International Journal of Students' Research in Technology & Management
eISSN: 2321-2543, Vol 7, No 3, 2019, pp 01-06
https://doi.org/10.18510/ijsrtm.2019.715

ON NON-QUADRATIC PENALTY FUNCTION FOR NON-LINEAR PROGRAMMING PROBLEM WITH EQUALITY CONSTRAINTS

Raju Prajapati[1*], Om Prakash Dubey[2], Ranjit Pradhan[3]
[1]Amity University Jharkhand, Ranchi, Jharkhand - 834001 (India), [2]Veer Kunwar Singh University, Ara, Bihar - 802301 (India), [3]Bengal College Of Engineering & Technology, Durgapur, W.B. - 713212 (India).
Email: [1*]raju.prajapati20102011@gmail.com, [2]omprakashdubeymaths@gmail.com, [3]ranjitmath@yahoo.com

Article History: Received on 12th May 2019, Revised on 29th June 2019, Published on 19th July 2019

Abstract

Purpose: The present paper focuses on the Non-Linear Programming Problem (NLPP) with equality constraints. NLPP with constraints could be solved by penalty or barrier methods.

Methodology: We apply the penalty method to the NLPP with equality constraints only. The non-quadratic penalty method is considered for this purpose. We considered a transcendental i.e. exponential function for imposing the penalty due to the constraint violation. The unconstrained NLPP obtained in this way is then processed for further solution. An improved version of evolutionary and famous meta-heuristic Particle Swarm Optimization (PSO) is used for the same. The method is tested with the help of some test problems and mathematical software SCILAB. The solution is compared with the solution of the quadratic penalty method.

Results: The results are also compared with some existing results in the literature.

Keywords*: Penalty Function, NLPP, Non-quadratic Penalty Function, Improved Particle Swarm Optimization, Optimization Test Problems.*

INTRODUCTION

The Non-Linear Programming Problem (NLPP) is computationally hard as compared to the Linear Programming Problem (LPP). Further, In NLPP, there are two types: unconstrained and constrained NLPP. The constrained NLPP, which is under consideration for this paper is defined as:

Minimize $f(x)$ (1)

$$\text{Subject to } g_i(x) \le 0, \forall i = 1,2....r \text{ and } h_j(x) = 0, \forall j = r+1, r+2....m$$

The solution of equation (1) is achieved by various ways: Lagrangian, KKT (Feng, M. and Li, S., 2018) conditions, penalty and barrier methods are some of them (Bertsekas, D.P., 1999). Further, there are various methods, which are derived from these existing methods. For example, the logarithmic barrier method uses the logarithmic function to impose barriers obtained through the constraints (Bertsekas, D.P., 1999, Ben-Tal, A. and Zibulevsky, M., 1997). Inverse barrier function introduced in (Den Hertog, D., Roos, C. and Terlaky, T., 1994) is applied by taking the reciprocal of each constraint and applying the constant and adding to the objective function. Semi-penalty function method and Semi-augmented Lagrangian penalty function have some modifications from the existing conventional penalty method (Nie, P.Y., 2006). Interior and exterior penalty methods introduced, in which the interior penalty function is applied for the ill-defined objective function. The conventional quadratic penalty function or quadratic loss function is mostly used for almost all NLPP. Further, there are some modifications in KKT (Feng, M. and Li, S., 2018) condition also. For example, Approximate Strong KKT conditions (ASKKT) is used for multi-objective optimization. Approximate KKT conditions (AKKT) was introduced in (Haeser, G. and Schuverdt, M.L., 2011), a slight variation of AKKT is sufficient for the convex program, either for vibrational inequalities or optimization. Some methods are derived from the Lagrangian method also, which exist in literature. For example, the augmented Lagrangian method (Bertsekas, D.P., 1999) and two-phase augmented lagrangian method are there for distributed non-convex optimization and convex quadratic semi-definite programming respectively. All these methods work well on NLPP directly or similar convex/non-convex problems. In addition to the above, there are some methods in the literature, which are applicable to the NLPP without constraints. The steepest descent method, gradient descent method, Newton method are some of them. (Eberhart, R. and Kennedy, J., 1995)

In the present paper, we are concerned about the NLPP with equality constraint only. Therefore, the problem under consideration is defined as:

Minimize $f(x)$ (2)

$$\text{Subject to } h_j(x) = 0, \forall j = 1,2...l$$

Which is derived from the equation (1) by eliminating all the inequality constraints and considering only l equality constraints. In this paper, we will be solving equation (2) by non-quadratic penalty function. A non-quadratic penalty function is introduced in (Nie, P.Y., 2006). The method is introduced as a semi non-quadratic penalty function and works well for equation (1). The word semi is used in the article to deal with equality constraints only leaving the inequality constraints as the original problem. We consider only the equality constraints to use and apply this method.

