + 0.3578x_1 + 4.731 \times 10^{-3}x_2 = 0 \\ f_4(\mathbf{x}) &= -0.7623x_1 + 0.2238x_2 + 0.3461 = 0 \\ f_5(\mathbf{x}) &= x_1^2 + x_2^2 - 1 = 0 \\ f_6(\mathbf{x}) &= x_3^2 + x_4^2 - 1 = 0 \\ f_7(\mathbf{x}) &= x_5^2 + x_6^2 - 1 = 0 \\ f_8(\mathbf{x}) &= x_7^2 + x_8^2 - 1 = 0 \\ \text{with } \mathbf{x} &= (x_1,x_2,\ldots,x_8)^T \in \mathbb{R}^8 \\ \text{and } D &= \{\mathbf{x} : -1 \le x_i \le 1 \;\; i = 1,2,\ldots,8\} \end{aligned}$$

6.2. Numerical Results

Problem 1:

Using the following parameter values:

$$\begin{aligned} m_{cluster} &= 250,\ \gamma = 0.2,\ \varepsilon = 10^{-7},\ \delta = 0.01, \\ k_{\text{cluster}} &= 10,\ m = 250,\ r = 0.95,\ \theta = \frac{\pi}{4}, \\ k_{\max} &= 250, \end{aligned}$$

the proposed technique gives results in **Table 1** in a single-run. In **Fig. 3**, the roots appear on the intersection of the two graphs, $f_1(x_1,x_2) = 0$ and $f_2(x_1,x_2) = 0$. Here we obtained simultaneously all six distinct roots in a single run with time 0.8 s. With the same parameter values but using pseudo-random points in step 1 of Clustering phase and SDOA phase, we have found simultaneously all those

Table 1. Results for problem 1.

Solution	x_1	x_2	$f_1(x_1,x_2)$	$f_2(x_1,x_2)$
1	−6.437160	0.155348	2.873270e−07	1.734320e−06
2	−0.932122	1.067870	3.938960e−07	−8.791480e−07
3	−0.155283	6.439840	−6.681690e−07	−4.95752e−06
4	0.163333	6.122430	5.424030e−07	−5.642370e−06
5	0.667121	0.690103	8.580740e−08	−6.350610e−07
6	−6.117110	−0.163476	7.602760e−08	5.699980e−06

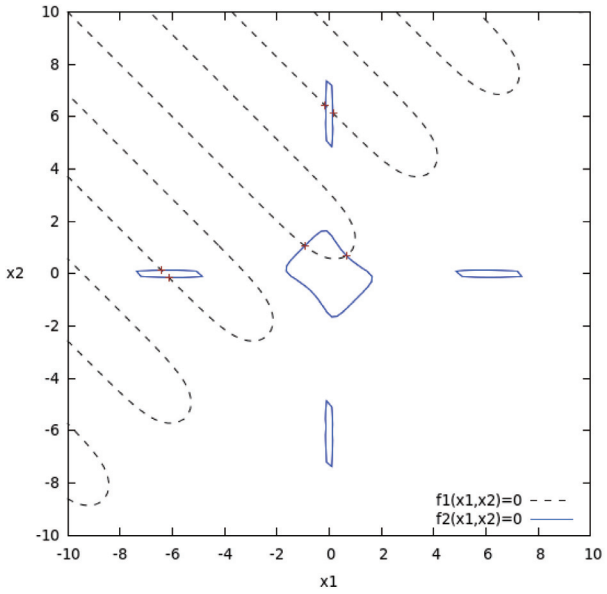

Fig. 3. The graph of $f_1(x_1,x_2)=0$ and $f_2(x_1,x_2)=0$ for problem 1.

six roots in 65 runs out of 100 runs with average time 0.76 s.

