

# A GLOBAL OPTIMIZATION ALGORITHM FOR UNCONSTRAINED OPTIMIZATION

ROKHSAREH ZAVVAR<sup>1\*</sup>, SAEED NEZHADHOSEIN<sup>2</sup>, AND AGHILEH HEIDARI<sup>3</sup>

<sup>1,2,3</sup> Department of Applied Mathematics, Faculty of Mathematics, Payam Noor University, P.O. Box 193953697, Tehran, Iran.

ABSTRACT. Here, using a local conjugate gradient (CG) method and random perturbations, a global optimization algorithm suggested for unconstrained optimization (UO) problems. For conjugate parameter in CG, the Polyak-Ribiere parameter and for random perturbations the Gaussian perturbation centred with zero mean and covariance  $\delta_k^2 I_n$ , in which  $\delta_k$  is decreasing slowly enough, are used. We give some numerical experiments on a set of UO problems.

#### 1. INTRODUCTION

Unconstrained optimization (UO) problems arise in a wide range of applications, for example see [1, 2]. The general form of UO problems is as following:

$$\min_{x \in \mathbb{R}^n} f(x) \tag{1.1}$$

where  $f : \mathbb{R}^n \to \mathbb{R}$  is a smooth non-linear function and its gradient is available. The iterative algorithms for solving the UO problems construct a sequence of solutions as  $\{x_k\}$ , with an initial point  $x_0 \in \mathbb{R}^n$ , by following recursive formula [3]:

$$x_{k+1} = x_k + s_k$$
,  $k = 0, 1, 2, ...$  (1.2)

where  $s_k$  is called step at *k*-th iteration. There are two general approaches for computing the steps in (1.1), including line search (LS) and trust region (TR), which both of them use quadratic approximations of the objective function. In TR approach, at each iterate, the step is computed by solving a sequence of sub-problems of quadratic constrained model, iteratively, [3, 4]. For LS approach the step at the *k*-th iteration is computed as  $s_k = \alpha_k d_k$ , where  $d_k$  is the search direction and  $\alpha_k$  is the step length along this direction. CG methods are the most well known methods in LS for low memory and less computations.

<sup>2010</sup> Mathematics Subject Classification. Primary 47A55; Secondary 39B52, 34K20, 39B82.

Key words and phrases. Global optimization, Conjugate gradient, Gaussian pertubation.

<sup>\*</sup> Speaker.

#### R. ZAVVAR ET AL.

Global optimal solution of the UO problems is important in many applications, see [6]. It demands powerful solvers in terms of time and memory, which named global optimization (GO) algorithms. There are two categories for the solution of GO problems, consist of stochastic and deterministic algorithms. The first is based on heuristic procedures such as genetic algorithm (GA), simulated annealing (SA) particle swarm optimization (PSO), see [7]. These techniques usually are simple but they need very large number of evaluation points of objective function and so are slow specially for the large scale UO problems. However, deterministic algorithms such as branch and bound method have some disadvantages such as computational difficulties as the dimension of the problem increased.

Here we proposed a global algorithm for UO problem, (1.1), based on CG method as a local deterministic algorithms and a perturbation technique, inspired by [8], which used to escape from the local solutions. In CG methods, a sequence of points  $\{x_k\}_{k\geq 0} \subset \mathbb{R}^n$  is generated, beginning from an initial guess  $x_0 \in \mathbb{R}^n$ , which a new feasible point  $x_{k+1}$  is generate form  $x_k$  by the iterative formula (1.2), where the search direction is as following:

$$d_0 = -g_0, \quad d_{k+1} = -g_{k+1} + \beta_k d_k, \quad k = 0, 1, 2, \dots$$
 (1.3)

where  $g_k = \nabla f(x_k)$  and  $\beta_k$  is a scaler called the CG update parameter. To achieve global solution, the solution of the CG method should be perturbed to avoid premature convergence to local minimum. Therefore, the last point, obtained by CG method, is perturbed by adding a suitable chosen stochastic term. The new sequence noted  $\{X_k\}_{k\geq 0} \subset \mathbb{R}^n$  is given by the following recursive formula [5]:

$$\begin{cases} X_0 = x_0 \\ X_{k+1} \in argmin\{f(y_k^i), i = 0, 1, \dots, m\} \end{cases}$$
(1.4)

That

$$y_{k}^{i} = \begin{cases} G(X_{k}), & i = 0\\ G(X_{k}) + \rho_{k}^{i}, & i = 1, 2, \dots, m \end{cases}$$
(1.5)

Where  $G(X_k)$  is the point obtained by a few iterations of the procedure (1.2)-(1.3), starting from  $X_k$ , and  $\rho_k^i$  for i = 0, 1, ..., m are the stochastic perturbations, where *m* is as integer parameter. Here, based on special random variables for  $\rho_k^i$  in (1.5) and Polyak-Ribiere CG parameter, we propose a new GO algorithm for UO problems.