Therefore, in the present paper, we are applying this algorithm over NLPP with equality constraints with the help of an improved PSO, i.e. we are checking the performance measure and further comparing it with the performance measure of the quadratic penalty function. We are using the SCILAB programming language for this purpose.

QUADRATIC PENALTY FUNCTION: A penalty function for (1) is defined as:

$$\text{Minimize } f(x)+\sum_i c_{1i}\max(0,g_i(x))^2+\sum_j c_{2j}(h_j(x))^2 \quad (3)$$

We are considering only the equality constraint, therefore the penalty function is

$$\text{Minimize } f(x)+\sum_i c_i(h_i(x))^2 \quad (4)$$

Which is also called quadratic penalty function or quadratic loss function.

Consider the simple example

$$\text{Minimize } 100/x$$

$$\text{Subject to } x=5. \quad (5)$$

The penalty could be imposed by the help of constraints available. Therefore, the problem (5) becomes

$$\text{Minimize } 100/x+c(x-5)^2 \quad (6)$$

The meaning of equation (6) is simple. We have to minimize a sum which consists of a given function and a penalty due to constraint. If we take $x=5$, we will have a minimum value of $100/x+c(x-5)^2$. In all other cases under the domain, we get a value greater than that. The purpose of applying the penalty is well served here. The minimization function will pull the entire function to the minimum value, which is required.

Non-quadratic Penalty functions: Inspired from the paper (Nie, P.Y., 2006), dealing with semi-non quadratic penalty function, we define the non-quadratic penalty function for (2) as follows:

$$\text{Minimize } f(x)+\sum_i c_i(e^{(h_i(x))^2}-1) \quad (7)$$

If we consider the above over the problem (5), then the non-quadratic penalty function could be

$$\text{Minimize } 100/x+c(e^{(x-5)^2}-1) \quad (8)$$

If we consider (8), we have a penalty again as $c(e^{(x-5)^2}-1)$, which is zero for $x=5$, and greater for all other values. But this penalty is not using the conventional approach. This comes under non-quadratic penalty methods. Also, the exponential function gives a higher value even for a small change x. Therefore, this penalty function is more sensitive to the input values compared to the quadratic penalty function.

The present paper is discussing the performance measure of this method.

PARTICLE SWARM OPTIMIZATION: Proposed by Kennedy and Eberheart in 1995, it is an evolutionary algorithm, also called a meta-heuristic inspired by nature. It is just similar to the flock of birds or fish searches for food and comes together if some food is found randomly at someplace. Initially, we take some particle called solution randomly over the searched space. In each iteration, we improve the solution by taking into consideration the best particle's position among all particles and each particle's own best position. I.e. each particle follows the best particle without forgetting its personal best position. Each particle is supposed to update its position by the following two equations:

$$v_{id}^{k+1}=v_{id}^{k}+c_1r_1^{k}(pbest_{id}^{k}-x_{id}^{k})+c_2r_2^{k}(gbest^{k}-x_{id}^{k}) \quad (9)$$

$$x_{id}^{k+1} = x_{id}^{k} + v_{id}^{k+1} \quad (10)$$

Where v_{id}^{k+1} represents the velocity of ith particle at k+1th iteration. r_1^k and r_2^k are two random numbers generated at kth iteration. c_1 and c_2 are two constants usually taken as 2. $pbest_{id}^{k}$ and $gbest^{k}$ are the best positions of i^{th} particle and overall best position of any particle at kth iteration. Further, x_{id}^{k+1} represents the position in k+1th iteration, which depends on x_{id}^{k+1} and v_{id}^{k+1}, the previous position and velocity updated.

