International Journal of Computational Intelligence and Applications
Vol. 14, No. 3 (2015) 1550017 (29 pages)
© Imperial College Press
DOI: 10.1142/S1469026815500170

General Relativity Search Algorithm: A Global Optimization Approach

Hamzeh Beiranvand* and Esmaeel Rokrok†

Department of Electrical Engineering
Lorestan University, Lorestan
Khoramabad, Lorestan, Iran
**beiranvand.ha@fe.lu.ac.ir*
†esmaeel.rokrok@gmail.com

Received 16 May 2014
Revised 11 June 2015
Published 14 September 2015

In this paper, a novel metaheuristic optimization algorithm inspired by General Relativity Theory (GRT) is presented. In this method that we named General Relativity Search Algorithm (GRSA), a population of particles is considered in a space free from all external non-gravitational fields and propel toward a position with least action. Based on GRT, particles have conserved masses and move along geodesic trajectories in a curved space. Step length and step direction for updating the particles are separately computed using particles velocity and geodesics, respectively. Velocity of particles is obtained by their energy–momentums. According to physical action principle, a population of particles goes to the position with minimum action. By inspiring this physical principle, GRSA will lead variables of an optimization problem move toward the optimal point. Performance of the proposed optimization algorithm is investigated by using several standard test functions and optimal Power System Stabilizers (PSSs) design in a multi-machine power system as a real-world application. Numerical simulations results demonstrate the efficiency, robustness and convergence speed of GRSA in solving various problems.

Keywords: Evolutionary algorithm; optimization; general relativity theory; standard test functions, power system stabilizer.

1. Introduction

The fundamental problem of optimization is to arrive at the best possible decision in any given set of circumstances.[1] A useful first step in taking an optimization problem is to classify the problem and, in particular, to discover whether some specialized algorithm is appropriate for its solution.[1] Many conventional techniques such as gradient-based search algorithms and various mathematical programming methods have been proposed to deal with optimization problems. However, these techniques have severe limitations in handling nonlinear, discontinuous functions and constraints because exact methods are more likely to get stuck at local optima.[2] Metaheuristic Algorithms (MHAs) do not rely on gradient information, whilst exact methods depend on. In this situation, evolutionary algorithms are still able to work satisfactorily.

In the last three decades, different MHAs have been introduced for solving optimization problems. Some of MHAs have received increased interest such as Genetic Algorithm (GA),[3] Ant Colony Optimization (ACO),[4] Particle Swarm Optimization (PSO),[5] Bacterial Foraging Algorithm (BFA),[6] Artificial Bee Colony (ABC),[7] Differential Evolution (DE),[8] Seeker Optimization Algorithm (SOA),[9] Gravitational Search Algorithm (GSA)[10] and another ones. These algorithms usually start with an initial random population of feasible solutions and then evolve and change the population position to obtain the global optimal point of the objective function. It can be understood that there are two important parameters that control the evolution of a solution.[11] These parameters are the step length and step direction. While, all of the above optimization algorithms use this process inherently. There is insufficient attention made to these parameters in MHAs.

The MHAs solve different practical optimization problems.[12] However, there is no specific algorithm to achieve the best solution for all optimization problems. Some algorithms result in a better solution for some particular problems rather than others. Hence searching for new heuristic optimization algorithm is an open problem.[13]

This paper presents a novel optimization algorithm inspired by GRT that we named it General Relativity Search Algorithm (GRSA). In GRSA, an initial population of particles will be generated randomly. Position components of the particles are represented as a tensor. A position where particles find least physical action is considered as the best position. Particles move toward the best position on geodesics trajectories based on their energy–momentum. Geodesic trajectories are formed according to geometry of the curved space. These trajectories have minimum resistance against particle motion under gravitational fields. Energy or mass in the space is the source of gravitational fields. A particle with greater energy–momentum yields greater gravitational field. Therefore, it has more effect on creating gravitational field and forming geodesic trajectories in the curved space. Tensor of particles changes in each step of iterations. Any particle obtains different mass and energy from previous stage. After the end of iterations, the best position for particles will be obtained. GRSA is tested using both benchmark functions and a real-world application. The results validate the efficiency of GRSA as a global optimization tool.

2. Background Review

Holland introduced GA in 1975.[3] Thereafter, a new era in computer science and engineering applications began. Due to the some prominent properties such as computational simplicity, flexibility, ease of use and local optimal point avoidance, the new technique was accepted warmly. Therefore, new similar nature inspired optimization algorithms were rapidly developed and MAs became popular. MAs affect the research and development of computer sciences and engineering, profoundly. Nowadays, MAs are widely used to solve different engineering applications.

The MHAs mimic natural, social, biological and physical processes. Most of MHAs are inspired from nature and they can be classified into (1) physical-based, (2) evolution-based, (3) social-based, and (4) swarm-based optimization algorithms. However, hybrid versions and their derivatives are common. For example, GA mimics the process of natural evolution. The two main concepts of natural evolution, which are natural selection and genetic dynamics, inspired the development of this method. PSO was developed through simulation of a simplified social system and traces its evolution to the emergent motion of a flock of birds searching for food.[5] DE developed in Ref. 8. DE uses weighted differences between solution vectors to change the population and employs a greedy selection process with inherent elitist features. ABC tries to model natural behavior of real honey bees in food foraging.[7] Also, GSA method is based on the law of gravity. Law of gravity asserts: "Every particle in the universe attracts every other particle with a force that is directly proportional to the product of their masses and inversely proportional to the square of the distance between them".[10] A list of popular MHAs is provided in Table 1. MHAs in Table 1 are sorted according to the time-line of their emergence.

Table 1. Time-line of the emergence of MHAs.

MHAs	Base	Year
Genetic Algorithm (GA)	Evolution	1975 Ref. 3
Genetic Programming (GP)	Evolution	1992 Ref. 14
Evolution Strategy (ES)	Evolution	1994 Ref. 15
Particle Swarm Optimization (PSO)	Swarm	1995 Ref. 5
Ant Colony Optimization (ACO)	Swarm	1996 Ref. 4
Differential Evolution (DE)	Evolution	1997 Ref. 8
Evolutionary Programming (EP)	Evolution	1999 Ref. 16
Marriage in Honey Bees Optimization Algorithm (MBO)	Swarm	2001 Ref. 17
Artificial Immune Algorithms (AIA)	Evolution	2002 Ref. 18
Bacterial Foraging Algorithm (BFA)	Evolution	2002 Ref. 6
Electromagnetism-Like Algorithm (ELA)	Physical	2003 Ref. 19
Gravitational Local Search (GLSA)	Physical	2003 Ref. 20
Artificial Fish-Swarm Algorithm (AFSA)	Swarm	2003 Ref. 21
Particle Collision Algorithm (PCA)	Physical	2005 Ref. 22
Termite Algorithm (TA)	Swarm	2005 Ref. 23
Harmony Search Algorithm (HSA)	Social	2005 Ref. 24
Evolutionary Algorithm (EA)	Evolution	2006 Ref. 25
Invasive Weed Optimization (IWO)	Evolution	2006 Ref. 26
Multi Point Tabu Search Algorithm (MPTSA)	Social	2006 Ref. 27
Cat Swarm Optimization (CSO)	Swarm	2006 Ref. 28
Seeker Optimization Algorithm (SOA)	Social	2006 Ref. 9
Big-Bang Big-Crunch (BBBC)	Physical	2006 Ref. 29
Small-World Optimization Algorithm (SWOA)	Physical	2006 Ref. 30
Multi Point Simulated Annealing Algorithm (MPSA)	Physical	2007 Ref. 31
Saplings Growing Up Algorithm (SGA)	Evolution	2007 Ref. 32
Central Force Optimization (CFO)	Physical	2007 Ref. 33

Table 1. (*Continued*)

MHAs	Base	Year
Imperialist Competitive Algorithm (ICA)	Social	2007 Ref. 34
Wasp Swarm Algorithm (WSA)	Swarm	2007 Ref. 35
Monkey Search (MS)	Swarm	2007 Ref. 36
Enzyme Algorithm (EA)	Evolution	2008 Ref. 37
Bee Collecting Pollen Algorithm (BCPA)	Swarm	2008 Ref. 38
Biogeography-Based Optimizer (BBO)	Evolution	2008 Ref. 39
Gravitational Search Algorithm (GSA)	Physical	2009 Ref. 10
Cuckoo Search (CS)	Swarm	2009 Ref. 40
Dolphin Partner Optimization (DPO)	Swarm	2009 Ref. 41
Artificial Bee Colony (ABC)	Swarm	2009 Ref. 7
Bat-inspired Algorithm (BA)	Swarm	2010 Ref. 42
Charged System Search (CSS)	Physical	2010 Ref. 43
Firefly Algorithm (FA)	Swarm	2010 Ref. 44
Galaxy-based Search Algorithm (GbSA)	Physical	2011 Ref. 45
Cuckoo Optimization Algorithm (COA)	Swarm	2011 Ref. 46
Artificial Chemical Reaction Optimization Algorithm (ACROA)	Physical	2011 Ref. 47
Krill Herd (KH)	Swarm	2012 Ref. 48
Curved Space Optimization (CSO)	Physical	2012 Ref. 49
Fruit fly Optimization Algorithm (FOA)	Swarm	2012 Ref. 50
Ray Optimization (RO) algorithm	Physical	2012 Ref. 51
Bird Mating Optimizer (BMO)	Swarm	2013 Ref. 52
Black Hole (BH) algorithm	Physical	2013 Ref. 53
Grey Wolf Optimizer (GWO)	Swarm	2014 Ref. 54

3. General Relativity Theory

General Relativity Theory (GRT) is the geometric theory of gravitation in modern physics.[55,56] GRT generalizes special relativity and Newton's law of universal gravitation, providing a unified description of gravity as a geometric property of space and time, or spacetime. The curvature of spacetime is directly related to the energy and momentum of whatever matter and radiation are present. There is no gravitational force deflecting objects from their natural, straight paths. Instead, gravity corresponds to changes in the properties of space and time, which in turn changes the straightest-possible paths that objects will naturally follow. The curvature is caused by the energy–momentum of matter. The stress–energy tensor is a tensor quantity in physics that describes the density and flux of energy and momentum in spacetime, generalizing the stress tensor of Newtonian physics. The stress–energy tensor is the source of the gravitational field in the Einstein field equations of general relativity, just as mass density is the source of such a field in Newtonian gravity.

