Optimization of Constrained Objective Function by Penalty Function Method

Mayank Pathania(204103314)  
*Machine Design, Mechanical Engineering Department*  
*IIT, Guwahati*Assam, India  
mayank\_pathania@iitg.ac.inDharmik Patel(204103318)  
*Machine Design, Mechanical Engineering Department*  
*IIT, Guwahati*Assam, India  
pdharmik@iitg.ac.inDeepak Sharma  
*Professor, Mechanical Engineering Department*  
*IIT, Guwahati*Assam, India  
dsharma@iitg.ac.in

*Abstract*—Constrained optimization is useful in every domain of engineering, as every problem comes with some limitations with it. In this report, optimization of constrained objective function by Penalty Function method has been presented. This Penalty Function method uses Marquardt’s method, Bounding-Phase method and Newton’s-Raphson method for unidirectional search.

Keywords—optimization, minimization, penalty, objective function, constrained optimization, unconstrained optimization, unidirectional search

# Introduction

We hear ‘Optimization’ word very often. What does it mean? Optimization of a given thing is to reach at maximum or minimum limit of that thing, whatever required. Like, optimization of production in a company means to achieve maximum production rate of a product or optimization of a cost of a product means to design a process of production which minimize the cost of a product.

We sometimes encounter with the question that why do we need to find the extreme point of a function. So, answer of that question is, a function, which is given can, be a cost function of a product or a production time function of a manufacturing plant or any other function. Then we need to minimize cost and time of production (minimization or maximization depends on a function, on which context it is taken). A function can be single variable or multi variable function. But in this report we will talk only about single variable function, which needs to be optimized.

There are many methods are available for optimization of an objective function. Here, we will discuss only about the Penalty Function method of constrained optimization and also see briefly about the Marquardt’s method, Bounding-Phase method and Newton-Raphson method of unconstrained optimization.

# Optimization Function

## Constrained Optimization Function

Objective function, which is needed to be optimized (either maximized or minimized), can be of mainly two types, constrained or unconstrained. Constrained objective function can be defined as the variables of objective function are dependent on other equation/s, which govern/s the value of variables. Those governing equations can be of equality or inequality types. Here, general constrained objective function is given:

Minimize

Subjected to,

## Unconstrained Optimization Function

Whereas in unconstrained objective function, there is only objective function, of single or multi-variables, presents. There will be no any other equations which governs value of variables.

Minimize

Now, we have seen two types of function. We will see optimization methods, next.

# Optimization Methods

You may have noticed that the addition of constraints to an optimization problem has the effect of making it much more difficult. Every point, we are getting from the previous iteration, can’t always satisfies all the constrained equations of a constrained optimization function. There can be possibilities that only some constrained equations are satisfied and others are not, i.e., constraint is violated. In this case, objective function is penalized. Using Transformation methods like Penalty Function method or Method of Multipliers, the constraint problem is transformed into sequence of unconstrained problems by adding penalty term of each violated constraint. Here, we will only see Penalty Function method, not Method of Multipliers.

The goal of penalty functions is to convert constrained problems into unconstrained problems by introducing an artificial penalty for violating the constraint.

## Penalty Function Method

Penalty function methods work in a series of sequence, each time modifying a set of penalty parameters and starting a sequence with the point obtained in the previous sequence.

Where, R a is the set of penalty parameters, Ω is the penalty term chosen to favor the selection of feasible point over infeasible point.

We saw that how penalty function has been made from given objective function and constrained equations. We will see, how Ω is chosen by any of the following two methods and then effect of R will be discussed.

### Interior Penalty Methods

These methods work well for interior points in a search space, i.e., feasible points. They penalize the points which come closer to the constraint boundary. To start the search, a feasible point is needed. For interior penalty terms, initially large value of R is taken, which is reduced gradually. Here, some interior penalty terms are given:

#### Log Penalty Term

It is an interior penalty term because it penalizes only feasible points as in natural logarithm can’t be used.

#### Invrese Penalty Term

### Exterior Penalty Methods

As name suggests, these methods penalize exterior points, i.e., infeasible points, not feasible points. A feasible or infeasible point can be taken as initial guess. In exterior penalty methods, value of is taken small at time of starting, which increases gradually.

#### Perabolic Penalty Term

Only equality constraints are handled by parabolic penalty term.

#### Infinite Barrier Penalty Term

Where, J is set of constrains at a current point, which are violated.

#### Bracket Operator Penalty Term

Bracket operator works when , i.e., when constraints are violated.