Problem 2:

Using the following parameter values:

$$m_{\text{cluster}}=2000,\ \gamma=0.3,\ \varepsilon=10^{-7},\ \delta=0.1,\ k_{\text{cluster}}=10,\ m=300,\ r=0.95,\ \theta=\frac{\pi}{4},k_{\max}=300,$$

the results of the proposed technique are presented in **Table 2**. We can see in **Fig. 4** that the roots appear as the intersection of the two graphs $f_1(x_1,x_2)=0$ and $f_2(x_1,x_2)=0$. Here we obtained simultaneously all the twelve distinct roots in a single run which took 2.98 s. With the same parameter values but using pseudo-random points, from total 100 runs of the proposed technique, we have found simultaneously all those twelve roots in 10 runs with average time 2.15 s. In particular, in the region $D^*=\{(x_1,x_2):\ 0.25\le x_1\le 1,\ 1.5\le x_2\le 2\pi\}$ there are only two roots as also have been obtained by Tsoulos and Stavrakoudis [4] and also Sacco and Henderson [5].

Problem 3:

A single run of the proposed technique was performed with the following parameter values:

$$m_{\text{cluster}}=1000,\ \gamma=0.1,\ \varepsilon=10^{-7},\ \delta=0.5,\ k_{\text{cluster}}=15,m=420,r=0.95,\theta=\frac{\pi}{4},k_{\max}=420$$

and the results are presented in **Table 3**. Luo et al. [2] and Krzyworscka [12] reported only one root as a solution, which is $\mathbf{x}=(-1,1,-1,1,-1,1)^T$. We obtained simultaneously two distinct roots in a single run which took 7.66 s. With the same parameter values but using pseudo-random points, we have discovered simultaneously all those two roots in 69 runs out of 100 runs with average time 15.8 s.

Problem 4:

We can eliminate x_2 from the system of nonlinear equations and find the roots as solutions of the following system of two nonlinear equations in two variables:

$$\begin{aligned} g_1(x_1,x_3) &= \frac{x_1x_2^3}{12}-\frac{(x_1-2x_3)(x_2-2x_3)^3}{12}-9369\\ &=0\\ g_2(x_1,x_3) &= \frac{2(x_2-x_3)^2(x_1-x_3)^2x_3}{x_2+x_1-2x_3}-6835\\ &=0\\ \text{with } x_2 &= 2x_3-x_1+\frac{165}{2x_3} \end{aligned}$$

A single run of the proposed technique was performed with the following parameter values:

$$m_{\text{cluster}}=2000,\ \gamma=0.001,\ \varepsilon=10^{-7},\ \delta=0.5,\ k_{\text{cluster}}=10,\ m=500,\ r=0.95,\ \theta=\frac{\pi}{4},\ k_{\max}=500,$$

and the results are presented in **Table 4**. Here we obtained simultaneously six distinct roots in a single run with time 0.97 s. Whereas using pseudo-random points, from total 100 runs of the proposed technique, we have found simultaneously all those six roots in 5 runs with average time 1.12 s.

Figure 5 present the graph of this system. We note that there are six points of intersection of the graph $g_1(x_1,x_3)=0$ and $g_2(x_1,x_3)=0$, which confirm that the system of equations of problem 4 has six distinct roots in the given domain. Luo et al. [1] also reported the six roots

Table 2. Results for problem 2.

Solution	x_1	x_2	$f_1(x_1,x_2)$	$f_2(x_1,x_2)$
1	−0.260599	0.622531	−7.744410e−08	−1.170240e−06
2	1.530510	−10.202200	−1.439340e−06	−1.100660e−05
3	1.663420	−16.282800	1.834820e−07	5.529120e−07
4	1.654580	−15.819200	−1.185450e−06	1.283770e−05
5	1.433950	−6.820760	−1.089920e−06	−8.665240e−06
6	1.578220	−12.176700	2.369380e−06	−1.521930e−05
7	1.337430	−4.140440	−5.178290e−07	8.814390e−06
8	0.500000	3.141590	−2.756650e−08	2.997340e−07
9	0.299449	2.836930	1.387110e−07	−4.428580e−07
10	1.604570	−13.362900	2.601460e−06	1.791030e−05
11	1.294360	−3.137220	−1.090360e−07	−8.951410e−06
12	1.481320	−8.383610	−2.211870e−06	1.276260e−05