In the general relativity, stress–energy tensor is being studied using Einstein's field equations. These equations are general relativity core and can be written in the form of tensor equation (1).

$$R_{ij} - \frac{1}{2} R g_{ij} = \frac{8\pi G}{c^4} T_{ij}, \tag{1}$$

where R_{ij} is Ricci tensor, R is the curvature scalar, g_{ij} is the geometry tensor, T_{ij} is the energy–momentum tensor (Einstein tensor), G is the gravitational constant and c is the speed of light. In a curved line coordination system, g_{ij} components are functions of coordination system and specify all elements and thus they form metric, and show the coordination system and curvature of the space. From Einstein's equations two results can be deduced. Firstly Mass is conserved and secondly Particle always moves along a geodesic. Mass and energy form particles. Mass and energy are transmutable to each other, while they have conserved value. The motion of matter from one place to another place causes an alteration of the gravitational field in the surrounding space.[57]

In GRT, a geodesic generalizes the notion of a "straight line" to curved spacetime. Importantly, the world line of a particle free from all external non-gravitational forces, is a particular type of geodesic. In GRT, gravity can be regarded as nota force but a consequence of curved spacetime geometry where the source of curvature is the stress–energy tensor. In a curved spacetime geometry three type of geodesics can be defined: (a) timelike geodesics have a tangent vector whose norm is negative, (b) null geodesics have a tangent vector whose norm is zero and (c) spacelike geodesics have a tangent vector whose norm is positive. An ideal particle whose gravitational field and size are ignored not subject to any non-gravitational field, will always follow timelike geodesics. Firs law of Newton replaces with this premise and motion and acceleration equation obtained from geodesic equations. Massless particles like the photon follow null geodesics and thus light ray follow null geodesics. Spacelike geodesics do not correspond to the path of any physical particle in a space that has space-sections orthogonal to a timelike Killing vector. In general relativity, total energy can be expressed as the sum of rest mass energy and kinetic energy as follow:

$$m\gamma c^2 = E_k + mc^2, \tag{2}$$

where m is the particle mass, c is the speed of light, E_k is the kinetic energy and γ is Lorentz transformation coefficient which is defined as:

$$\gamma = \sqrt{\frac{g_{tt}}{g_{tt} + g_{ss}\upsilon^2}}, \tag{3}$$

where g_{ss} and g_{tt} are spacetime geometry-dependent variables and v is the speed of particle motion. In GRT, 2-dimensional curved spacetime may imagine by a curved surface as illustrated in Fig. 1 in a 3-dimensional Cartesian coordination system.

4. General Relativity Search Algorithm

GRSA starts with an initial population of particles that their positions are generated randomly in the curved spacetime. A particle position is a feasible solution of an optimization problem. Positions of the particles are represented in a tensor. A particle tracks a geodesic based on its energy–momentum and surrounding gravitational fields in curved spacetime. It moves toward the best position where it has minimum

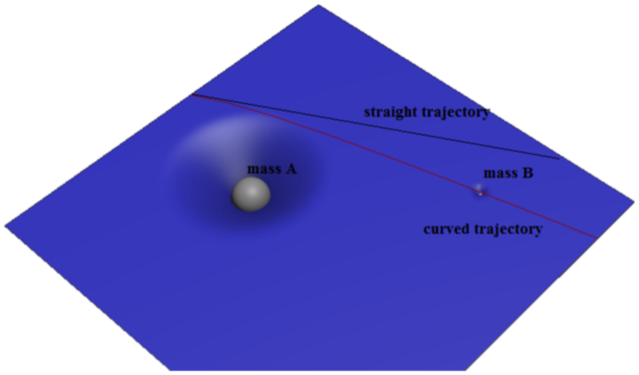


Fig. 1. Curvature of 2-dimensional spacetime in the presence of mass A (large) and B (small).

action. Objective function of the optimization problem is computed for all particles and considered as their actions. Step length and step direction for updating position of a particle is determined by its velocity and geodesic tangent. Particles with big gravitational fields have significant effect on forming geodesics for other particles, while evolution of these particles mainly depend on their energy–momentum and geodesic tangent.

GRSA evolves to find solution through iterative calculation of step length and step direction for any particle position. Step length and motion direction in GRSA are calculated using existing relativistic equations for energy–momentum, geometry of curved spacetime and geodesic tangent vectors. Finally, using Einstein's field equation the effect of gravitational field of particles on spacetime is incorporated. Therefore, new position of particles is yielded. In next subsections, different stages of the algorithm are presented.

4.1. *GRSA initiation*

Initial random population is set into a tensor as follows. Each element of tensor T will be defined as:

$$T_{ij}(1) = T_j^{\min} + \mathrm{rand}()(T_j^{\max} - T_j^{\min}), \quad i \in \{1, 2, \ldots, n\}, \quad j \in \{1, 2, \ldots, d\}, \quad (4)$$

where n is the number of particles and d is the particle or spacetime dimension. $T_j^{\max}$ and $T_j^{\min}$ are maximum and minimum of the jth element of any particle T_i, respectively. Function rand provides a uniform random number in the interval [0,1]. This initial tensor of search space, T, is uniformly divided into S search subspaces such that each search subspaces includes h particles. Figure 2 illustrates the notion of division of initial search space tensor into S search subspaces. Objective function of the optimization problems is computed for all particles and minimum action among all of them is selected as the best solution in this stage.

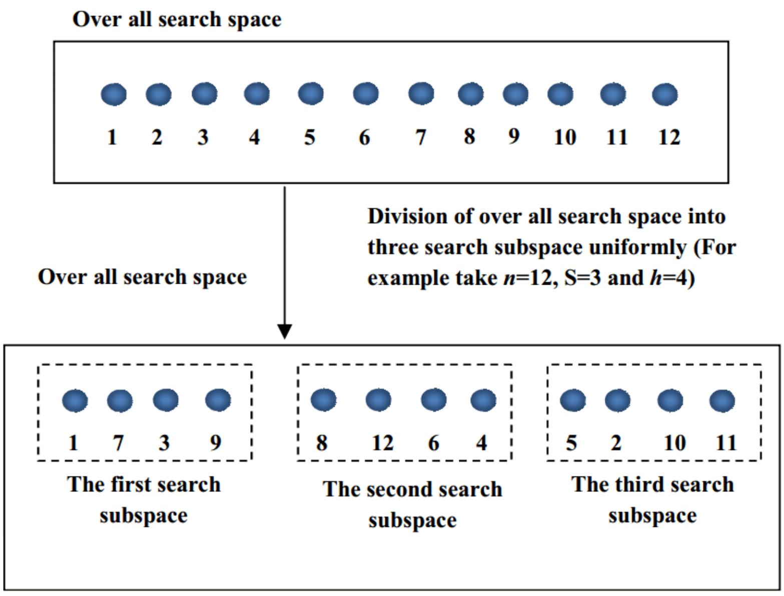

Over all search space
1 2 3 4 5 6 7 8 9 10 11 12
Division of over all search space into three search subspace uniformly (For example take n=12, S=3 and h=4)
Over all search space
1 7 3 9
The first search subspace
8 12 6 4
The second search subspace
5 2 10 11
The third search subspace

Fig. 2. Division of over all search space into subspaces uniformly.

4.2. *Step length computation*

In this stage, the search step length is calculated using relativistic total energy–momentum in Eq. (2). Initially, the ratio of kinetic energy to the rest mass energy is defined as:

$$\xi = \frac{E_k}{mc^2}. \tag{5}$$

We introduce an intuitive equation for ξ at time t (tth iteration) for ith test particle in sth ($s \epsilon \{1, 2, \dots, S\}$) subspace as:

$$\xi_i(t) = \frac{I_{\mathrm{rand}} - n}{n-1}\mathrm{GM}_1 + \frac{1 - I_{\mathrm{rand}}}{n-1}\mathrm{GM}_2, \tag{6}$$

where $\xi(t)$ is the energy conversion ratio at tth iteration ($t \epsilon \{1, 2, \dots, t_{\max}\}$), I_{rand} is a random selected index of one of the positions in sth subspace. For example, according to Fig. 3, if the third particle of the second search subspace is selected randomly then $I_{\mathrm{rand}} = 7$. Also, GM_1 and GM_2 are geometry coefficients of spacetime which specify the rate of energy conversion.

In order to keep conservation law, GM_1 and GM_2 must satisfy the following equation:

$$\mathrm{GM}_1 + \mathrm{GM}_2 = 1, \mathrm{GM}_1, \mathrm{GM}_2 \in [\mathrm{GM}_{\min}, \mathrm{GM}_{\max}], \tag{7}$$

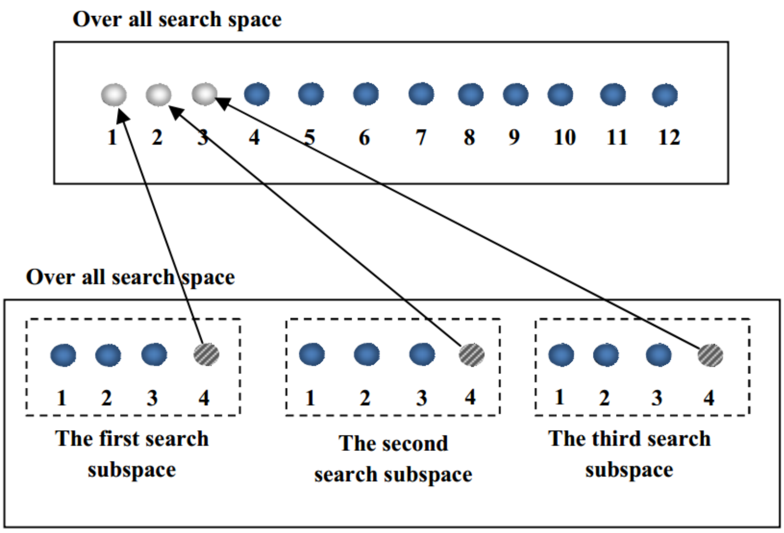

Over all search space
1
2
3
4
5
6
7
8
9
10
11
12
Over all search space
1
2
3
4
The first search subspace
1
2
3
4
The second search subspace
1
2
3
4
The third search subspace

Fig. 3. Mutation operation between worst position in over all search space and the best positions in all search subspaces.

where $\mathrm{GM}_{\min}$ and $\mathrm{GM}_{\max}$ are minimum and maximum limits of geometry coefficients, respectively. We define initial Lorentz transformation coefficient in Eq. (2) as:

$$\gamma_{0,i}(t) = 1 + \xi_i(t). \tag{8}$$

The quantity γ_0 is a scalar value that shows the change in energy–momentum of an agent. In order to transmute γ_0 into a vector form, γ, a random vector, K_g, is defined. K_g (is a $1 \times d$ vector) and its components are uniformly random numbers in range [0, 1]. We define final Lorentz coefficient vector, $\gamma_i(t)$, as:

$$\gamma_i(t) = \gamma_{0,i}(t) + (1 - \gamma_{0,i}(t))K_g. \tag{9}$$

Now, by using Eq. (3) particles velocity can be calculated as:

$$V_{ij}(t) = K_{V,ij}(t)\sqrt{\frac{1}{\gamma_{ij}^2(t)} - 1}, \tag{10}$$

where $V_{ij}(t)$ is the jth component of the velocity vector for ith particle at tth iteration. Variable $K_{V,ij}(t)$ that is called spacetime coefficient shows impact of spacetime geometry on particles motion. From Eq. (2), this coefficient is equal to $K_V = \sqrt{(g_{tt}/g_{ss})}$. We define $K_{V,ij}(t)$ as the distance of the best position and a random particle position in sth search subspace as:

$$K_{V,ij}(t) = |T_j^{\mathrm{Best},s}(t) - T_j^{\mathrm{rand},s}(t)|, \tag{11}$$

where $T_j^{\mathrm{Bset},s}$ is the jth component of the best position in sth search subspace and $T_j^{\mathrm{rand},s}$ is the jth component of a random position in sth search subspace. Thus, step length is obtained as follows:

$$\lambda_{ij}(t) = \omega_t V_{ij}(t), \quad i \in \{1, 2, \dots, n\},\ j \in \{1, 2, \dots, d\}, \tag{12}$$

where $\lambda_{ij}(t)$ is step length of jth component of ith particle at tth iteration. Variable ω_t is a weighted factor which decreased from 0.9 to 0.1 linearly with increasing iterations and will cause smaller step length in adjacent to the global optimal point. Similarly, for all particles in subspaces the step length can be calculated.