* Effect of R

If by applying the constrained equations, optimum solution of constrained problem is remaining as same as unconstrained problem, then penalty parameter will solve the constrained problem because constrained equations do not differ the optimum point of unconstrained problem.

If constrained equations make optimum point of unconstrained problem infeasible, the we should apply a series of sequences of unconstrained objective function by penalizing the objective function. When unconstrained objective function is subjected to constrained equations, then usually optimum solution lies on one of the constrained equations.

In successive sequences of penalty function method, R value changes which depends on interior and exterior penalty terms.

* Algorithm

1. Choose two termination parameters ; an initial solution ; a penalty term Ω; and an initial penalty parameter . Choose a parameter c to update R such that is used for interior penalty terms and is used for exterior penalty terms. Set .
2. Form .
3. Starting with , find such that is minimum for a fixed value of . Use to terminate the unconstrained search. (Here, we use unidirectional search to find next point.)
4. Is ? If yes, set and **terminate**; else go to Step 5.
5. Choose . Set and go to Step 2.

This is the algorithm for Penalty Function method for optimization of constrained objective function. Now, we will see Marquardt’s method for unconstrained multi-variable problem and in that unidirectional search as a combination of Bounding-Phase method and Newton-Raphson method.

## Unidirectional Search

As we transformed constrained objective function into unconstrained multi-variable objective function by using Penalty Function method, now we can use any method of unconstrained optimization to get the optimum point at particular sequence. We have already performed multi-variable optimization in previous assignment, so algorithm is described briefly.

### Marquardt’s Method

Marquardt’s method is gradient-based optimization method for unconstrained objective function of multi-variables.

* Algorithm

1. Choose initial point to start the algorithm, the maximum number of iterations M, termination parameter to terminate algorithm, after attaining particular value of (Gradient of a function), . Set (iteration counter) and (or any large number).
2. Calculate .
3. Check, if or , then terminate the algorithm, else, proceed ahead.
4. Calculate descent direction . And fine the next point .^^
5. If , then go to step 6, else go to step 7.
6. and . Go to step 2.
7. and go to step 4.

(^^To find the next point, we can use unidirectional search methods of single-variable, this feature is included in our code of Marquardt’s method.)

### Bounding-Phase Method

Unidirectional search is preformed using Bounding-Phase method and Newton-Raphson method.

* Algorithm

1. Take an initial guess () from the user and also increment value . Set k = 0.
2. Find function value at and at two points at distance on either side of . If, , then take the value of ∆ as positive value, else if , then take the value of ∆ as negative value, else ask for the another initial guess.
3. Calculate .
4. If , then increase the k by 1 and go to step 3, else terminate the algorithm and our extreme point will lie in the interval .

### Newton-Raphson Method

* Algorithm

1. Choose initial guess and a small number . Set t = 1 and compute .
2. Compute .
3. Calculate and .
4. If , then terminate the algorithm, else set and go to step 2.

* Process of constrained objective function

We make penalty function using constrained objective function, constrained equations, suitable penalty term and parameter. Here, we have kept that Penalty Function will run for maximum of 7 times, because after that value of is becoming , which is too high with respect to the function values. Then penalty function and initial guess are given to the Marquardt’s method.

In Marquardt’s method we call the unidirectional search, which is a combination of Bounding-Phase method and Newton-Raphson method, which gives the optimum value of variables for particular value of .

Then termination condition is checked and according to the satisfaction of condition either Penalty Function algorithm is terminated or value or is updated.

* Complexity of the Algorithm

We understood the process of algorithm, that one by one different methods have been called at different levels of algorithm to calculate the optimum point and function value at a particular sequence.

We already know that Marquardt’s method is computationally expensive, because it involves calculation of Hessian matrix and then its inverse. And this calculation increases complexity and also slows down the algorithm.

Here, we are calling Marquardt’s method in every sequence, which further takes more time to perform algorithm.

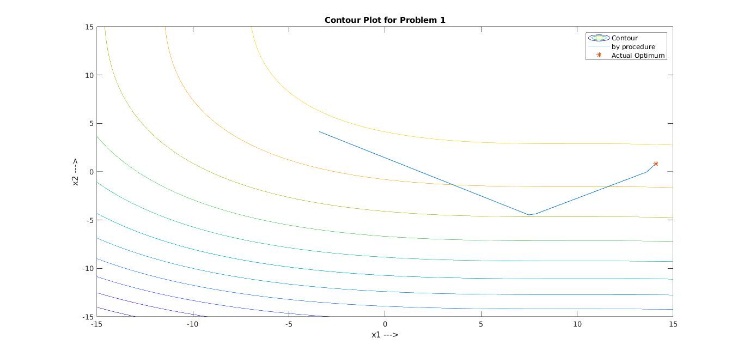
# Results And Discussion

In this section, we are going to discuss results of our constrained problems statements, which are given in the Appendix A. In Appendix B, the value of Penalty Function, constraints violation are given for these 10 optimum solution.