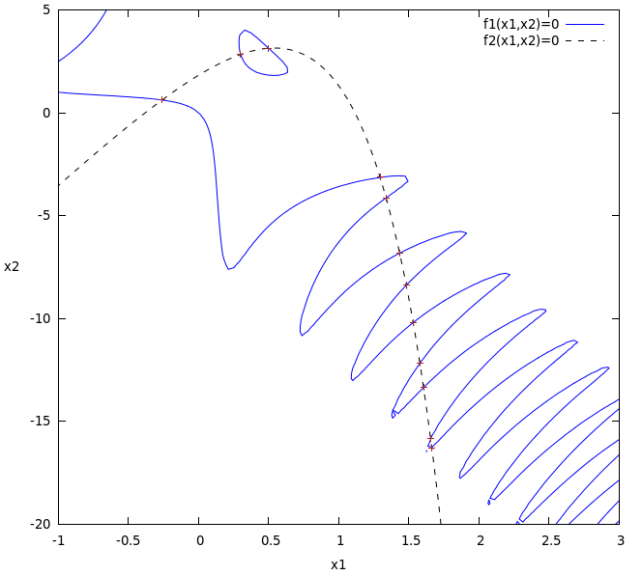

Fig. 4. The graph of $f_1(x_1,x_2)=0$ and $f_2(x_1,x_2)=0$ for problem 2.

of the system:

$$\mathbf{x}_1 = (12.5655, 22.8949, 2.7898)^T,$$
$$\mathbf{x}_2 = -\mathbf{x}_1,$$
$$\mathbf{x}_3 = (8.943089, 23.271482, 12.912774)^T,$$
$$\mathbf{x}_4 = -\mathbf{x}_3,$$
$$\mathbf{x}_5 = (-2.363740, 35.756376, 3.015078)^T \text{ and}$$
$$\mathbf{x}_6 = -\mathbf{x}_5.$$

Problem 5:

A single run of the proposed technique was performed with the following parameter values:

$$m_{\text{cluster}} = 3000, \gamma = 0.1, \varepsilon = 5 \times 10^{-4}, \delta = 0.01,$$
$$k_{\text{cluster}} = 10, m = 200, r = 0.95, \theta = \frac{\pi}{4}, k_{\max} = 200$$

and the results are presented in **Table 5**. Here we obtained simultaneously three distinct roots in a single run which took 2.15 s. With the same parameter values but using pseudo-random points in step 1 of Clustering phase and SDOA phase, from total 100 runs of the proposed technique, we have found simultaneously all those three roots in 3 runs with average time 0.69 s.

Table 3. Results for problem 3.

Solution	$x_i\ (i=1,2,\ldots,6)$	$f_i(\mathbf{x})\ (i=1,2,\ldots,6)$
Sol. 1	−1	0
	1	0
	−1	0
	1	0
	−1	0
	1	0
Sol. 2	−1.043200	−1.737000e−07
	−0.550936	−1.505690e−07
	0.431936	−6.547360e−07
	1.759660	1.496010e−07
	−2.104870	−1.618560e−07
	2.195810	2.951250e−07

Problem 6:

A single run of the proposed technique was performed with the following parameter values:

$$m_{\text{cluster}} = 1500, \gamma = 0.2, \varepsilon = 10^{-6}, \delta = 0.01,$$
$$k_{\text{cluster}} = 5, m = 300, r = 0.95, \theta = \frac{\pi}{4}, k_{\max} = 300$$

and the results are presented in **Table 6**. Here we obtained simultaneously sixteen distinct roots in a single run which took 23.45 s. With the same parameter values but using pseudo-random points in step 1 of Clustering phase and SDOA phase, from total 100 runs of the proposed technique , we have found simultaneously all those sixteen roots in 100 runs with average time 28.51 s.

7. Conclusions

Combination of the proposed Clustering technique with SDOA have been shown the capability to locate and find all the real roots of systems of nonlinear equations in the

Table 4. Results for problem 4.