IMPROVEMENTS IN PSO:

1. Inertia Weight: Consider the following updated equation (9): $v_{id}^{k+1} = wv_{id}^{k} + c_1 r_1^k (pbest_{id}^{k} - x_{id}^{k}) + c_2 r_2^k (gbest^{k} - x_{id}^{k})$ where, w is the inertia weight (Angeline, P.J., 1998). It changes the impact of the previous velocity over the next velocity. It is usually taken to be 1. Inertia weight varying from 0.9 to 1.2 is useful in better results. Larger inertia weight means larger global searchability. Similarly, smaller inertia weight means local searchability.
2. Constriction Factor: Consider the updated equation of (9) as: $v_{id}^{k+1} = \chi(v_{id}^{k} + c_1 r_1^k (pbest_{id}^{k} - x_{id}^{k}) + c_2 r_2^k (gbest^{k} - x_{id}^{k}))$ where χ is called the constriction factor (Angeline, P.J., 1998). The value of this is $\chi = \frac{2}{|2 - \varphi - \sqrt{\varphi^2 - 4\varphi}|}$ where $\varphi = c_1 + c_2 > 4$. Usually, this makes convergence faster. Generally, this value is taken to be 4.1 for better convergence. The value of χ is .729 for this case. The weight factor may be treated as a special case of constriction factor in PSO.
3. Proper selection of particles: The selection of the initial particle plays an important role in the entire solution of PSO. The number of iteration taken is less if we generate all initial particles close to the optimal solution. Also, if we take the higher number of the particle at the start, we find the solution in comparatively less number of iterations. This is because a higher number of particles results in faster exploration within the search space. In, a special selection strategy is chosen in which each iteration consists of some best particles from the previous iteration, and removal of worst particles, which unnecessarily consumes the system's time. Therefore, proper particle selection is a must in PSO.
4. Random number generator: Consider the updated equation (9): $v_{id}^{k+1} = v_{id}^{k} + c_1 r_1^k (pbest_{id}^{k} - x_{id}^{k}) + c_2 (1 - r_1^k)(gbest^{k} - x_{id}^{k})$, which consists of only one random number (Li, W.T., Shi, X.W. and Hei, Y.Q., 2008). The two benefits are oblivious. First, we get a decrease in system time, which was used earlier to generate two random numbers. Second, we get an impact either from $pbest_{id}$ or $gbest$ from on the next iteration. The cases of either random numbers as small or both large are avoided, which results in very little progress or going out from the searched region respectively. (Rao, R. and Patel, V., 2013)

Although there are many more strategies developed till now in PSO improvement (Du, K.L. and Swamy, M.N.S., 2016), the present algorithm uses the improved PSO in inertia weight and random number generator mentioned above.

METHODOLOGY

Consider again the equation (7): $f(x) + \sum_i c_i (e^{(h_i(x))^2} - 1)$, which is an unconstrained optimization problem. This unconstrained problem uses the penalty as a non-quadratic penalty. The exponential term is used in place of the general quadratic penalty or quadratic loss function. Further, the unconstrained optimization problem obtained in this way is processed for the solution. The solution is done by an improved PSO algorithm. (Andrei, N., 2008)

Also, the method is compared with the solution of the unconstrained optimization problem (4) $f(x) + \sum_i c_i (h_i(x))^2$, which is obtained by the usual penalty function, called the quadratic penalty function.

Therefore, in our present paper, we are mainly focusing two things: (i) Solution of NLPP by non-quadratic penalty function ii) Solution by a particular improved version of PSO, which consists of only one random number in each iteration, which further decrease the computational load and guarantees an improvement in each iteration.

International Journal of Students' Research in Technology & Management
eISSN: 2321-2543, Vol 7, No 3, 2019, pp 01-06
https://doi.org/10.18510/ijsrtm.2019.715

The computational experiment is conducted over 5 test problems (Rao, R. and Patel, V., 2013, Andrei, N., 2008, and Younis, A., and Dong, Z., 2010] with arbitrary constraint, which are as follows:

Test problem 1: Rosenbrock function constrained to a circle

$$\text{Minimize } f(x,y) = (1-x)^2 + 100(y-x^2)^2, \text{ subject to } x^2 + y^2 = 2$$

Test problem 2: Rastigrin function constrained to a circle

$$\text{Minimize } f(x,y) = 10 + (x^2 + 10\cos 2\pi x) + (y^2 + 10\cos 2\pi y)$$

$$\text{Subject to } x^2 + y^2 = 1$$

Test problem3: Sphere function constrained to a plane:

$$\text{Minimize } f(x,y) = \sum_i x_i^2$$

$$\text{Subject to } \sum_i x_1 = 1$$

Test problem 4: Booth function constrained over a circle:

$$\text{Minimize } f(x,y) = (x+2y-7)^2 + (2x+y-5)^2$$

$$\text{Subject to } x^2 + y = 4$$

Test problem 5: Himmelblau's function constrained over a circle.

$$\text{Minimize } f(x,y) = (x^2 + y - 11)^2 + (x + y^2 - 7)$$

$$\text{Subject to } x^2 + y^2 = 13$$

All the test problems considered above are mostly used test problems in literature. The constraint is been taken arbitrarily.