## Problem 1

Objective function of Problem 1 is of 2 variables ( and ) and having 2 constrained equations ( and ). So, in Penalty function, there will be total 6 (2 + 4(bound constraints on and ) constraints.

Penalty Function algorithm has been run for 10 times and optimum point obtained from each time has been tabled below, with value of in the last sequence. On performing 10 times, we are getting answer, i.e., algorithm is converging to very close to optimum solution, almost all the time. Why it is not converging sometimes is because of getting such direction during unidirectional search which leads to the another local minimum function value point.



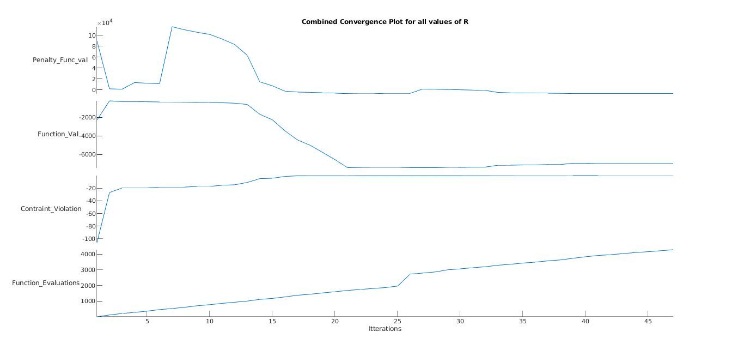
As we have used exterior penalty term, value of , starting from , is increased by factor after performing each sequence.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| No. |  |  |  | Function Value |
| 1 | 10000000 | 14.1168 | 0.8858 | -6913.6738 |
| 2 | 14.0929 | 0.8384 | -6966.9275 |
| 3 | 14.0852 | 0.8227 | -6984.5941 |
| 4 | 14.0893 | 0.8312 | -6975.0361 |
| 5 | 14.0887 | 0.8300 | -6976.4029 |
| 6 | 14.0966 | 0.8461 | -6958.2574 |
| 7 | 14.1559 | 0.9637 | -6826.6250 |
| 8 | 14.0894 | 0.8312 | -6975.0205 |
| 9 | 14.0902 | 0.8927 | -6973.3948 |
| 10 | 14.0886 | 0.8297 | -6976.7353 |

Our optimum value of objective function is -6961.8139, at

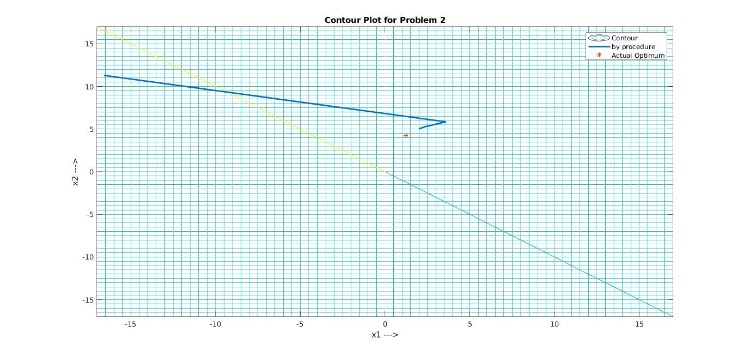
|  |  |  |  |
| --- | --- | --- | --- |
| Solutions |  |  | Function Value |
| Best | 14.0966 | 0.8461 | -6958.2574 |
| Worst | 14.1559 | 0.9637 | -6826.6250 |
| Mean | 14.0994 | 0.8572 | -6952.6667 |
| Median | 14.0898 | 0.8348 | -6974.2077 |
| Standard Deviation | 0.0217 | 0.0446 | 48.5723 |

From the fmincon() of MATLAB, we got solution, and . We can see that our best solution is very close to the solution got by MATLAB inbuilt function.



## Problem 2

Problem 2 has also 2 variables ( and ) objective function. It is constrained by two equations ( and ). Here, also 6 constraints will come in Penalty function.