4.3. *Step direction computation*

Step direction for a particle is obtained from three existing types of its geodesics. Direction of timelike, spacelike and null geodesics is defined in the following relations, respectively:

$$\delta_{\mathrm{tl},i}(t) = \mathrm{sign}(T_i(t) - T_i(t-1)), \tag{13}$$

$$\delta_{\mathrm{sl},i}(t) = \mathrm{sign}(T_i(t) - \mathrm{TG}), \tag{14}$$

$$\delta_{\mathrm{null},i}(t) = \mathrm{sign}(T_i(t) - \mathrm{TB}_i), \tag{15}$$

where δ_{tl}, δ_{sl} and δ_{null} are the timelike, the spacelike and the null geodesic directions of ith particle at tth iteration, respectively. Sign is the signum function and δ_{tl}, δ_{sl} and $\delta_{\mathrm{null}}\, \epsilon \{-1, 0, 1\}$. Also, TG is the global best particle position in the tensor. TB_i is the best position that the particle is obtained in the exploration so far. In order to have a purposive search using different geodesics directions, we define final equation for particle step direction as:

$$\delta_{ij}(t) = -\mathrm{sign}(\delta_{tl,ij}(t) + K_{f,j}\delta_{sl,ij}(t) + (1 - K_{f,j})\delta_{\mathrm{null},ij}(t)), \tag{16}$$

where $\delta_{ij}(t)$ represents step direction of jth component of ith particle and K_f is a random vector with $K_{f,i}\, \epsilon \{0, 1\}$. As it is obvious from Eq. (16), if $K_{f,j}$ becomes one, then the corresponding component of timelike geodesic tangent will be in agreement with spacelike geodesic and else it will be in the same direction with null geodesic. Therefore, the tendency to move toward a particle with big energy–momentum increases along a timelike geodesic.

4.4. *Updating particles tensor*

By computing step length from Eq. (12) and step direction from Eq. (16), new position for particles is:

$$T_{ij}(t+1) = T_{ij}(t) + \lambda_{ij}(t)\delta_{ij}(t). \tag{17}$$

To improve search capabilities, a mutation operation will be performed for particles with worst positions in over all search space. In this process, numbers of S worst positions in over all search space are chosen and the mutation operation will be performed with the best position in all subspaces. Therefore, we apply Einstein's field equation to these positions. We can rewrite Einstein's field equation (i.e., Eq. (1)) as:

$$T_{ij} = \frac{c^4}{8\pi G} R_{ij} - \frac{1}{2} R \frac{c^4}{8\pi G} g_{ij} = \alpha_{1,j} R_{ij} + \alpha_{2,j} g_{ij}, \tag{18}$$

$$R_i \in \{T_1, T_2, \dots, T_S\} \mid f(T_1) \geq f(T_2) \geq \cdots \geq f(T_S), \tag{19}$$

$$g_i \in \{T_1^{(1)}, T_1^{(2)}, \dots, T_1^{(S)}\} \mid f(T_1^{(i)}) = \max_{i \in \{1,2,\dots,S\}} \{f(T_1^{(i)}), f(T_2^{(i)}), \dots, f(T_h^{(i)})\}, \tag{20}$$

where $f(T_i)$ denotes objective function value. R is the set of the worst positions in over all search space and g is the set of the best position of all search subspaces. Equations (19) and (20) stand for a minimization problem. Also, for a maximization problem, these equations can be rewritten as follow:

$$R_i \in \{T_1, T_2, \dots, T_S\} \mid f(T_1) \leq f(T_2) \leq \cdots \leq f(T_S). \tag{21}$$

$$g_i \in \{T_1^{(1)}, T_1^{(2)}, \dots, T_1^{(S)}\} \mid f(T_1^{(i)}) = \min_{i \in \{1,2,\dots,S\}} \{f(T_1^{(i)}), f(T_2^{(i)}), \dots, f(T_h^{(i)})\}. \tag{22}$$

To perform mutation operation, variables α_1 and α_2 are defined in a way that α_1 becomes a random vector with $\alpha_{1,j} \epsilon \{0,1\}$ and α_2 is complementary vector for α_1 just as in follows:

$$\alpha_{1,j} + \alpha_{2,j} = 1. \tag{23}$$

Therefore, the mutation operation is given by:

$$T_{kj}(t+1) = \alpha_{1,j} R_{kj}(t) + \alpha_{2,j} g_{kj}(t), \quad k \in \{1,2,\dots,S\},\ j \in \{1,2,\dots,d\}. \tag{24}$$

Finally, the tensor will be updated as:

$$T(t+1) = [T_1(t+1), T_2(t+1), \dots, T_n(t+1)]^T. \tag{25}$$

After updating the tensor, a new position for the particles population is obtained. In fact, an iteration of the algorithm has been completed. Figure 3 demonstrates the mutation operation relationship between particles and search subspaces. In this figure, particle positions are sorted in descending/ascending order for a minimization/maximization problem. Particles with bright color represent set R and those with dark color represent set g.

The steps of the proposed GRSA approach are illustrated below:

(i) Generate a random initial tensor for population of particles,
(ii) Divide initial tensor to some subspaces uniformly,
(iii) Calculate Action (cost function) for all particles,
(iv) Assign ratio of kinetic energy to the mass energy conversion for each particle,
(v) Calculate velocity of particles,
(vi) Compute step length of motion for particles,
(vii) Compute step direction for particles,

(viii) Update position of particles,
(ix) Perform mutation operation for number of S particles with worst positions using Einstein's equation,
(x) Repeat steps (iii) to (ix) until the stop criteria is satisfied,
(xi) End.

Figure 4 depicts the flowchart of GRSA.

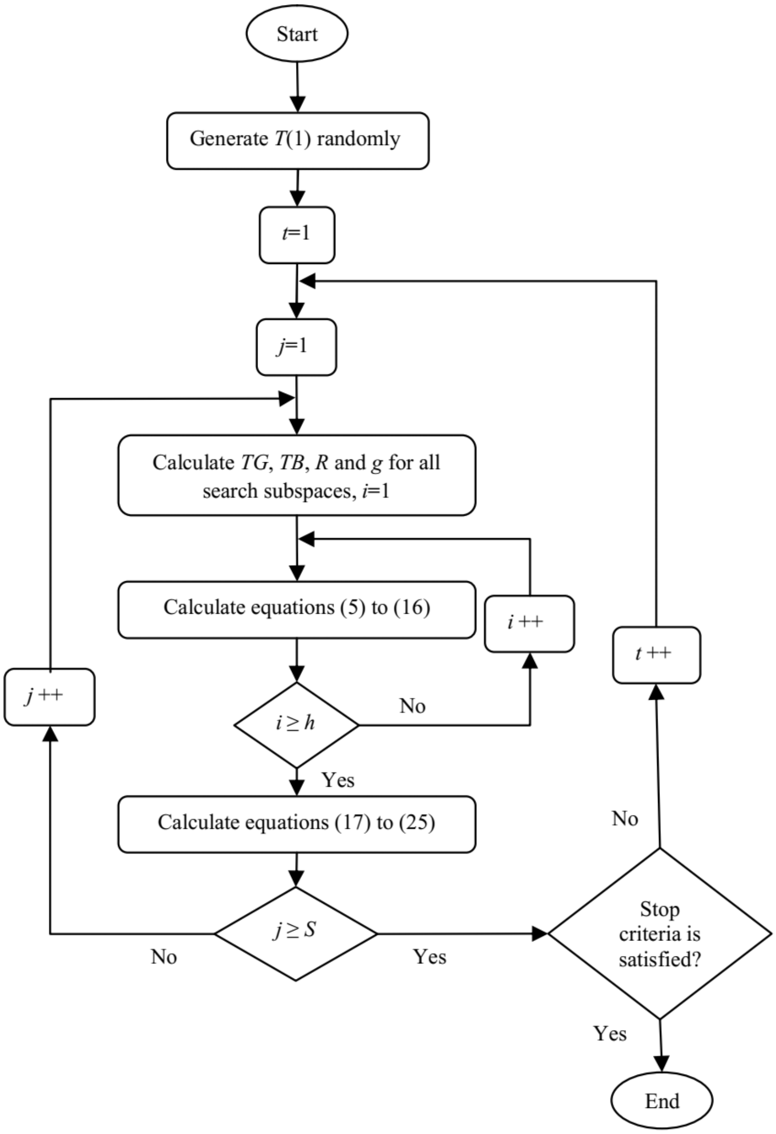

Start
Generate T(1) randomly
t=1
j=1
Calculate TG, TB, R and g for all search subspaces, i=1
Calculate equations (5) to (16)
i ++
t ++
j ++
i ≥ h
No
Yes
Calculate equations (17) to (25)
No
j ≥ S
No
Yes
Stop criteria is satisfied?
Yes
End

Fig. 4. Flowchart of the proposed GRSA.

Table 2. Strategy parameters of GRSA.

Strategy Parameter	Default Value
h (search subspace size)	$10 \times d$
S (number of search subspaces)	$[d/3]$
n (search space size)	$S \times h$
$t_{\max}$ (maximum number of iterations)	$10 \times n$
$\mathrm{GM}_{\max}$ (maximum of geometry coefficient)	1
$\mathrm{GM}_{\min}$ (minimum of geometry coefficient)	0
GM_1 (geometry coefficient)	0.01
GM_2 (geometry coefficient)	0.99

4.5. *Parameter tuning*

The strategy parameters for GRSA, h, S, n, $t_{\max}$, $\mathrm{GM}_{\min}$, $\mathrm{GM}_{\max}$, GM_1 and GM_2 connected to Eqs. (3), (4), (6) and (8), respectively, have to be chosen. The default parameter settings are summarized in Table 2. These parameters are obtained based on empirical observation. In general, the selection related parameters h, S, n and $t_{\max}$ are comparatively uncritical and can be chosen in a wide range without disturbing the procedure while GM_1 and GM_2 affect final solution drastically.

5. Comparison between GRSA and PSO

In this section, GRSA is compared theoretically with a frequently used evolutionary algorithm, PSO. Initially, we review PSO algorithms in detail.

The PSO algorithm is a search optimization method which was introduced by Kennedy and Eberhart.[5] In PSO, each particle evolves and changes its position in the search space with a velocity according to its own previous best position and the search space best position. Each particle updates its position as:

$$X_i(t+1) = X_i(t) + V_i(t+1), \tag{26}$$

where $X_i(t)$ and $V_i(t)$ are position and velocity vectors of the ith particle, respectively. Moreover, each particle updates its velocity as:

$$V_{ij}(t+1) = wV_{ij}(t) + c_1 r_{1,j}(XB_{ij}(t) - X_{ij}(t)) + c_2 r_{2,j}(XG_j(t) - X_{ij}(t)), \tag{27}$$

where $j \epsilon \{1, 2, \ldots, d\}$ and shows the dimension of the particle; c_1 and c_2 are two positive acceleration constants; $w \epsilon [0, 1)$ and is an inertia weight specifying how much of the particle's previous velocity is preserved; $r_{1,j}$ and $r_{2,j}$ are two uniform random numbers in the interval [0,1]; XB_{ij} is the ith particle best position so far; and XG_j is the best position found by the entire search space so far.