EXPERIMENTAL RESULT

The above function is processed through equation (7) and equation (4) using the PSO algorithm in SCILAB. The improved PSO used consists of weight factor 0.01 (similar to) and one random number is used. A total of 800 iterations are performed and 50 particles were taken. Also, the input was given within an interval length of a maximum of 20 units in each dimension. The results are observed as follows:

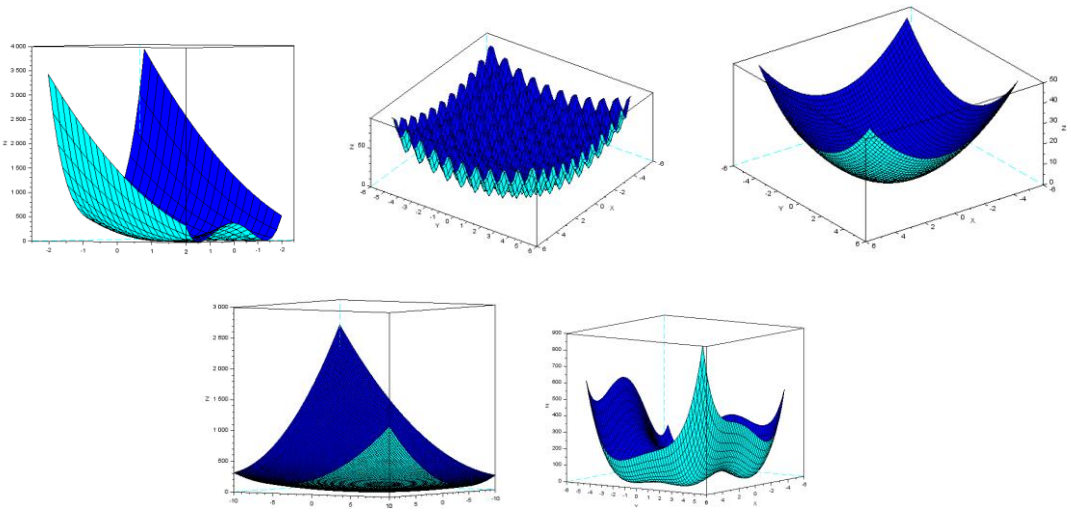

Figure 1: representation of all the test function without arbitrary constraints

Table 1: (exponential penalty, c= 10)

Test function	x range	y range	x optimal	y optimal	z optimal
Test function 1	[-2.048,2.048]	[-2.048,2.048]	1	1	3.466D-14
Test function 2	[-5.12,5.12]	[-5.12,5.12]	.0000003	-.9962131	.996644
Test function 3	[-5,5]	[-5,5]	-2.626D-89	5.043D-89	3.23D-177
Test function 4	[-10,10]	[-10,10]	1.018005	2.9663056	.0025274
Test function 5	[-5,5]	[-5,5]	3.093843	2.0966583	.2428109

Table 2: (quadratic penalty, c= 10)

Test function	x range	y range	x optimal	y optimal	z optimal
Test function 1	[-2.048,2.048]	[-2.048,2.048]	1	1	2.208D-15
Test function 2	[-5.12,5.12]	[-5.12,5.12]	-.9962175	4.688D-08	.9966479
Test function 3	[-5,5]	[-5,5]	-2.52D-161	1.36D-161	1.59D-320
Test function 4	[-10,10]	[-10,10]	1.0848342	2.9165948	.1015092
Test function 5	[-5,5]	[-5,5]	3.093843	1.8411388	.4518183