, increment factor

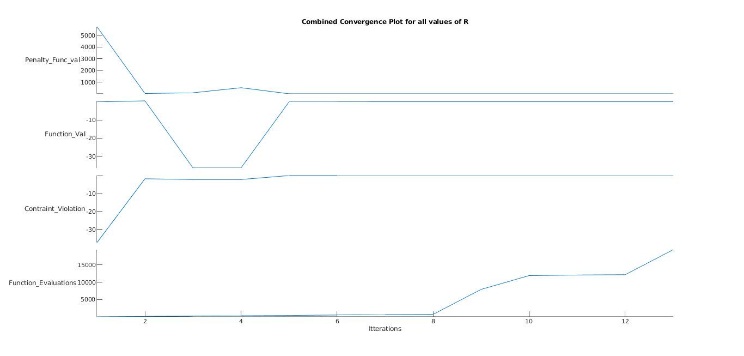
On performing algorithm 10 times, we are getting close solution to the optimum solution many of the times.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| No. |  |  |  | Function Value |
| 1 | 10000 | 1.0628 | 3.7494 | -0.0098 |
| 2 | 100000000 | 1.7339 | 4.7467 | 0.0291 |
| 3 | 10000 | 1.2280 | 4.2529 | 0.0958 |
| 4 | 1000000 | 1.5852 | 3.5450 | 0.0024 |
| 5 | 10000 | 1.2278 | 4.2451 | 0.0958 |
| 6 | 1000000 | 1.0375 | 3.8063 | -0.0022 |
| 7 | 100000000 | 1.7340 | 4.7456 | 0.0291 |
| 8 | 10000 | 1.0299 | 3.8270 | -0.0011 |
| 9 | 10000 | 1.0323 | 3.8202 | -0.0014 |
| 10 | 100000 | 1.3097 | 3.4435 | 0.0262 |

Our optimum value of objective function is 0.0958, at

|  |  |  |  |
| --- | --- | --- | --- |
| Solutions |  |  | Function Value |
| Best | 1.2280 | 4.2529 | 0.0958 |
| Worst | 1.7339 | 4.7467 | 0.0291 |
| Mean | 1.2981 | 4.0182 | 0.0293 |
| Median | 1.2279 | 3.8236 | 0.018 |
| Standard Deviation | 0.2864 | 0.4611 | 0.0369 |

For this problem, we got and optimum point , by fmincon function of MATLAB and our solution by penalty function is also around it.



## Problem 3

This problem contains objective function with only 3 variables (, and ), but it’s constrained equations(, , , , and ) are of 8 variables. These 8 variables are depended on each other by different constrained equations.

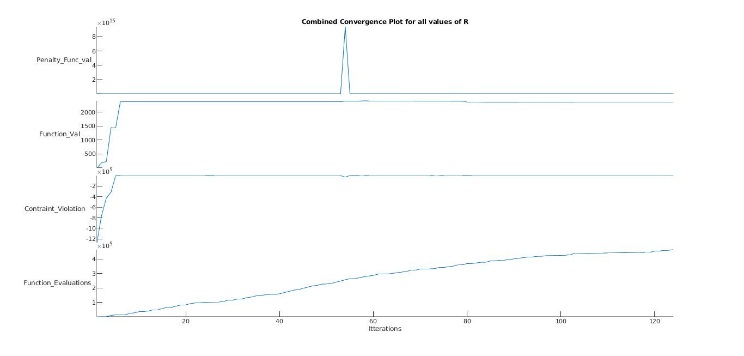
, increment factor .

Performing the algorithm 10 times, algorithm is not converging to the optimum solution any of the time. Instead of converging at optimum solution, it is diverging and gives the value of variables in the multiple of . Here, few value at are shown for 2 runs of the algorithm:

|  |  |  |
| --- | --- | --- |
| No. | 1 | 2 |
|  | 10000000 | 10000000 |
|  | 1758.4886 | 4.7725e19 |
|  | 6904.1073 | -1.2740e18 |
|  | 1474.3077 | -5.7744e19 |
|  | 832.4859 | 4.1384e19 |
|  | 617.8326 | 1.2049e20 |
|  | 991.6018 | 3.6896e19 |
|  | 793.6226 | 5.3104e19 |
|  | 996.2960 | -4.6726e20 |
| Function Value | 10136.9036 | -1.1293e19 |

Our optimum value of objective function is 7049.3307, at

Here, we can see that for one time we have got solution which can be taken granted, whereas for the second time we have got solution which can’t be accepted. This type of solution is because of unidirectional search, in which Hessian matrix would have been become of zero value, whose inverse is not possible.



