EFFICIENT AND ADAPTIVE LAGRANGE-MULTIPLIER METHODS FOR CONTINUOUS NONLINEAR OPTIMIZATION

Tao Wang and Benjamin W. Wah

Department of Electrical and Computer Engineering
Coordinated Science Laboratory
University of Illinois, Urbana-Champaign
Urbana, IL 61801, USA
E-mail: {wangtao,wah}@manip.crhc.uiuc.edu
URL: http://manip.crhc.uiuc.edu

Keywords: Lagrange Multiplier method, global optimization, inequality constraint, dynamic weighting, MaxQ method.

ABSTRACT

In this paper, we address three important issues in applying Lagrangian methods to solve optimization problems with inequality constraints. First, we propose a *MaxQ* method that transforms inequality constraints to equality constraints. It overcomes divergence and oscillations that occur in the slack-variable method. Some strategies to speed up its convergence are also examined. Second, we develop a method to monitor the balance between descents in the original-variable space and ascents in the Lagrange-multiplier space in Lagrangian methods. During the search, we adjust this balance adaptively in order to improve convergence speed. Third, we introduce a nonlinear traveling trace to pull a search trajectory out of a local equilibrium point in a continuous fashion without restarting the search and without losing information already obtained in the local search. This strategy extends existing Lagrangian methods from a local search of equilibrium points to a global search. We implement these three strategies in *Novel* (Nonlinear Optimization via External Lead) and apply it to find improved solutions for a collection of benchmark problems.

1. INTRODUCTION

Many applications in engineering, decision science, as well as operations research are formulated as optimization problems. The *constrained nonlinear optimization problem* that we study in this paper takes the following form:

$$\begin{array}{ll} \textit{Minimize} & f(X) \\ \textit{subject to} & g(X) \le 0 \quad X = (x_1, \ldots, x_n) \in R^n \quad (1) \\ & h(X) = 0 \end{array}$$

where $f(X)$ is an objective function, $g(X) = [g_1(X), \cdots, g_k(X)]^T$ is a set of k inequality constraints, and $h(X) = [h_1(X), \cdots, h_m(X)]^T$ is a set of m equality constraints. All $f(X)$, $g(X)$, and $h(X)$ are assumed to be differentiable real-valued nonlinear continuous functions.

Active research [4, 3] in the past has produced transformational and non-transformational methods to solve (1). *Non-transformational approaches* include discarding, enumerative, and back-to-feasible-region methods. They have difficulties dealing with nonlinear objective and constraints. *Transformational approaches*, on the other hand, transform a problem into another form before solving it. Some well-known methods include penalty, barrier, and Lagrangian methods. The first two methods have difficulties when they start from an infeasible region or when feasible solutions are hard to find.

Lagrange-multiplier methods (or Lagrangian methods) introduce Lagrange multiplier variables to gradually resolve constraints through iterative updates. They are exact methods that optimize an objective using Lagrange multipliers to meet the Kuhn-Tucker conditions [4]. In view of their advantages, we use them to handle constraints in this paper.

2. HANDLING INEQUALITY CONSTRAINTS

Lagrangian methods work well with equality constraints, but cannot deal directly with inequality constraints (1), except in some simple cases in which one can directly solve the first-order condition in closed form. In general, inequality constraints are first transformed into equivalent equality constraints before Lagrangian methods are applied.

Research supported in part by National Science Foundation Grants MIP 92-18715 and MIP 96-32316.

2.1. Transformation Using Slack Variables

One possible transformation [4] to handle inequality constraint $g_i(X) \leq 0$ is to add a slack variable z_i, which results in an equality constraint $g_i(X) + z_i^2 = 0$. After simplification [4], the augmented Lagrangian function for (1) is

$$L_z(X, \lambda, \mu) = f(x) + \lambda^T h(X) + ||h(X)||_2^2 + \frac{1}{2} \sum_{i=1}^{k} \left[max^2(0, \mu_i + g_i(X)) - \mu_i^2 \right] \quad (2)$$

where λ and μ are Lagrange multipliers, which control the balance between descents in the X space and ascents in the (λ, μ) space. They also play a role in controlling the convergence speed and solution quality of the Lagrangian method indirectly. At an equilibrium point, the forces due to descents and ascents reach a balance through appropriate Lagrange-multiplier values.