Solution	x_1	x_2	x_3	$f_1(x_1,x_2,x_3)$	$f_2(x_1,x_2,x_3)$	$f_2(x_1,x_2,x_3)$
1	−12.256500	−22.894900	−2.789820	−2.842170e−14	−8.305780e−04	5.900980e−05
2	−8.943090	−23.271500	−12.912800	−2.842170e−14	−5.675470e−04	−2.559920e−03
3	8.943090	23.271500	12.912800	−2.842170e−14	−5.675470e−04	−2.559920e−03
4	12.256500	22.894900	2.7898200	−2.842170e−14	−8.305780e−04	5.900980e−05
5	2.363740	−35.756400	−3.015080	−2.842170e−14	−3.012750e−03	2.403290e−04
6	−2.363740	35.756400	3.015080	−2.842170e−14	−3.012750e−03	2.403290e−04

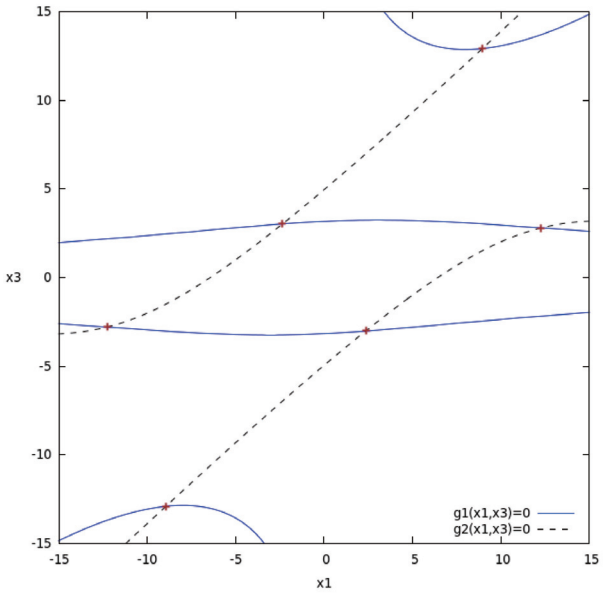

Fig. 5. The graph of $g_1(x_1,x_3)=0$ and $g_2(x_1,x_3)=0$.

Table 5. Results for problem 5.

Solution	$x_i\,(i=1,2,3,4,5)$	$f_i(\mathbf{x})\ \ (i=1,2,3,4,5)$
Sol. 1	0.915990	6.9e−05
	0.915939	1.8e−05
	0.915901	−2e−05
	0.915894	−2.7e−05
	1.420360	−3.5008e−04
Sol. 2	0.999970	2.1e−05
	0.999842	−1.07e−04
	0.999895	−5.4e−05
	1.000190	2.36e−04
	1.000160	5.09531e−05
Sol. 3	−0.579156	−8.7e−05
	−0.579098	−2.9e−05
	−0.578951	1.18e−04
	−0.579052	1.7e−05
	8.895330	1.58588e−04

set problems, only in a single run for each problem and without a priori knowledge of the number of the roots. The use of Sobol sequence of points instead of pseudo-random points to generate initial populations of points in the feasible region may increase the potential capacity of SDOA during diversification in the early phase to locate the potential positions of the candidate roots of the system of nonlinear equations.

Acknowledgements

The authors are grateful to the anonymous referees for comments and suggestions which have helped to strengthen this paper. Particular thanks are due to Novriana Sumarti for generous help in the process of editing this paper. The authors also acknowledge support from ITB Research Grant No.238 K/I1.C01/PL/2015.

References:

[1] K. Tamura and K. Yasuda,"Spiral Dynamics Inspired Optimization," J. of Advanced Computational Intelligence and Intelligent Informatics (JACIII), Vol.15, No.8, pp. 1116-1122, 2011.

[2] Y. Z. Luo, G. J. Tang, and L. N. Zhou, "Hybrid approach for solving systems of nonlinear equations using chaos optimization and quasi-Newton method," Applied Soft Computing, Vol.8, pp. 1068-1073, 2008.

[3] R. L. Burden and J. D. Faires, "Numerical Analysis," 7^{th} ed., Brooks/Cole, 2001.

[4] I. G. Tsoulos and A. Stavrakoudis, "On locating all roots of systems of nonlinear equations inside bounded domain using global optimization methods," Nonlinear Analysis: Real World Applications, Vol.11, pp. 2465-2471, 2010.