# 2. A GLOBAL OPTIMIZATION ALGORITHM

In this section we propose a new GO algorithm based the procedure in (1.4)-(1.5). The idea is to perturb the solution obtained by the procedure G(.) in (1.5), which is obtained by Polyak-Ribiere method with the CG parameter as  $\beta_k^{PR} = \frac{g_k^T y_{k-1}}{\|g_{k-1}\|^2}$ , where  $y_{k-1} = g_k - g_{k-1}$ . Also, in CG method, we use the step length with a procedure named fixed step-length or without line search proposed by Sun and Zhang [9], where

$$\alpha_k = \frac{g_k^T d_k}{L_k \|d_k\|^2} \tag{2.1}$$

2

where  $L_k$  is a random parameter based on Lipschitz condition, which denoted by L in the following, see [5]. For perturbation, the normal Gaussian distribution, with zero mean and covariance  $\delta_k^2 I_n$ , is used, i.e the random vectors are chosen as  $\rho_k^i \sim \mathcal{N}(0, \delta_k^2 I_n)$  in (1.5), where  $\delta_k$  are decreasing parameters. Each iterate of the proposed algorithm consist of two main phase: Local phase and Perturbation phase. Local phase, with at most  $J_{max}$  iteration, used to achieve an approximate local solution and the perturbation phase, using *m* perturbation vectors, applied to escape from local point and search different area of the feasible domain for finding the global optimal solution. Also the maximum of the iteration in algorithm defined by the  $K_{max}$ , as a parameter. The local phase can break after at most  $J_{max}$  iterate or when we achieve to a appropriate solution.

Here, we present the proposed algorithm, which is named GPR, as follows: In GPR algorithm,

## Algorithm 1 GPR algorithm

{Step 0:}(Initialization) Input the values  $X_0 = x_0$ ,  $K_{max}$ ,  $J_{max}$ , m, L,  $\varepsilon$  and the sequence  $\{\delta_k\}_{k\geq 0}$ . Also let k = 0. {Step 1:} Let j = 0 and  $X_k^0 = x_k$ . {Step 2:}(Local phase) For j = 0 until  $K_{max}$  repeat: if  $\|\nabla f(X_k^j)\| \leq \varepsilon$  break, else Calculate  $d_k^j$ , using (1.3), and  $\alpha_k^j$ , using (2.1), and let  $X_k^{j+1} = X_k^j + \alpha_k^j d_k^j$ . {Step 3:}(Perturbation phase) Generate m perturbation vectors  $p_k^i$ , i = 1, 2, ..., m by common law  $\mathcal{N}(0, \delta_k^2 I_n)$ . {Step 4:} Let  $Y_k = X_k^j$  and  $S_k = \{Y_k\} \cup \{Y_k + p_k^i, i = 1, 2, ..., m\}$ . {Step 5:} Select  $X_{k+1} \in argmin_{y \in S_k} f(y)$ . {Step 6:} Set k = k + 1. If  $k = K_{max}$ , stop and let  $x^* = X_k$ , else go to step 1.

Algorithm 1,  $\alpha_k^j$ ,  $d_k^j$  and  $X_k^j$  are the step length, search direction and the solution of the *j*-th iterate of the local phase in *k*-th iterate of the main loop.

## 3. NUMERICAL EXPERIMENTS

In this section, we report the numerical results of ten test functions with our proposed GPR algorithm, which is reported in Table 1. Also, the results are compared with the Genetic algorithm. The algorithms are implemented in Matlab R2011a environment on a Notebook with Windows 7 Ultimate, CPU 2.53 GHz and 4.00 GB RAM. In GPR algorithm, we set the parameters as  $\varepsilon = 10^{-6}$ , m = 10,  $K_{max} = 100$ ,  $J_{max} = 5$ , L = 2 and  $\delta_k = \frac{2}{k+1}$ . The results of the GPR and Genetic algorithms, specified in the Table 2, where the values  $M_{Errorx}$  and  $M_{Errory}$  are the average error of the numerical results after 20 runs for each test problem. The test functions  $F_2$ ,  $F_4$  and  $F_6$  are multi-modal functions. From the last row of the Table 2, it is clear that GPR algorithm is more accurate than the Genetic algorithm in point of the summation of the errors.