To emphasize how the relative weights affect the convergence speed and solution quality, we introduce an additional weight w into (2) and get:

$$L_o(X, \lambda, \mu) = w\ f(x) + \lambda^T h(X) + ||h(X)||_2^2 + \frac{1}{2} \sum_{i=1}^{k} \left[max^2(0, \mu_i + g_i(X)) - \mu_i^2 \right] \quad (3)$$

where $w > 0$ is a weight on the objective. When $w = 1$, $L_o(X, \lambda, \mu) = L_z(X, \lambda, \mu)$, which is the original Lagrangian function.

Starting from an initial point $(X(0), \lambda(0), \mu(0))$, we can solve (3) by using *LSODE*[1] and observe a search trajectory $(X(t), \lambda(t), \mu(t))$. When an equilibrium point is on the boundary of the feasible region, the dynamic equation approaches it from both inside and outside of the feasible region. We observe three behaviors of the search trajectory: (a) The trajectory gradually reduces its oscillations and eventually converges. (b) The trajectory oscillates within some range but never converges. (c) The magnitude of oscillations increases, and the trajectory eventually diverges.

To illustrate these three behaviors, consider Problem 2.3 in [2]. We set the initial point at $t = 0$ as follows: $X(t = 0)$ is set at the middle of the search space, and $\lambda(t = 0) = \mu(t = 0) = 0$. The total time used by *LSODE* is $t_{max} = 10^5$, which is divided into small units of $\Delta t = 1.0$, resulting in a maximum of 10^5 iterations $(= t_{max}/\Delta t)$. The stopping condition is the Lyupunov condition:

$$||dX(t)/dt||^2 + ||d\lambda(t)/dt||^2 + ||d\mu(t)/dt||^2 \leq \delta \quad (4)$$

where $\delta = 10^{-25}$. The system stops when it converges or when it reaches the maximum number of iterations.

[1] *LSODE* is a solver for first-order ordinary differential equations, a public-domain package available from http://www.netlib.org.

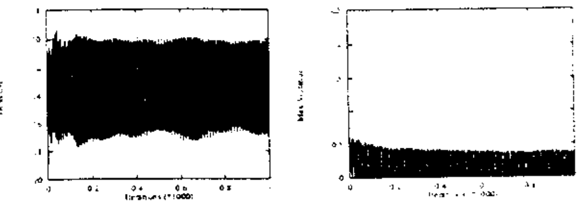

Figure 1: The objective function and maximum violation oscillate in the slack-variable method when $w = \frac{1}{10}$.

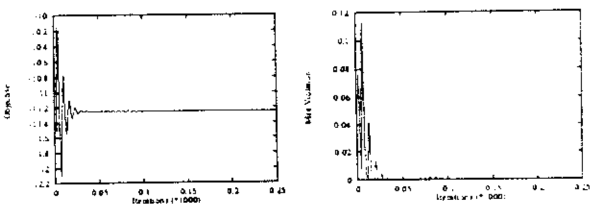

Figure 2: The objective function and maximum violation converge after oscillations subside in the slack-variable method when $w = \frac{1}{15}$.

When $w = 1$, the trajectory diverges very quickly into infinity, meaning that the original,Lagrangian method will diverge. If we scale the objective by 10 (*i.e.*, $w = 1/10$), then $f(X(t))$ oscillates within the range $[-17, -10]$, while the maximum violation $v_{max}(t)$ is between 0 and 0.4, as shown in Figure 1. Here, $v_{max}(t)$ at time t is defined as

$$v_{max}(t) = \max\{|h_i(X(t))|, \max[0, g_j(X(t))]\} \quad (5)$$

If we reduce w to 1/15, then the oscillations subside, and the trajectory eventually converges (see Figure 2).

2.2. Transformation Using the MaxQ Method

To avoid oscillations in the method based on slack variables, we would like a search to converge to a local minimum directly when it is on the boundary of a feasible region and when the trajectory is outside the feasible region. This is done by our proposed *MaxQ* method.

Without loss of generality, we omit equality constraints in the following discussion for simplicity, knowing that the equality constraints are handled in the way described in Section 1. The *MaxQ* method transforms an inequality constraint as follows:

$$g_i(X) \leq 0 \iff p_i(X) = max^{q_i}(0, g_i(X)) = 0 \quad (6)$$

where the q_i's are control parameters. The corresponding augmented Lagrangian function is:

$$L_q(X, \mu) = f(X) + \mu^T p(X) + ||p(X)||_2^2 \quad (7)$$

where $p(X) = [p_1(X), p_2(X), \cdots, p_k(X)]^T$.