[5] W. F. Sacco and N. Henderson, "Finding all solutions of nonlinear systems using a hybrid meta-heuristic method with Fuzzy Clustering Means," Applied Soft Computing, Vol.11, pp. 5424-5432, 2011.

[6] C. Grosan and A. Abraham, "A new approach for solving Nonlinear equations systems," IEEE Trans. Systems, Man and Cybernetics, Vol.38, No.3, pp. 698-714, 2008.

[7] W. Song, Y. Wang, H. X. Li, and Z. Cai, "Locating multiple optimal solutions of nonlinear equations systems based on multi objective optimization," IEEE Trans. Evol. Comput., in press.

[8] V. Aggarwal, "Solving transcendental equations using Genetic Algorithm," http://web.mit.edu/varun_ag/www/ste_gas.pdf

[9] R. Seydel, "Tools for Computational Finance," Springer-Verlag, 2002.

[10] S. Joe and S. Y. Kuo, "Constructing Sobol sequences with better two dimensional projections," SIAM J. Sci. Comput., Vol.30, pp. 2635-2654, 2008.

[11] K. Chen, P. Giblin, and A. Irving, "Mathematical Explorations with MATLAB," Cambridge University Press, 1999.

[12] S. Krzyworzcka, "Extension of the Lanczos and CGS methods to systems of nonlinear equations," J. Comput. Appl. Math., Vol.69, No.1, pp. 181-190, 1996.

[13] R. B. Kearfott, "Some tests of generalized bisection," ACM Trans. Math. Softw., Vol.13, No.3, pp. 197-220, 1987.

Table 6. Results for problem 6.

Solution	x_i $(i=1,2,\ldots,8)$	$f_i(\mathbf{x})$ $(i=1,2,\ldots,8)$	Solution	x_i $(i=1,2,\ldots,8)$	$f_i(\mathbf{x})$ $(i=1,2,\ldots,8)$
Sol. 1	0.164432	1.57788e−07	Sol. 2	0.164432	1.57788e−07
	−0.986388	2.95737e−08		−0.986388	2.95737e−08
	0.718453	1.41406e−07		0.718453	1.41406e−07
	−0.695576	−1.48e−07		−0.695576	−1.48e−07
	0.997196	−8.30832e−07		0.997196	−8.30832e−07
	0.0748383	6.84985e−07		−0.074838	6.84985e−07
	−0.527809	5.8866e−07		−0.527809	5.8866e−07
	−0.849363	−1.5375e−07		0.849363	−1.5375e−07
Sol. 3	0.164432	1.57788e−07	Sol. 4	0.671554	−6.5485e−07
	−0.986388	2.95737e−08		0.740955	−6.22162e−08
	0.718453	1.41406e−07		0.951893	4.77875e−07
	−0.695576	−1.48e−07		−0.306431	1.148e−07
	−0.997196	−8.30832e−07		0.963439	−9.13059e−07
	0.074838	6.84985e−07		0.267927	2.4121e−07
	−0.527809	5.8866e−07		0.404641	−4.1595e−07
	−0.849363	−1.5375e−07		−0.914475	−1.13549e−06
Sol. 5	0.164432	1.57788e−07	Sol. 6	0.671554	−2.43009e−07
	−0.986388	2.95737e−08		0.740956	6.12775e−07
	0.718453	1.41406e−07		−0.65159	−3.13585e−07
	−0.695576	−1.48e−07		−0.758571	3.386e−07
	−0.997196	−8.30832e−07		0.96216	5.68852e−07
	−0.074838	6.84985e−07		0.272485	−5.09859e−07
	−0.527809	5.8866e−07		−0.437578	−5.9175e−08
	0.849363	−1.5375e−07		−0.899181	9.76845e−07
Sol. 7	0.671554	2.7895e−07	Sol. 8	0.671554	2.92567e−09
	0.740955	6.11184e−07		0.740956	−4.14038e−07
	0.951893	−4.366e−07		0.951893	−4.366e−07
	−0.306432	1.148e−07		−0.306431	3.386e−07
	−0.963439	−9.13059e−07		0.963439	5.68852e−07
	0.267928	8.54073e−07		−0.267928	2.4121e−07
	0.404641	1.19905e−07		0.404642	1.19905e−07
	−0.914475	−1.13549e−06		0.914475	−3.26211e−07
Sol. 9	0.164432	9.92527e−08	Sol. 10	0.671554	−6.5485e−07
	−0.986388	1.91704e−07		0.740955	−6.22162e−08
	−0.947064	2.4186e−07		0.951893	−4.366e−07
	−0.321046	−1.48e−07		−0.306431	1.148e−07
	−0.997566	−8.30832e−07		−0.963439	−9.13059e−07
	0.069727	7.54212e−07		−0.267928	2.4121e−07
	0.411033	−2.21115e−07		0.404641	1.19905e−07
	−0.91162	−8.48511e−07		0.914475	−1.13549e−06
Sol. 11	0.671554	−2.12574e−07	Sol. 12	0.164431	−1.09689e−07
	0.740955	4.710808e−07		−0.986389	9.39254e−07
	−0.651591	−3.13585e−07		−0.947064	−1.1594e−07
	−0.758571	1.148e−07		−0.321046	3.905e−07
	−0.96216	−9.13059e−07		−0.997566	8.13082e−07
	0.272485	7.93322e−07		−0.069727	7.54212e−07
	−0.437578	−5.9175e−08		0.411033	−2.21115e−07
	−0.899181	9.76845e−07		0.911620	−8.48511e−07
Sol. 13	0.671554	1.89282e−08	Sol. 14	0.671554	−2.12574e−07
	0.740956	−1.0235e−07		0.740955	4.71808e−07
	−0.651591	−3.13585e−07		−0.651591	−3.13585e−07
	−0.758571	3.386e−07		−0.758571	1.148e−07
	−0.962160	5.68852e−07		0.962160	−9.13059e−07
	−0.272485	7.93322e−07		−0.272485	7.93322e−07
	−0.437578	−5.9175e−08		−0.437578	−5.9175e−08
	0.899181	9.76845e−07		0.899181	9.76845e−07
Sol. 15	0.164432	9.92527e−08	Sol. 16	0.164432	9.92527e−08
	−0.986388	1.91704e−07		−0.986388	1.91704e−07
	−0.947064	2.4186e−07		−0.947064	2.4186e−07
	−0.321046	−1.48e−07		−0.321046	−1.48e−07
	0.997566	−8.30832e−07		0.997566	−8.30832e−07
	0.069727	7.54212e−07		−0.069727	7.54212e−07
	0.411033	−2.21115e−07		0.411033	−2.21115e−07
	−0.911620	−8.48511e−07		0.911620	−8.48511e−07