Both GRSA and PSO obtain optimal solution of an optimization problem by parallel search in the search space, however their evolution strategy is different. Some important differences of GRSA and PSO are as follows:

(1) Initially, the search ideas of these algorithms are different. PSO simulates the social behavior of birds and GRSA inspires by GRT which is a physical phenomena,
(2) In PSO, the velocity of an agent is calculated using only two best positions, XB_i and XG. But in GRSA, the agent speed is calculated based on the relativistic energy–momentum equation,
(3) In PSO, updating is performed considering the speed of the solutions. While, in GRSA the step length and step direction are used to form updating rule separately,
(4) In PSO, updating is performed without considering the distance between solutions. While, in GRSA the search step length is directly proportional to the distance between random solutions.

6. Numerical Simulation Results and Analysis

A set of 23 benchmark functions is used to perform numerical simulations. These functions simulate a vast spectrum of practical optimization problems with different complexity and dimension. The aim is to show the efficiency of existing or new proposed search algorithms through 23 benchmark functions which are given in Tables 3–5. In these tables, n is objective function dimension. More detailed description of each function is given in Ref. 16.

Functions f_1 to f_7 that are given in Table 3, are unimodal large dimensional functions. A function with single optimum point over its domain is a unimodal function. For these functions, the convergence speed of search algorithm is more important than the final results. The effectiveness of the proposed algorithm is compared with existing optimization algorithms such as basic GA and PSO through a set of numerical simulations. In the simulations, GA parameter for linear crossover is 0, the crossover probability is 0.9 and mutation probability is 0.1. PSO parameters

Table 3. Unimodal high-dimensional test functions.

Test Function	n	Domain	$f_{\min}$
$f_1(x) = \sum_{i=1}^{n} x_i^2$	30	$[-100, 100]^n$	0
$f_2(x) = \sum_{i=1}^{n} \lvert x_i\rvert + \prod_{i=1}^{n} \lvert x_i\rvert$	30	$[-10, 10]^n$	0
$f_3(x) = \sum_{i=1}^{n} \left(\sum_{j=1}^{i}(x_j)\right)^2$	30	$[-100, 100]^n$	0
$f_4(x) = \max_i\{\lvert x_i\rvert, 1 \le i \le n\}$	30	$[-100, 100]^n$	0
$f_5(x) = \sum_{i=1}^{n-1}[100(x_{i+1} - x_i^2)^2 - (x_i - 1)^2]$	30	$[-30, 30]^n$	0
$f_6(x) = \sum_{i=1}^{n} (\lfloor x_i + 0.5\rfloor)^2$	30	$[-100, 100]^n$	0
$f_7(x) = \sum_{i=1}^{n} i x_i^4 + \mathrm{random}[0, 1)$	30	$[-1.28, 1.28]^n$	0

Table 4. Multimodal high-dimensional test functions.

Test Function	n	Domain	$f_{\min}$
$f_8(x) = \sum_{i=1}^{n} -x_i \sin(\sqrt{\lvert x_i\rvert})$	30	$[-500, 500]^n$	-12569.5
$f_9(x) = \sum_{i=1}^{n}[x_i^2 - 10\cos(2\pi x_i) + 10]$	30	$[-5.12, 5.12]^n$	0
$f_{10}(x) = -20\exp\left(-0.2\sqrt{\frac{1}{n}\sum_{i=1}^{n} x_i^2}\right) - \exp\left(\frac{1}{n}\sum_{i=1}^{n}\cos(2\pi x_i)\right)$	30	$[-32, 32]^n$	0
$f_{11}(x) = \frac{1}{4000}\sum_{i=1}^{n} x_i^2 - \prod_{i=1}^{n}\cos\left(\frac{x_i}{\sqrt{i}}\right) + 1$	30	$[-600, 600]^n$	0
$f_{12}(x) = \frac{\pi}{n}\Big\{10\sin^2(\pi y_i) + \sum_{i=1}^{n-1}(y_i - 1)^2[1 + 10\sin^2(\pi y_{i+1})] + (y_n - 1)^2\Big\} + \sum_{i=1}^{n-1} u(x_i, 10, 100, 4)$, $y_i = 1 + \frac{1}{4}(x_i + 1)$, $u(x_i, a, k, m) = \begin{cases} k(x_i - a)^m, & x_i > a, \\ 0, & -a \le x_i \le a, \\ k(-x_i - a)^m, & x_i < a, \end{cases}$	30	$[-50, 50]^n$	0
$f_{13}(x) = 0.1\Big\{\sin^2(3\pi x_1) + \sum_{i=1}^{n-1}(x_i - 1)^2[1 + \sin^2(3\pi x_{i+1})] + (x_n - 1)^2[1 + \sin^2(2\pi x_n)]\Big\} + \sum_{i=1}^{n-1} u(x_i, 10, 100, 4)$	30	$[-50, 50]^n$	0

are set to $c_1 = c_2 = 2$ and $w = 0.7$ for all case studies. Both the initial population and total number of iterations are selected as 100.

Optimization performance of each search algorithm is investigated by using three indexes. These indexes are the Best Function Value (BFV), Average Value (AV) and Standard Deviation (STD) of function values that are obtained at the end of 30 runs of these search optimization algorithms. BFV, AV and STD indexes are employed to

Table 5. Multimodal low-dimensional test functions.

Test Function	n	Domain	$f_{\min}$
$f_{14}(x) = \left[\frac{1}{500} + \sum_{j=1}^{25}\frac{1}{j + \sum_{i=1}^{n}(x_i - a_{ij})^6}\right]^{-1}$	2	$[-65.536, 65.536]^n$	1
$f_{15}(x) = \sum_{j=1}^{11}\left[a_i - \frac{x_1(b_i^2 + b_i x_2)}{b_i^2 + b_i x_3 + x_4}\right]^2$	4	$[-5, 5]^n$	0.0003075
$f_{16}(x) = 4x_1^2 - 2.1x_1^4 + \frac{1}{3}x_1^6 + x_1x_2 - 4x_2^2 + 4x_2^4$	2	$[-5, 5]^n$	-1.0316285
$f_{17}(x) = \left(x_2 - \frac{5.1}{4\pi^2}x_1^2 + \frac{5}{\pi}x_1 - 6\right)^2 + 10\left(1 - \frac{1}{8\pi}\right)\cos x_1 + 10$	2	$[-5, 10] \times [0, 15]$	0.398
$f_{18}(x) = [1 + (x_1 + x_2 + 1)^2 \times (19 - 14x_1 + 3x_1^2 - 14x_2 + 6x_1x_2 + 3x_2^2)] \times [30 + (2x_1 - 3x_2)^2 \times (18 - 32x_1 + 12x_1^2 + 48x_2 - 36x_1x_2 + 27x_2^2)]$	2	$[-2, 2]^n$	3
$f_{19}(x) = -\sum_{i=1}^{4} c_i\exp\left[-\sum_{j=1}^{n} a_{ij}(x_j - p_{ij})^2\right]$	4	$[0, 1]^n$	-3.36
$f_{20}(x) = -\sum_{i=1}^{4} c_i\exp\left[-\sum_{j=1}^{n} a_{ij}(x_j - p_{ij})^2\right]$	6	$[0, 1]^n$	-3.32
$f_{21}(x) = -\sum_{i=1}^{5}[(x - a_i)(x - a_i)^T + c_i]^{-1}$	4	$[0, 10]^n$	-10
$f_{22}(x) = -\sum_{i=1}^{7}[(x - a_i)(x - a_i)^T + c_i]^{-1}$	4	$[0, 10]^n$	-10
$f_{23}(x) = -\sum_{i=1}^{10}[(x - a_i)(x - a_i)^T + c_i]^{-1}$	4	$[0, 10]^n$	-10

Table 6. Simulation results of functions f_1 to f_7.

Function	Algorithms	BFV	AVB	STD
f_1	GA	106.2432	173.6455	49.2521
	PSO	30.2765	67.4297	22.7423
	GRSA	**0.0212**	**1.2994**	**4.0989**
f_2	GA	4.2304	5.6516	0.6431
	PSO	4.2304	5.6516	0.6431
	GRSA	3.8855	6.7817	2.4407
f_3	GA	526.4150	1.4641e+003	449.1398
	PSO	495.2562	1.1602e+003	405.8844
	GRSA	**280.1394**	**1.5618e+003**	**1.1841e+003**
f_4	GA	4.8623	7.1303	1.0944
	PSO	5.8939	12.4644	2.9916
	GRSA	**1.2071**	**6.7939**	**3.1609**
f_5	GA	1.7264e+003	4.6402e+003	2.2645e+003
	PSO	480.0151	3.0053e+003	2.6075e+003
	GRSA	**33.7456**	**292.3374**	**447.6176**
f_6	GA	73	173.6000	50.3434
	PSO	57	139.0667	61.8312
	GRSA	**0**	**1**	**1.3131**
f_7	GA	0.0133	0.0344	0.0103
	PSO	0.0318	0.1055	0.0494
	GRSA	**0.0069**	**0.0251**	**0.0248**

evaluate the proposed algorithm in comparison with GA and PSO. BFV, AV and STD are criteria for getting the global optimum, good convergence and robustness, respectively. Simulation results of GA, PSO and GRSA for solving unimodal functions (f_1 to f_7) are listed in Table 6. Table 6 shows the robustness and effectiveness of the proposed GRSA in obtaining the optimal solution versus GA and PSO. Figure 5

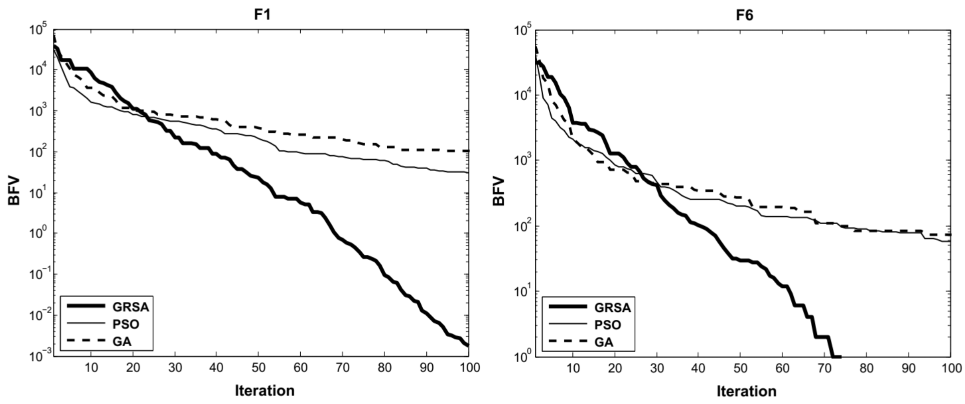


Fig. 5. Convergence curves of GA, PSO and GRSA for test functions f_1 and f_6.