It is important to choose suitable control parameters q_i, $i = 1, \cdots, k$, because they are related to the convergence speed of our algorithm. One way of selecting q_i is to make it very close to 1, *i.e.*, $q_i \rightarrow 1$. At this time, the system will approach a feasible region slowly since $p_i'(X) \simeq 1$ if $g_i(X) > 0$, which is independent of how far the current point X is away from the feasible region. Thus, larger q_i's are needed for faster convergence if the current point X is far from the feasible region. In contrast, if we choose $q_i \gg 1$, then $p_i'(X) \simeq 0$ as $g_i(X) \rightarrow 0$, meaning that *LSODE* converges very slowly towards the equilibrium point on the boundary of the feasible region.

Taking these facts into account, in order to have fast convergence, we adapt q_i dynamically as the search goes to an equilibrium point. q_i is large if $g_i(X) \gg 0$, and q_i is gradually reduced to a value approaching 1 when the search is close to the equilibrium point. One possible choice of q_i is:

$$q_i(g_i(X)) = \frac{s_0}{1 + exp(-s_1 g_i(X))} \qquad (8)$$

where $s_0 = 2$ and $s_1 > 0$ are two parameters that control the shape of function $q_i(x)$. When $g_i(X)$ approaches 0, q_i will approach 1.

Note that when $s_0 = 2$, $\nabla_X p_i(X)$ changes very fast from 1 to 0 near the equilibrium point as $g_i(X) \longrightarrow 0$, making it difficult for *LSODE* to find a suitable step size to reach the equilibrium point. To let the gradient change smoothly, we set s_0 to 2.5 or 3.0, and s_1 to satisfy $q_i(g_i(X)) = 2$ when $g_i(X) = 1$. Thus, $s_1 = -Ln[s_0/2 - 1]$.

When equilibrium point X^* is within the feasible region, *i.e.*, $g_i(X^*) < 0$, $p_i(X^*) = 0$, and $\nabla_X f(X^*) = 0$, meaning that X^* is an equilibrium point of the dynamic system given by,

$$\frac{d}{dt}X(t) = 0 \text{ and } \frac{d}{dt}\mu_i(t) = 0, \quad i = 1, \cdots, k. \qquad (9)$$

Thus the trajectory will converge to this equilibrium point X^*.

When equilibrium point X^* is on the boundary of the feasible region, it will be asymptotically approached from outside the feasible region. This can be proved by showing that X^* is asymptotically a regular point of the constraints $p_i(X) = 0$ [4] because inequality constraint $g_i(X) \leq 0$ has been equivalently transformed into equality constraint $p_i(X) = 0$.

To improve convergence rate, we dynamically convert inequality constraints that are very close to the boundary into equality constraints. Since we solve the dynamic system using *LSODE*, let X and X_0 be the points of two successive iterations. The conversion of inequality constraint $g_j(X) \leq 0$ occurs when the following two conditions are satisfied: (a) The dynamic system converges to some point X when it changes very little for γ iterations. (b) The dynamic system converges to the boundary when $g_j(X)$ is very close to zero.

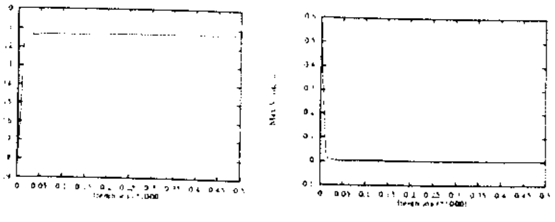
Figure 3: The objective and maximal violation converge in *MaxQ* with $w = \frac{1}{10}$.

If dynamic conversion is performed on inequality constraint $g_j(X) \leq 0$, then the terrain of the Lagrangian function $L_q(X, \mu)$ will be changed and totally different. To maintain the search direction in the original variable space X, we have to adjust Lagrange multiplier μ_j. Let the current point be (X, μ_j) just before the conversion, and $(X, \hat{\mu}_j)$ be the point after the conversion. The relationship between them is then derived as

$$\hat{\mu}_j = \left[\mu_j g_j^{q_j - 1}(X) + 2g_j^{2q_j - 1}(X)\right] \\ [q_j + q_j'(X) g_j(X) Ln\ g_j(X)] - 2g_j(X) \quad (10)$$

As in the slack-variable method, we add an extra weight w in the original augmented Lagrangian function in *MaxQ*,

$$L_m(X, \lambda, \mu) = w\ f(x) + \lambda^T h(X) + ||h(X)||_2^2 \\ + \mu^T p(X) + ||p(X)||_2^2 \qquad (11)$$

where $w > 0$ is a weight on the objective, λ is the Lagrange multiplier for equality constraints, μ for inequality constraints, and $p(X) = [p_1(X), p_2(X), \cdots, p_k(X)]^T$.