Name:
Kuntjoro Adji Sidarto

Affiliation:
Department of Mathematics, Faculty of Mathematics and Natural Sciences, Institut Teknologi Bandung

Address:
Jl. Ganesha 10, Bandung 40132, Indonesia

Brief Biographical History:
1976 B.Sc. in Mathematics, Institut Teknologi Bandung
1977 Faculty Staff, Department of Mathematics, Faculty of Mathematics and Natural Sciences, Institut Teknologi Bandung
1978 DEA in Mathematics, University of Montpellier II, France
1981 Ph.D. in Mathematics, University of Montpellier II, France

Main Works:
- "An Investigation on Gas Lift Performance Curve in an Oil-Producing Well," Int. J. of Mathematics and Mathematical Sciences, Vol.2007, Article ID 81519, 15 pages.
- "A model for transmission of partial resistance to anti-malarial drugs," Mathematical Biosciences and Engineering, Vol.6, No.3, pp. 649-662, 2009.

Membership in Academic Societies:
- Indonesian Mathematical Society

Name:
Adhe Kania

Affiliation:
Department of Mathematics, Faculty of Mathematics and Natural Sciences, Institut Teknologi Bandung

Address:
Jl. Ganesha 10, Bandung 40132, Indonesia

Brief Biographical History:
2005 B.Sc. in Mathematics, Institut Teknologi Bandung
2013 M.Sc. in Computational Sciences, Institut Teknologi Bandung
2014 Faculty Staff, Department of Mathematics, Faculty of Mathematics and Natural Sciences, Institut Teknologi Bandung