Table 7. Simulation results of functions f_8 to f_{13}.

Function	Algorithms	BFV	AVB	STD
f_8	GA	−1.1121e+004	−9.7111e+003	796.2669
	PSO	−7.8333e+003	−6.3254e+003	673.6822
	GRSA	**−1.2569e+004**	**−1.1998e+004**	**806.3384**
f_9	GA	28.5587	48.0958	11.8329
	PSO	27.9103	43.8791	12.0843
	GRSA	**1.9899**	**10.1609**	**5.2855**
f_{10}	GA	1.5019	2.4706	0.3613
	PSO	3.0150	4.9310	1.2307
	GRSA	**2.0512e−008**	**2.2951e−006**	**3.9401e−006**
f_{11}	GA	1.0550	1.1365	0.0341
	PSO	0.7077	1.0053	0.0914
	GRSA	**1.2212e−015**	**0.0021**	**0.0069**
f_{12}	GA	1.5019	2.4706	0.3613
	PSO	3.0150	4.9310	1.2307
	GRSA	**2.0512e−008**	**2.2951e−006**	**3.9401e−006**
f_{13}	GA	0.7148	1.3284	0.4445
	PSO	6.3180	33.3441	17.4949
	GRSA	**1.2477e−019**	**0.0011**	**0.0034**

shows the convergence curves of test functions f_1 and f_6 for GA, PSO and GRSA. This figure illustrates that the GRSA convergence speed is faster than GA and PSO.

Multimodal high-dimensional functions have many local optimum points and almost are the most difficult functions to optimize. For multimodal functions, the final results are more important because they reflect the ability of search algorithm in escaping from local optima and finding global optimum or reach near it. Functions f_8 to f_{13} are multimodal high-dimensional functions. For these functions, the final results are very important. To optimize multimodal functions, the initial population is selected 100 and total number of iterations is set to 500. For 30 runs of the search optimization algorithms, BFV, AV and STD are obtained. Results are listed in Table 7 for multimodal high-dimensional functions f_9 to f_{13}, respectively. Results show that GRSA is better than GA and PSO, absolutely. Furthermore, convergence curves of test functions f_9 and f_{11} for different search algorithms are depicted in Fig. 6 that shows effectiveness and convergence speed of GRSA. Obtained results verify that GRSA is an effective method in finding the optimal solution of multimodal large dimensional functions.

Functions f_{14} to f_{23} are multimodal low-dimensional functions. For these functions, the convergence rate and final optimal results are both important. Therefore, it is important that search algorithm converges to an optimal value in few numbers of iterations. To optimize functions f_{14} to f_{23}, both the initial population and total iterations are set to 100. For 30 runs of each algorithm, BFV, AV and STD are obtained. Results are given in Table 8. Numbers in Table 8 validate the effectiveness

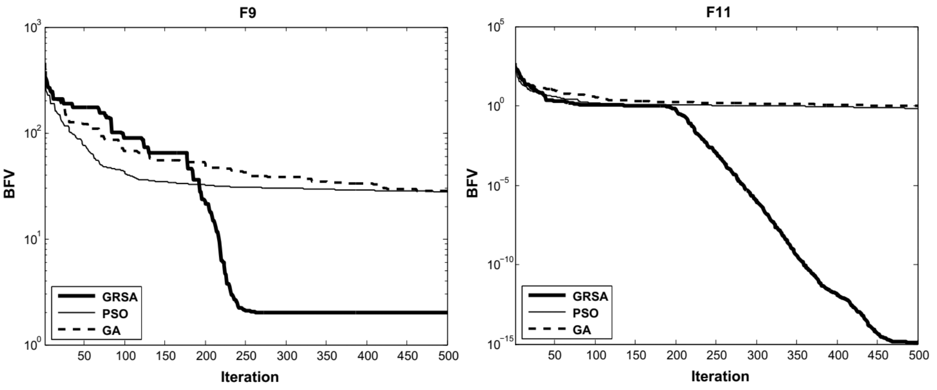
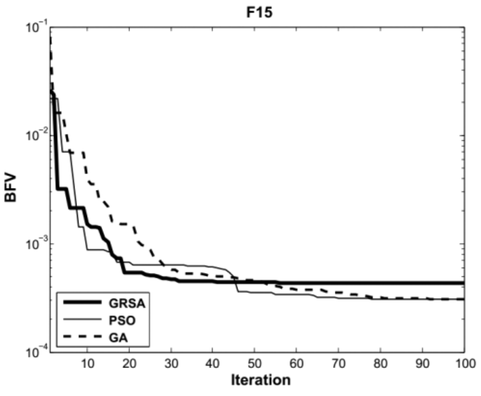


Fig. 6. Convergence curves of GA, PSO and GRSA for test functions f_9 and f_{11}.

Table 8. Simulation results of functions f_{14} to f_{23}.

Function	Algorithms	BFV	AVB	STD
f_{14}	GA	0.9980	1.6308	0.8836
	PSO	0.9980	1.6270	0.7131
	GRSA	**0.9980**	**1.0311**	**0.1815**
f_{15}	GA	3.0870e−004	0.0015	0.0037
	PSO	3.0749e−004	0.0010	0.0037
	GRSA	**4.3015e−004**	**6.7833e−004**	**9.2107e−005**
f_{16}	GA	−1.0316	−1.0316	4.5168e−016
	PSO	−1.0316	−1.0316	6.0135e−012
	GRSA	**−1.0316**	**−1.0316**	**1.3743e−005**
f_{17}	GA	0.3979	0.3979	0
	PSO	0.3979	0.3979	1.7726e−011
	GRSA	**0.3979**	**0.3979**	**8.7887e−013**
f_{18}	GA	3.0000	3.0000	1.6596e−015
	PSO	7.7827	7.7827	2.7101e−015
	GRSA	**3.0000**	**3.0000**	**2.2048e−011**
f_{19}	GA	−3.8628	−3.8628	2.4028e−015
	PSO	−3.8628	−3.8628	1.5858e−011
	GRSA	**−3.8628**	**−3.8628**	**2.7101e−015**
f_{20}	GA	−3.3220	−3.2744	0.0592
	PSO	−3.3220	−3.2784	0.0583
	GRSA	**−3.3220**	**−3.2744**	**0.0592**
f_{21}	GA	−10.1532	−9.4122	2.0785
	PSO	−10.1532	−7.8103	3.2196
	GRSA	**−10.1532**	**−6.7117**	**3.5676**
f_{22}	GA	−10.4029	−9.8071	1.9034
	PSO	−10.4029	−8.5225	3.2171
	GRSA	**−10.4029**	**−7.8142**	**3.4929**
f_{23}	GA	−10.5364	−9.3124	2.7934
	PSO	−10.5364	−9.2892	2.8713
	GRSA	**−10.5364**	**−9.2140**	**3.0086**



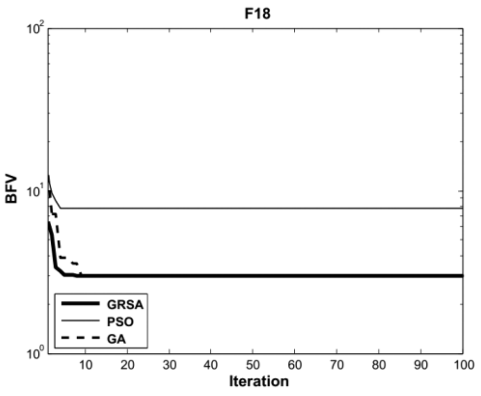


Fig. 7. Convergence curves of GA, PSO and GRSA for test functions f_{15} and f_{15}.

of the proposed GRSA for optimizing these functions. Convergence curves of functions f_{15} and f_{18} for GA, PSO and GRSA are depicted in Fig. 7. This figure shows that GRSA solves multimodal low-dimensional functions effectively. Functions f_1, f_6, f_9, f_{11}, f_{15} and f_{18} are selected examples. Other functions show similar properties. Therefore, their convergence curves are not shown.

GRSA is investigated by using 23 benchmark functions. BFV, AV and STD indexes are employed to compare GRSA versus GA and PSO. Results of functions f_1 to f_7 verify the superior performance of GRSA in solving unimodal large dimensional functions. Furthermore, GRSA is an effective method in finding the optimal solution of multimodal large dimensional functions (f_8 to f_{13}) and provides suitable solutions to multimodal small dimensional functions (f_{14} to f_{23}).

Previous discussion and figures illustrates GRSA performance in solving different optimization problems and also verifies GRSA robustness in finding near global optimal solutions. Based on numerical simulation results and comparison with two existing search algorithms, we can conclude that GRSA can be applied to a vast group of optimization problems. Specially, it can be applied to multimodal large dimensional functions in a large search space for obtaining near global optimal solutions, satisfactorily. In other word, GRSA has good performance in finding near global optimal solutions for nonlinear, complex and multimodal large dimensional functions in a large search space. In the next section GRSA is employed to solve a real-world problem in electrical engineering.

7. Real-World Application of GRSA in Electrical Engineering (Optimal PSS Design)

To increase the damping of power system oscillations and improve system oscillation stability, the installation of supplementary excitation control, PSS, is a simple, effective and economical method. In fact, PSSs are one of the main components of power systems and have been widely used to damp low frequency oscillation and enhance power system stability.

Stabilizer design based on the concepts of classical control theory has been widely used in actual power systems. The stabilizer's parameters are mostly tuned by using appropriate mathematical models. The main drawback of these controllers is their inherent lack of robustness.[58]

Various MHAs such as GA, PSO, TS, COA, SPEA and others have been widely applied to the problem of multi-machine PSSs design.[58–62] These methods show a good performance for the solution of the optimal PSS design problem. However, when the objective function of the optimization problem is multimodal and has a large dimension, then these have missed effectiveness to obtain the global optimum solution and trapped in local optimal points. To overcome these drawbacks, GRSA is used to solve optimal PSS design problem.

7.1. *Power system model*

The Differential Algebraic Equations (DAEs) describe dynamic model of a power system.[63] The differential equations of an m-machine power system representing the synchronous machine and the Automatic Voltage Regulator (AVR), can be described as follows:

$$T'_{d0i}\frac{dE'_{qi}}{dt} = -E'_{qi} - (X_{di} - X'_{di})I_{di} + E_{fdi}, \tag{28}$$

$$T'_{q0i}\frac{dE'_{di}}{dt} = -E'_{di} - (X_{qi} - X'_{qi})I_{qi}, \tag{29}$$

$$\frac{d\delta_i}{dt} = \omega_i - \omega_s, \tag{30}$$

$$\frac{2H_i}{\omega_s}\frac{d\omega_i}{dt} = T_{Mi} - E'_{di}I_{di} - E'_{qi}I_{qi} - (X'_{qi} - X'_{di})I_{di}I_{qi} - D_i(\omega_i - \omega_s), \tag{31}$$

$$T_{Ai}\frac{dE_{fdi}}{dt} = -K_{Ai}E_{fdi} + K_{Ai}(V_{refi} - V_i), \tag{32}$$

where δ is the rotor angle, ω is the rotor speed, ω_s is the synchronous speed, E'_d and E'_q are internal transient voltage behind x'_d and x'_q, I_d and I_q are d-axis and q-axis components of generator stator currents, H is the generator inertia constant, D is the damping coefficient, T_M is the mechanical power input, x_d is the d-axis reactance, x_q is the q-axis reactance, x'_d is the d-axis transient reactance, x'_q is the q-axis transient reactance, T'_{d0} is the open circuit d-axis time constant, T'_{q0} is the open circuit q-axis time constant, E_{fd} is the excitation system voltage, V is the generator terminal voltage, V_{ref} is the regulator reference voltage, K_A is the regulator gain, and T_A is the regulator time constant. In Eqs. (28) to (32), subscript i denotes ith synchronous generator.