CONCLUSION AND FUTURE WORK

With the analysis of a special penalty function, which is exponential and non-quadratic in nature, we have simply observed the better accuracy (on an average) as compared to the conventional quadratic penalty function. We can see the results obtained for several test functions in this regard. The above algorithm has been applied over the well-known test functions with any arbitrary constraint(s). Further, we can also increase the number of constraints and the number of dimensions also. The above algorithm has a simple demerit. Since the penalty function is exponential in nature, it's highly sensitive i.e. it increases rapidly as compared to the quadratic penalty. Therefore, the specification of the domain in each dimension is more necessary in this algorithm. Also, there should not be any crest or trough going infinitely high within the specified region of the unconstrained test functions (with exponential penalty function). This will do nothing except the failure of software in finding the correct solution. We have chosen over 800 iterations for our results. This is just for showing the better accuracy level of results (in most cases) within an existing article, which used 1000 iterations and 100 particles. Only the test problems are different (Parsopoulos, K.E. and Vrahatis, M.N., 2002). The comparison of the above two algorithms is better in comparatively less number of iterations. We could also choose different values of penalty constants. The above work could also be generalized to the Non-Linear Programming Problem with inequality constrained or mixed constrained.

REFERENCES

1. Andrei, N., 2008. An unconstrained optimization test functions collection. *Adv. Model. Optim*, *10*(1), pp.147-161.
2. Angeline, P.J., 1998, May. Using selection to improve particle swarm optimization.In Evolutionary Computation Proceedings, 1998. IEEE World Congress on Computational Intelligence., The 1998 IEEE International Conference on (pp. 84-89). IEEE.
3. Ben-Tal, A. and Zibulevsky, M., 1997. Penalty/barrier multiplier methods for convex programming problems. *SIAM Journal on Optimization*, *7*(2), pp.347-366. https://doi.org/10.1137/S1052623493259215
4. Bertsekas, D.P., 1999. *Nonlinear programming* (pp. 191-276). Belmont: Athena scientific.
5. Den Hertog, D., Roos, C. and Terlaky, T., 1994. Inverse barrier methods for linear programming. *RAIRO-Operations Research*, *28*(2), pp.135-163. https://doi.org/10.1051/ro/1994280201351
6. Du, K.L. and Swamy, M.N.S., 2016. Particle swarm optimization. In *Search and optimization by metaheuristics*(pp. 153-173). Birkhäuser, Cham. https://doi.org/10.1007/978-3-319-41192-7_9
7. Eberhart, R. and Kennedy, J., 1995, October. A new optimizer using particle swarm theory. In Micro Machine and Human Science, 1995.MHS'95., Proceedings of the Sixth International Symposium on (pp. 39-43). IEEE.
8. Eberhart, R.C. and Shi, Y., 2000. Comparing inertia weights and constriction factors in particle swarm optimization.In Evolutionary Computation, 2000.Proceedings of the 2000 Congress on (Vol. 1, pp. 8488).IEEE.
9. Feng, M. and Li, S., 2018. An approximate strong KKT condition for multiobjective optimization. *TOP*, *26*(3), pp.489-509. https://doi.org/10.1007/s11750-018-0491-6
10. Haeser, G. and Schuverdt, M.L., 2011. On approximate KKT condition and its extension to continuous variational inequalities. *Journal of Optimization Theory and Applications*, *149*(3), pp.528-539. https://doi.org/10.1007/s10957-011-9802-x
11. Li, W.T., Shi, X.W. and Hei, Y.Q., 2008. An improved particle swarm optimization algorithm for pattern synthesis of phased arrays. Progress In Electromagnetics Research, 82, pp.319-332. https://doi.org/10.2528/PIER08030904

12. Li, X., Sun, D. and Toh, K.C., 2018. QSDPNAL: a two-phase augmented Lagrangian method for convex quadratic semidefinite programming. *Mathematical Programming Computation*, pp.1-41. https://doi.org/10.1007/s12532-018-0137-6
13. Nie, P.Y., 2006. A new penalty method for nonlinear programming. *Computers & Mathematics with Applications*, *52*(6-7), pp.883-896. https://doi.org/10.1016/j.camwa.2006.05.012
14. Parsopoulos, K.E. and Vrahatis, M.N., 2002. Particle swarm optimization method for constrained optimization problems. Intelligent Technologies–Theory and Application: New Trends in Intelligent Technologies, 76(1), pp.214-220. https://doi.org/10.1142/9789812777140_0021
15. Rao, R. and Patel, V., 2013. Comparative performance of an elitist teaching-learning-based optimization algorithm for solving unconstrained optimization problems. *International Journal of Industrial Engineering Computations*, *4*(1), pp.29-50. https://doi.org/10.5267/j.ijiec.2012.09.001
16. Younis, A. and Dong, Z., 2010. Trends, features, and tests of common and recently introduced global optimization methods. *Engineering Optimization*, *42*(8), pp.691-718. https://doi.org/10.1080/03052150903386674