## R. ZAVVAR ET AL.

Function	Function rule	<i>x</i> <sub>0</sub>
$F_1$	$  x  ^2$	[2,3]
$F_2$	$2 - (e^{-x^2} + 2e^{-(x-3)^2})$	1
$F_3$	Rosenbrock(n = 4)	$\left[0.9, 0.9, 0.9, 0.9\right]$
$F_4$	$x\cos(x)$	0.7
$F_5$	Rosenbrock(n = 2)	[0.9, 0.9]
$F_6$	x(x-1)(2x-3)(x-4)(x-5)(x-6)	2
$F_7$	$1 + (x_1 + x_2 + 1)^2 (19 - 14x_1 + 3x_1^2 - 14x_2 + 6x_1x_2 + 3x_2^2)$	[0.1, 0.9]
$F_8$	$0.25x_1^4 - 0.5x_1^2 + 0.1x_1 + 0.5x_2^2$	[-1, 0]
$F_9$	$4x_1^2 - 2.1x_1^4 + \frac{1}{3}x_1^6 + x_1x_2 - 4x_2^2 + 4x_2^4$	[0.8, -0.6]
<i>F</i> <sub>10</sub>	$4x_1^2 - 4x_1x_2 + 2x_2^2$	[0.001, 0.001]

TABLE 1. The rules of the test functions and the initial points of GPR algorithm.

TABLE 2. The numerical results of the GPR and Genetic algorithms for test functions in Table 1.

	GPR algorithm		Genetic algorithm	
Function	M <sub>Errorx</sub>	<i>M</i> <sub>Errory</sub>	M <sub>Errorx</sub>	<i>M</i> <sub>Errory</sub>
$F_1$	$4.3717 \times 10^{-9}$	$1.9267 \times 10^{-17}$	$8.9997 \times 10^{-3}$	$8.0946 \times 10^{-5}$
$F_2$	$1.8 \times 10^{-3}$	$6.58 \times 10^{-6}$	$2.6598 \times 10^{-3}$	$1.2205 \times 10^{-4}$
$F_3$	$3.93 \times 10^{-2}$	$3.86 \times 10^{-4}$	$9.6986 \times 10^{-1}$	$8.8297  imes 10^{-1}$
$F_4$	$1.85 \times 10^{-5}$	$2.86 \times 10^{-5}$	$1.7323 \times 10^{-3}$	$3.4258 \times 10^{-5}$
$F_5$	$1.22 \times 10^{-2}$	$3.6147 \times 10^{-5}$	$2.8013 \times 10^{-1}$	$2.5385 \times 10^{-2}$
$F_6$	$1.7067 \times 10^{-5}$	$3.25 \times 10^{-5}$	$4.5648 \times 10^{-3}$	$2.2991 \times 10^{-3}$
$F_7$	$8.5367 \times 10^{-9}$	$4.9167 \times 10^{-14}$	$9.5689 \times 10^{-1}$	2.9980
$F_8$	$1.9473 \times 10^{-5}$	$1.3926 \times 10^{-5}$	$4.7958 \times 10^{-2}$	$1.8663 \times 10^{-3}$
$F_9$	$6.05 \times 10^{-5}$	$2.85 \times 10^{-5}$	$2.7395 \times 10^{-2}$	$4.9736 \times 10^{-3}$
$F_{10}$	$5.6865 \times 10^{-9}$	$2.4969 \times 10^{-17}$	$4.9584 \times 10^{-2}$	$1.9045 \times 10^{-3}$
Sum of Errors	$5.35  imes 10^{-2}$	$5.32 \times 10^{-4}$	2.35	3.93

#### REFERENCES

- 1. E. G. Birgin, I. Chambouleyron and J. M. Martnez, *Estimation of the Optical Constants and the Thickness of Thin Films Using Unconstrained Optimization*, Journal of Computational Physics, **151** 2 (1999) 862 880.
- 2. C. K. Ng, L.Z. Liao and D. Li, A Globally Convergent and Efficient Method for Unconstrained Discrete-Time Optimal Control, Journal of Global Optimization, 23 (2002) 401 421.
- 3. J. Nocedal and S. J. Wright, Numerical Optimization, Springer, New York, NY, USA, Second edition, (2006).
- 4. W. Sun, Y.X. Yuan, Optimization Theory and Methods Nonlinear Programming, Springer, NewYork, (2006).
- 5. R. Ziadi, R. Ellaia and A. Bencherif-Madani *Global optimization through a stochastic perturbation of Polak-Ribiere conjugate gradient method*, Journal of computational and applied mathematics, **317** (2016).
- 6. J. D. Pinter, *Global optimization: scientific and engineering case studies*, Springer Science & Business media, (2006).
- 7. J. Brownlee, Clever algorithms: nature-inspired programming recipes, Jason Brownlee, (2011).

- 8. M. Pogu, J. E. Souza De Cursi *Global optimization by random perturbation of the gradient method with a fix parameter*, Journal of Global Optimization, **5** (1994) 159 180.
- 9. J. Sun, J. Zhang Global convergence of conjugate gradient methods without line search, Ann. Oper. Res, 103 (2001) 161 173.