The input mechanical torque T_{Mi} is treated as a constant in the excitation controller design, i.e., it is assumed that the governor action is slow enough not to have any significant impact on the machine dynamics. The electrical torque is as follows:

$$T_{Ei} = E'_{di} I_{di} + E'_{qi} I_{qi} + (X'_{qi} - X'_{di}) I_{di} I_{qi}. \tag{33}$$

The electrical torque is used in Eq. (31). In a power system with n buses, m synchronous generators and $m - n$ load buses, the algebraic constraints of the power system are:

$$\begin{aligned} 0 = {} & V_i e^{j\theta_i} + (R_{si} + jX'_{di})(I_{di} + jI_{qi}) e^{j(\delta_i - \frac{\pi}{2})} \\ & - [E'_{di} + (X'_{qi} - X'_{di}) I_{qi} + jE'_{qi}] e^{j(\delta_i - \frac{\pi}{2})} \quad i = 1, \ldots, m \end{aligned} \tag{34}$$

$$\begin{aligned} & V_i e^{j\theta_i} (I_{di} - jI_{qi}) + P_{Li}(V_i) + jQ_{Li}(V_i) \\ & \quad = \sum_{k=1}^{n} V_i V_k Y_{ik} e^{j(\theta_i - \theta_k - \alpha_{ik})}, \quad i = 1, \ldots, m \end{aligned} \tag{35}$$

$$P_{Li}(V_i) + jQ_{Li}(V_i) = \sum_{k=1}^{n} V_i V_k Y_{ik} e^{j(\theta_i - \theta_k - \alpha_{ik})}, \quad i = m+1, \ldots, n \tag{36}$$

where P_L and Q_L are active and reactive powers of load, respectively. $Y\mathrm{e}^{j\alpha}$ is the admittance matrix and θ is the angle of bus voltage V. In a power system with m generators, the loads can be represented by constant impedances. Therefore, order reduction can be used to eliminate the load related elements in the admittance matrix of the lines network. The linear model of the system can be written as:

$$\Delta\dot{\mathbf{x}} = \mathbf{Ax} + \mathbf{Bu}, \tag{37}$$

where $\mathbf{x}$ is the vector of state variables, $\mathbf{A}$ is the system state matrix, $\mathbf{B}$ is input matrix and $\mathbf{u}$ is the control input vector.

7.2. *Design criteria*

This paper considers a widely used conventional lead-lag PSS and the IEEEtype-ST1 excitation system that are shown in Fig. 8. The conventional lead-lag PSS can be described as:

$$G_i(s) = \frac{V_{\mathrm{PSS}i}(s)}{\Delta\omega_i(s)} = K_{Gi} \frac{T_w s}{(1 + sT_w)} \frac{(1 + sT_{1i})(1 + sT_{3i})}{(1 + sT_{2i})(1 + sT_{4i})}, \tag{38}$$

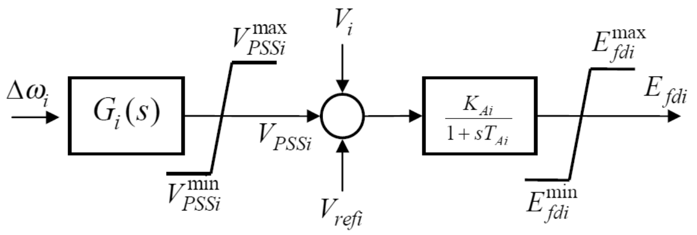


Fig. 8. Conventional lead-lag PSS is connected to IEEE-type-ST1 excitation system.

where T_w is the washout time constant, $V_{\text{PSS}i}$ is the PSS output signal at the ith machine, and $\Delta\omega_i$ is the speed deviation of this machine. The time constant T_w is equal to 10 in this study. The optimal values of the stabilizer gain K_{Gi} and the time constants T_{1i}, T_{2i}, T_{3i} and T_{4i} are to be determined.

In order to increase damping of electromechanical modes (EMs) and to determine the stabilizer gain and the time constants, an eigenvalue-based objective function is defined as follows:

Minimize J

$$\begin{aligned} &J = \max\{\text{real}(\lambda_i) \mid \lambda_i \in \text{EMs}\} + K_P \sum \{\text{real}(\lambda_j) \mid \lambda_j > 0\} \\ &\text{EMs} = \left\{ \lambda_k \middle| 0 < \frac{im(\lambda_k)}{2\pi} < 5 \right\} \end{aligned} . \tag{39}$$

Subject to:

$$\begin{aligned} &0.001 \le K_{Gi} \le 50, \\ &0.001 \le T_{1i} \le 1, \\ &0.02 \le T_{2i} \le 1, \\ &0.001 \le T_{3i} \le 1, \\ &0.02 \le T_{4i} \le 1, \end{aligned} \tag{40}$$

where λ_i is ith eigenvalue of the system state matrix, $\mathbf{A}$, and K_P is a penalty factor which penalizes the modes with unstable eigenvalue. In this paper $K_P = 50$.

7.3. *PSS design results*

In this paper, a widely used Western Systems Coordinating Council (WSCC) 3-machine, 9-bus power system is considered.[63] Single-line diagram of the WSCC power system is shown in Fig. 9. The G1 is the slack bus generator and its rotor angle

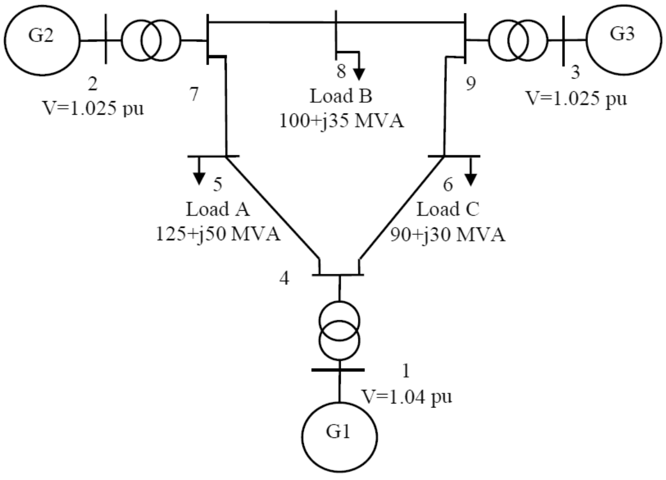


Fig. 9. Single line diagram of WSCC 3-machine power system.

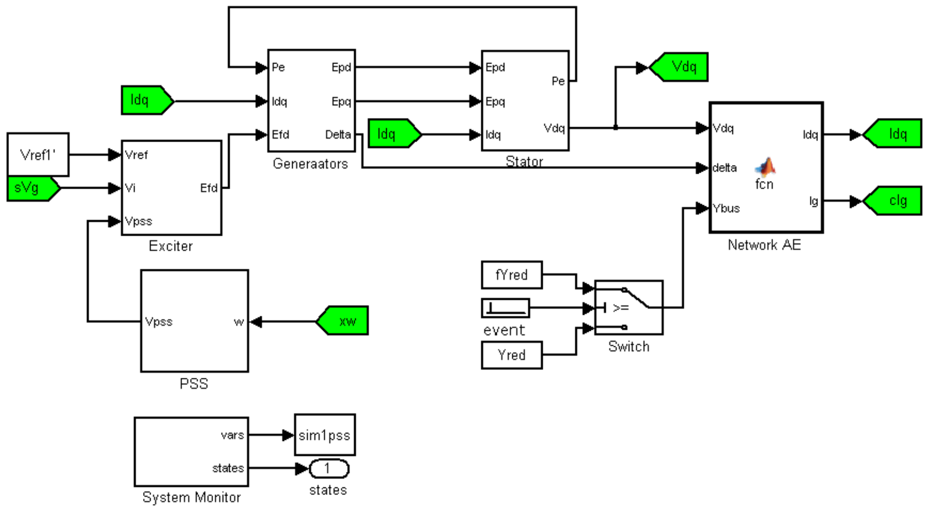


Fig. 10. Implementation of the multi-machine power system in MATLAB/Simulink.

(δ_1) is selected as the reference angle. The generators are represented as fourth-order models. The fourth-order model DAEs are described in Sec. 7.1. All dynamic parameters of the WSCC power system are given in the Ref. 63.

A power flow program[64] is used to calculate dynamic initial conditions. The DAEs (Eqs. (28) to (36)) solution determines the nonlinear dynamic behavior of the power system. The DAEs can be solved using an Ordinary Differential Equation (ODE) solver. This paper uses MATLAB/Simulink to solve the DAEs using ODEs. MATLAB/Simulink implementation of the WSCC power system is shown in Fig. 10.

By calculating the system state matrix, **A**, the proposed objective function can be formed. The objective function is evaluated using 20 successive runs of GA, PSO and GRSA. The statistical indices are summarized in Table 9. The optimal parameters of designed PSSs are listed in Table 10. Table 9 shows the superiority of GRSA to GA and PSO. It can be seen from Table 9 that the BFV, AVB and STD of GRSA are generally smaller than those of GA and PSO. Furthermore, the convergence curves of GA, PSO and GRSA versus iterations are depicted in Fig. 11. According to Fig. 11, the upper left figure compares GRSA versus GA and PSO and validates the effectiveness of the GRSA. The upper right and lower left and lower right figures of

Table 9. Statistical indices for optimal PSS design using GA, PSO and GRSA.

Parameter	GA	PSO	GRSA
BFV	−3.8633	−3.8347	**−3.8765**
AVB	−2.7428	−2.1387	**−3.6342**
STD	0.7782	1.1926	**0.3054**

Table 10. Optimal PSS parameters.

	Algorithm	K_G	T_1	T_2	T_3	T_4
G2	GA	3.9669	0.6445	0.0279	0.7465	0.0208
	PSO	4.0700	0.6849	0.0287	0.6848	0.0200
	GRSA	**4.5140**	**0.6581**	**0.0211**	**0.6737**	**0.0230**
G3	GA	1.3938	0.6189	0.0200	0.8567	0.0224
	PSO	0.8952	0.8409	0.0200	0.9367	0.0200
	GRSA	**0.8845**	**0.7403**	**0.0299**	**0.9705**	**0.0207**

Fig. 11 show the convergence curves of each algorithm separately. The global search ability of the algorithms can be measured by calculating the numbers of curves that converge to near global optimal point and divide them by total number of successive runs. From Fig. 11 GRSA has 16/20 successful runs, PSO has 2/20 successful runs and PSO has 2/20 successful runs. This shows the superior performance of GRSA than GA and PSO in approaching the global optimal point.