# Conclusion

* Penalty Function method is computationally expensive as it includes Marquardt’s method. And from the previous assignment, we know that Marquardt’s method is complex as it involves calculation of Hessian matrix and its inverse.
* For the 1st and 2nd problem, we are getting close solution to the optimum solution, most of the times. Sometimes it is not converging to the optimum solution, one of the reason for that might be that solution is converging to the other local optimum point. In these problems, restart helps in getting optimum point sometimes.
* For the 3rd problem, optimum solution is not obtained almost all the times. Here, reason is, Hessian matrix in Marquardt’s method coming as zero, turns the direction of unidirectional search in direction other than optimum direction for that iteration and our problem starts diverging. Restarting doesn’t help in this problem.
* From these 3 problems, we can say that on increasing constrained equation on variables. It becomes difficult to get the optimum point.

# Appendix

## Constrained Optimization Problems

### Problem 1

Minimize ,

Subjected to ,

,

,

* Number of Variables: 2 variables

### Problem 2

Maximize ,

Subjected to ,

,

,

* Number of Variables: 2 variables

### Problem 3

Minimize .

Subjected to ,

,

,

,

,

,

,

, ,

,

* Number of Variables: 8 variables

## Results

### Problem 1

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| No. |  |  |  | Constraints Violation | Penalty Function Value | Function Value |
| 1 | 10000000 | 14.1168 | 0.8858 | 0 | -6913.6738 | -6913.6738 |
| 2 | 14.0929 | 0.8384 | -0.0043 | -6849.2379 | -6966.9275 |
| 3 | 14.0852 | 0.8227 | -0.0196 | -5064.6258 | -6984.5941 |
| 4 | 14.0893 | 0.8312 | -0.0114 | -63294966 | -6975.0361 |
| 5 | 14.0887 | 0.8300 | -0.0125 | -6190.3609 | -6976.4029 |
| 6 | 14.0966 | 0.8461 | -0.0002 | -6958.0253 | -6958.2574 |
| 7 | 14.1559 | 0.9637 | 0 | -6826.6250 | -6826.6250 |
| 8 | 14.0894 | 0.8312 | -0.0112 | -6314.3912 | -6975.0205 |
| 9 | 14.0902 | 0.8927 | -0.0097 | -6357.2866 | -6973.3948 |
| 10 | 14.0886 | 0.8297 | -0.0128 | -6154.2713 | -6976.7353 |

### Problem 2

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| No. |  |  |  | Constraints Violation | Penalty Function Value | Function Value |
| 1 | 10000 | 1.0628 | 3.7494 | 0 | 0.0098 | -0.0098 |
| 2 | 100000000 | 1.7339 | 4.7467 | 0 | -0.0291 | 0.0291 |
| 3 | 10000 | 1.2280 | 4.2529 | 0 | -0.0958 | 0.0958 |
| 4 | 1000000 | 1.5852 | 3.5450 | 0 | -0.0024 | 0.0024 |
| 5 | 10000 | 1.2278 | 4.2451 | 0 | -0.0958 | 0.0958 |
| 6 | 1000000 | 1.0375 | 3.8063 | 0 | 0.0022 | -0.0022 |
| 7 | 100000000 | 1.7340 | 4.7456 | 0 | -0.0291 | 0.0291 |
| 8 | 10000 | 1.0299 | 3.8270 | 0 | 0.0011 | -0.0011 |
| 9 | 10000 | 1.0323 | 3.8202 | 0 | 0.0014 | -0.0014 |
| 10 | 100000 | 1.3097 | 3.4435 | 0 | -0.0262 | 0.0262 |

### Problem 3

|  |  |  |
| --- | --- | --- |
| No. | 1 | 2 |
|  | 10000000 | 10000000 |
|  | 1758.4886 | 4.7725e19 |
|  | 6904.1073 | -1.2740e18 |
|  | 1474.3077 | -5.7744e19 |
|  | 832.4859 | 4.1384e19 |
|  | 617.8326 | 1.2049e20 |
|  | 991.6018 | 3.6896e19 |
|  | 793.6226 | 5.3104e19 |
|  | 996.2960 | -4.6726e20 |
| Constraints Violation | -17.0508 | -1.4930e37 |
| Penalty Function Value | 1.9423e9 | 2.2291e81 |
| Function Value | 10136.9036 | -1.1293e19 |