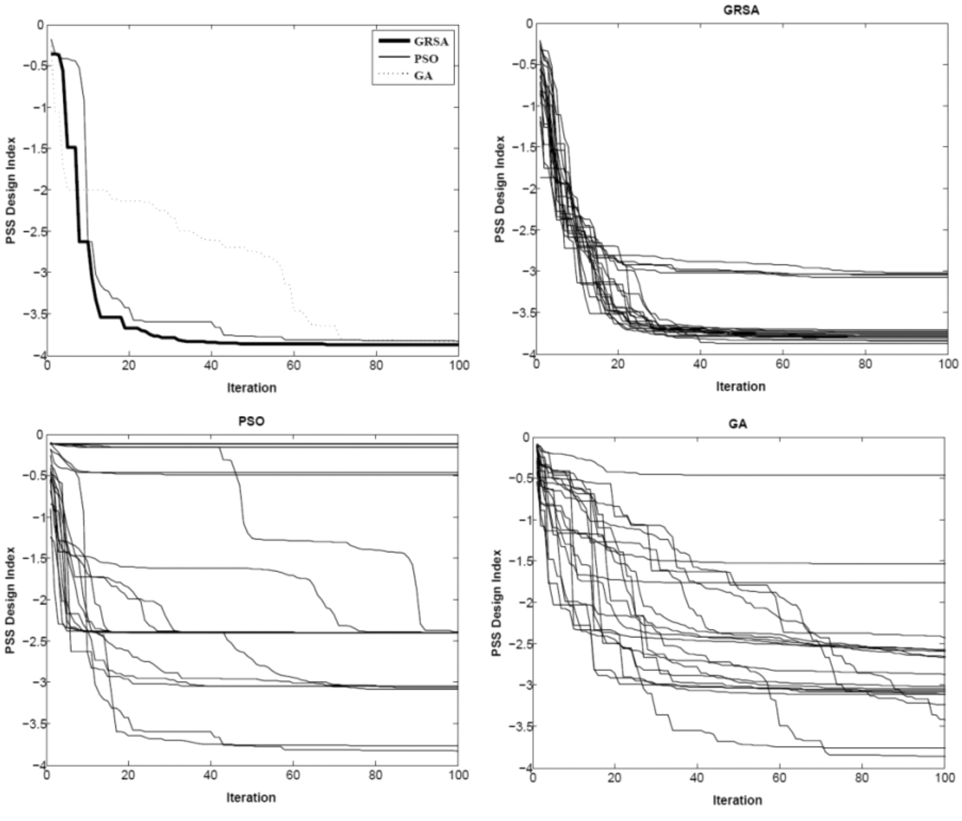


Fig. 11. Convergence curves of GA, PSO and GRSA in finding optimal design of PSSs.

Table 11. The power system eigenvalues.

Mode	Base	GA	PSO	GRSA
1	$\mathbf{-0.6856 \pm 12.7756i}$	$\mathbf{-5.1084 \pm 14.2570i}$	$\mathbf{-5.1839 \pm 13.9827i}$	$\mathbf{-4.6738 \pm 13.7207i}$
2	$\mathbf{-0.1229 \pm 8.2867i}$	$\mathbf{-3.9943 \pm 8.0030i}$	$\mathbf{-3.8455 \pm 8.0922i}$	$\mathbf{-4.4736 \pm 7.8868i}$
3	$-2.3791 \pm 2.6172i$	$-3.9133 \pm 2.7461i$	$-3.8278 \pm 2.6126i$	$-3.8860 \pm 2.9425i$
4	$-4.6706 \pm 1.3750i$	$-3.8633 \pm 2.4751i$	$-3.8468 \pm 2.4741i$	$-3.8765 \pm 2.2625i$
5	$-3.5199 \pm 1.0156i$	$-3.8630 \pm 2.0062i$	$-3.9850 \pm 2.0816i$	$-3.8766 \pm 2.0643i$

The designed PSSs performance in the power system is evaluated using eigenvalue analysis and time-domain simulations. The example power system eigenvalues are obtained and listed in Table 11 and their related damping ratio and frequency are listed in Table 12. Modes 1 and 2 are EMs due to their low damping ratios. After optimization, these modes are drastically changed from $-0.6856 \pm 12.7756i$ and $-0.1229 \pm 8.2867i$ up to $-4.6738 \pm 13.7207i$ and $-4.4736 \pm 7.8868i$, respectively. Damping ratio of the worst EM is enhanced from 0.0148 up to 0.4934 for GRSA. Tables 11 and 12 show that the GRSA-based PSSs will effectively control the power system low frequency oscillations due to EMs.

To demonstrate the effectiveness and the robustness of the GRSA-based PSSs the example power system is subjected to various severe disturbances as well as small disturbance. The following disturbances are considered and time-domain simulations are performed:

- A 100 ms symmetrical three-phase fault at bus 9 at the end of lines 8 and 9 at $t = 1$ s.
- 50% load change (decrement) at bus 5 at $t = 1$ s with duration of 100 ms.
- 0.05 pu increment of V_{ref} of the generator 3 at $t = 1$ s with duration of 100 ms.

After the fault clearance, the original system will be restored. Variation of the generator speed represents the stability status of the power system. Therefore, the speed deviations ω_2–ω_1 and ω_3–ω_1 are depicted graphically in Fig. 12. Time-domain simulations of Fig. 12 show that the designed PSSs using GRSA effectively damp the oscillations considering severe and small disturbances. Therefore, GRSA can be used as general tool for optimal design of PSSs as well as similar engineering problems.

Table 12. Damping ratio and frequency of eigenvalues.

Base		GA		PSO		GRSA	
Damping Ratio	Frequency	Damping Ratio	Frequency	Damping Ratio	Frequency	Damping Ratio	Frequency
0.0536	**2.0333**	**0.3373**	**2.2691**	**0.3476**	**2.2254**	**0.3224**	**2.1837**
0.0148	**1.3189**	**0.4466**	**1.2737**	**0.4292**	**1.2879**	**0.4934**	**1.2552**
0.6726	0.4165	0.8186	0.4371	0.8260	0.4158	0.7972	0.4683
0.9593	0.2188	0.8420	0.3939	0.8411	0.3938	0.8637	0.3601
0.9608	0.1616	0.8875	0.3193	0.8864	0.3313	0.8827	0.3285

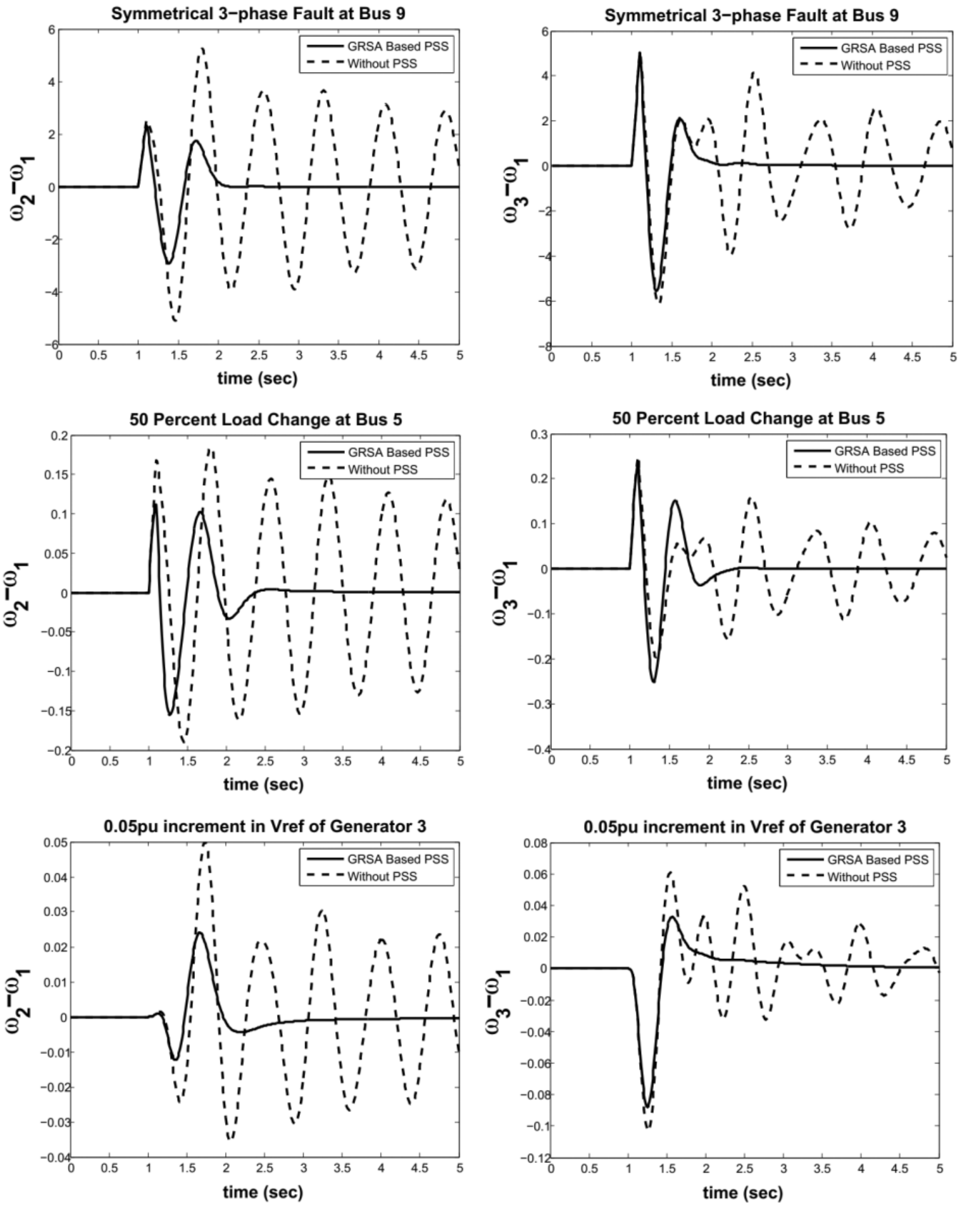


Fig. 12. Time-domain simulations of the WSCC 3-machine power system.

8. Conclusion

In this paper, a metaheuristic optimization algorithm inspired by GRT was introduced. We named the proposed algorithm GRSA. GRSA starts with a random initial population of particles that are considered as a tensor. Updating rule in GRSA consists of two terms: step length and step direction. Step length was calculated by using velocity of the particles. Spacetime geometry properties and particle's energy–momentum were used to calculate velocity of the particles. Step direction of particles was obtained by geodesic tangent vector in curved spacetime geometry. Particles have been moved to a position with minimum action. In GRSA, particles are solution agents for the optimization problem. Particles actions are determined by values of objective function. Therefore, step length and step direction for solution agents of optimization problem can be obtained and the solution agents move toward optimum point.

Numerical simulation results show that GRSA has major advantages in solving unimodal/multimodal large dimensional optimization problems in comparison with GA and PSO. Statistically, GRSA shows good performance in finding near global optimal solutions than GA and PSO. In addition, GRSA is applied to optimal PSS design as a real-world practical application. Eigenvalue analysis and time-domain simulation of the PSSs in the example power system validate the effectiveness of GRSA in solving real-world applications.

References

1. G. R. Walsh, *Methods of Optimization* (John Wiley & Sons Ltd, 1975).
2. B. Zhao, C. X. Guo and Y. J. Cao, A multiagent-based particle swarm optimization approach for optimal reactive power dispatch, *IEEE Trans. Power Syst.* **20** (2005) 1070–1078.
3. J. H. Holland, *Adaptation in Natural and Artificial Systems: An Introductory Analysis with Applications to Biology, Control, and Artificial Intelligence* (U Michigan Press, 1975).
4. M. Dorigo, V. Maniezzo and A. Colorni, Ant system: Optimization by a colony of cooperating agents, *IEEE Trans. Syst., Man, Cybern. B Cybern.* **26** (1996) 29–41.
5. J. Kennedy and R. Eberhart, Particle swarm optimization, in *Proc. IEEE Int. Conf. Neural Networks* (1995), pp. 1942–1948.
6. K. M. Passino, Biomimicry of bacterial foraging for distributed optimization and control, *IEEE Control Syst. Mag.* **22**(3) (2002) 52–67.
7. D. Karaboga and B. Basturk, A powerful and efficient algorithm for numerical function optimization: Artificial bee colony (ABC) algorithm, *J. Global Optim.* **39** (2007) 459–471.
8. R. Storn and K. Price, Differential evolution–a simple and efficient heuristic for global optimization over continuous spaces, *J. Global Optim.* **11** (1997) 341–359.
9. C. Dai, Y. Zhu and W. Chen, Seeker optimization algorithm, *Comput. Intell. Secur.* **4456** (2007) 167–176.
10. E. Rashedi, H. Nezamabadi-pour and S. Saryazdi, GSA: A gravitational search algorithm, *Inf. Sci.* **179** (2009) 2232–2248.
11. N. Hansen and A. Ostermeier, Completely derandomized self-adaptation in evolution strategies, *Evol. Comput.* **9** (2001) 159–195.
12. Y. D. Valle, G. K. Venayagamoorthy, S. Mohagheghi, J. C. Hernandez and R. G. Harley, Particle swarm optimization: Basic concepts, variants and applications in power systems, *IEEE Trans. Evol. Comput.* **12.2** (2008) 171–195.
13. D. H. Wolpert and W. G. Macready, No free lunch theorems for optimization, *IEEE Trans. Evol. Comput.* **1** (1997) 67–82.

14. J. R. Koza, *Genetic Programming: On the Programming of Computers by Means of Natural Selection*, Vol. 1 (MIT press, 1992).
15. I. Rechenberg, Evolution strategy, *Comput. Intel. Imitat. Life* **1** (1994).
16. X. Yao, Y. Liu and G. Lin, Evolutionary programming made faster, *IEEE Trans. Evol. Comput.* **3** (1999) 82–102.
17. H. A. Abbass, MBO: Marriage in honey bees optimization — A haplometrosis polygynous swarming approach, in *Proc. 2001 Congr. Evolutionary Computation* (2001), pp. 207–214.
18. L. N. De Castro and F. J. Von Zuben, Learning and optimization using the clonal selection principle, *IEEE Trans. Evol. Comput.* **6**(3) (2002) 239–251.
19. S. I. Birbil and S. C. Fang, An electromagnetism-like mechanism for global optimization, *J. Global Optim.* **25.3** (2003) 263–282.
20. B. Webster and P. J. Bernhard, A local search optimization algorithm based on natural principles of gravitation, in *Proc. 2003 Int. Conf. Information and Knowledge Engineering*, Las Vegas, Nevada, USA (2003), pp. 255–261.
21. X. Li, A new intelligent optimization-artificial fish swarm algorithm, Ph.D. thesis, Zhejiang University of Zhejiang, China (2003).
22. W. F. Sacco and R. E. de Oliveira Cassiano, A new stochastic optimization algorithm based on a particle collisions, *Transactions of the American Nuclear Society* **92** (2005) 657–659.
23. R. Martin and W. Stephen, Termite: A swarm intelligent routing algorithm for mobile-wireless Ad-Hoc networks, in *Stigmergic Optimization* (Springer Berlin Heidelberg, 2006), pp. 155–184.
24. K. S. Lee and Z. W. Geem, A new meta-heuristic algorithm for continues engineering optimization: Harmony search theory and practice, *Comput. Methods Appl. Mech. Eng.* **194** (2005) 3902–3933.
25. B. Alatas and E. Akın, An efficient genetic algorithm for automated mining of both positive and negative quantitative association rules, *Soft Comput.* **10.3** (2006) 230–237.
26. A. R. Mehrabian and C. Lucas, A novel numerical optimization algorithm inspired from weed colonization, *Ecol. Informatics* **1**(4) (2006) 355–366.
27. D. Niizuma, K. Yasuda and A. Ishigame, Multi-point Tabu search for traveling salesman problems, *IEEE Trans. Electr. Electron. Eng.* **1**(1) (2006) 126–129.
28. S. C. Chu, P. W. Tsai and J. S. Pan, Cat swarm optimization, in *PRICAI 2006: Trends in Artificial Intelligence* (Springer Berlin Heidelberg, 2006), pp. 854–858.
29. O. K. Erol and I. Eksin, A new optimization method: big bang–big crunch, *Adv Eng. Softw.* **37** (2006) 106–111.
30. H. Du, X. Wu and J. Zhuang, Small-world optimization algorithm for function optimization, in *Advances in Natural Computation* (Springer, 2006), pp. 264–273.
31. L. Lamberti and C. Pappalettere, Weight optimization of skeletal structures with multi-point simulated annealing, *Comput. Model. Eng. Sci.* **18** (2007) 183–221.
32. A. Karci, Theory of saplings growing up algorithm, in *Adaptive and Natural Computing Algorithms* (Springer Berlin Heidelberg, 2007), pp. 450–460.
33. R. A. Formato, Central force optimization: A new metaheuristic with applications in applied electromagnetics, *Prog. Electromag. Res.* **77** (2007) 425–491.
34. E. A. Gargari and C. Lucas, Imperialist competitive algorithm: An algorithm for optimization inspired by imperialistic competition, *IEEE Congr. Evolutionary Computation*, Singapore (2007), pp. 4661–4667.
35. P. C. Pinto, T. A. Runkler and J. M. Sousa, Wasp swarm algorithm for dynamic MAX-SAT problems, in *Adaptive and Natural Computing Algorithms* (Springer, 2007), pp. 350–357.

36. A. Mucherino and O. Seref, Monkey search: A novel metaheuristic search for global optimization, in *AIP Conf. Proc.* (2007), p. 162.
37. X. S. Yang, *Nature-Inspired Metaheuristic Algorithms* (Luniver Press, 2008), p. 128.
38. X. Lu and Y. Zhou, A novel global convergence algorithm: Bee collecting pollen algorithm, in *Advanced intelligent computing theories and applications. With Aspects of Artificial Intelligence* (Springer, 2008), pp. 518–525.
39. D. Simon, Biogeography-based optimization, *IEEE Trans. Evol. Comput.* **12** (2008) 702–813.
40. X. S. Yang and S. Deb, Cuckoo search via Lévy flights, *World Congress Nature & Biologically Inspired Computing, 2009, NaBIC 2009* (2009), pp. 210–214.
41. Y. Shiqin, J. Jianjun and Y. Guangxing, A dolphin partner optimization, *WRI Global Congress Intelligent Systems, 2009, GCIS 2009* (2009), pp. 124–128.
42. X. S. Yang, A new metaheuristic bat-inspired algorithm, in *Nature Inspired Cooperative Strategies for Optimization (NICSO 2010)* (Springer, 2010), pp. 65–74.
43. A. Kaveh and S. Talatahari, A novel heuristic optimization method: charged system search, *Acta Mech.* **213** (2010) 267–289.
44. X. S. Yang, Firefly algorithm, stochastic test functions and design optimization, *Int. J. Bio-Inspired Comput.* **2** (2010) 78–84.
45. H. Shah-Hosseini, Principal components analysis by the galaxy-based search algorithm: A novel metaheuristic for continuous optimization, *Int. J. Comput. Sci. Eng.* **6** (2011) 132–140.
46. R. Rajabioun, Cuckoo optimization algorithm, *Appl. Soft Comput.* **11** (2011) 5508–5518.
47. B. Alatas, ACROA: Artificial chemical reaction optimization algorithm for global optimization, *Expert Syst. Appl.* **38** (2011) 13170–13180.
48. A. H. Gandomi and A. H. Alavi, Krill Herd: A new bio-inspired optimization algorithm, *Commun. Nonlinear. Sci. Numer. Simulat.* **17**(12) (2012) 4831–4845.
49. F. F. Moghaddam, R. F. Moghaddam and M. Cheriet, Curved space optimization: A random search based on general relativity theory, preprint (2012), arXiv:1208.2214, pp. 1208–2214.
50. W. T. Pan, A new fruit fly optimization algorithm: Taking the financial distress model as an example, *Knowl-Based Syst.* **26** (2012) 69–74.
51. A. Kaveh and M. Khayatazad, A new meta-heuristic method: ray optimization, *Comput. Struct.* **112** (2012) 283–294.
52. A. Askarzadeh and A. Rezazadeh, A new heuristic optimization algorithm for modeling of proton exchange membrane fuel cell: Bird mating optimizer, *Int. J Energy Res.* **37** (2012) 1196–1204.
53. A. Hatamlou, Black hole: A new heuristic optimization approach for data clustering, *Inf. Sci.* **222** (2013) 175–184.
54. S. A. Mirjalili, S. M. Mirjalili and A. Lewis, Grey wolf optimizer, *Adv. Eng. Softw.* **69** (2014) 46–61.
55. A. S. Eddington, *Space, Time and Gravitation* (Cambridge University Press, 1987).
56. J. Stewart, *Advanced General Relativity* (Cambridge University Press, 1994).
57. P. A. M. Dirac, *General Theory of Relativity* (Wiley Interscience, 1975).
58. A. Ghasemi, H. Shayeghi and H. Alkhatib, Robust design of multimachine power system stabilizers using fuzzy gravitational search algorithm, *Electr. Power Energy Syst.* **51** (2013) 190–200.
59. Y. L. Abdel-Magid and M. A. Abido, Optimal multiobjective design of robust power system stabilizers using genetic algorithms, *IEEE Trans. Power Syst.* **18**(3) (2003) 1125–1132.

60. H. Shayeghi, H. A. Shayanfar, S. Jalilzadeh and A. Safari, Multi-machine power system stabilizers design using chaotic optimization algorithm, *Energy Convers. Manag.* **51** (2010) 1572–1580.
61. S. Mishra, M. Tripathy and J. Nanda, Multi-machine power system stabilizer design by rule based bacteria foraging, *Electr. Power Syst. Res.* **77** (2007) 1595–1607.
62. H. Shayeghi, H. A. Shayanfar, A. Safari and R. Aghmasheh, A robust PSSs design using PSO in a multi-machine environment, *Energy Convers. Manag.* **51** (2010) 696–702.
63. P. W. Sauer and M. A. Pai, *Power System Dynamics and Stability* (Prentice Hall, 1998).
64. R. D. Zimmerman, C. E. Murillo-Sánchez and R. J. Thomas, MATPOWER: Steady-state operations, planning, and analysis tools for power systems research and education, *IEEE Trans. Power Syst.* **26**(1) (2011) 12–19.