To show how *MaxQ* avoids divergence and oscillations in the slack-variable method, consider the same problem 2.3 in [2]. The starting point is at the middle of the search space, which is the same as that used in the slack-variable method. Three cases were tested: no scaling, scaling the objective by 10 (*i.e.*, $w = 1/10$), and scaling the objective by 15. Our results show that all three cases converge with similar behavior, and the second case is shown in Figure 3. Obviously, *MaxQ* has smoother and better convergence as compared to the slack-variable method.

3. ADAPTIVE LAGRANGIAN METHOD

In the last section, we have studied two methods to deal with inequality constraints. The slack-variable method is sensitive to the relative weight between the objective and the constraints, leading to divergence, oscillations, or convergence. Although *MaxQ* is not as sensitive to this weight, careful weighting may help accelerate its convergence speed. Because the relative weight for fast convergence is problem-dependent, it is impossible to set it in advance. Here, we propose a strategy to adapt this weight based on the behavior of the search progress.

3.1. Dynamic Weight-Adaptation Strategy for *MaxQ*

The basic idea is to monitor the progress of the search trajectory and adapt weight w to the search progress. We divide search time into non-overlapping windows of size N_w iterations each. In each window, we compute some metrics to measure the progress of the search relative to that of previous windows. For the t^{th} window ($t = 1, 2, \cdots$), we calculate the average value of $v_{max}(t)$ over all the iterations in this window.

$$\bar{v}_t = \frac{1}{N_w} \sum_{j=(t-1)N_w+1}^{tN_w} v_{max}(j) \tag{12}$$

At the end of each window or every N_w iterations, we decide whether to update w based on the performance metrics (12). In our current implementation, we use the average value of maximum violation $v_{max}(t)$. In general, other application-specific metrics can also be used. Based on these measurements, we adjust w accordingly.

As explained before, the major problem with the *MaxQ* sometimes is its slow convergence, which can be measured by how fast its maximal violation decreases. Therefore, we monitor the reduction of the average value $\bar{v}_t$ of the maximal violation. If it is found to decrease slowly, *i.e.*, $\bar{v}_{t-1} - \bar{v}_t \leq \beta \bar{v}_{t-1}$, where β is a threshold (e.g. $\beta = 10\%$), we will reduce weight w by half, $w \Longleftarrow w/2$. Its effect is to put more weight on the constraints, thus pushing the trajectory more quickly to a feasible region. Note that when comparing the values between two successive windows $t-1$ and t, both must use the same weight w; otherwise, the comparison is not meaningful because the terrain may be different. Hence, after adapting w, we should wait at least two windows before changing it again.

3.2. Illustration of Weight Adaptation for *MaxQ*

To use the dynamic weight-adaptation method, we set the time interval $\triangle t = 1$ for *LSODE*, and the window size $N_w = 10$. Corresponding to Figure 3, we start from the initial weight $w(t=0) = 1/10$ and the same starting point $(X(0), \lambda(0), \mu(0))$. Figure 4 shows the resulting search profile, in which the search converges using only 756 iterations. This is a significant improvement over the case without dynamic weight adaptation.

It is noted that the solution quality in Figure 4 is the same as that in Figure 3 in the sense that both obtain the objective value -11.25 when the search converges.

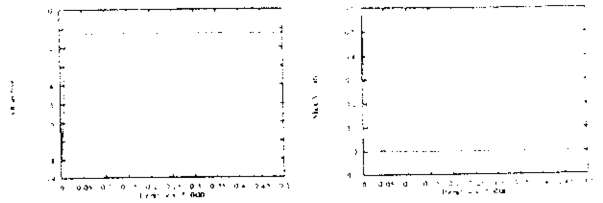

Figure 4: The objective and maximal violation converge using dynamic weight adaptation in *MaxQ*.

4. GLOBAL SEARCH

Lagrangian method is a local search method that is easy to get stuck in a local minimum. To tackle this problem, many global search algorithms have been developed that focus on bringing the search out of local minima. They use either deterministic [6, 3] or probability heuristics [5] or sampling [6]. These methods are normally computational expensive and have difficulties to deal with large problems.

We have developed *Novel* [7], a hybrid global and local search method for both constrained and unconstrained global optimization. It is a trajectory-based method that uses an *external* force to pull the search out of local minima, while using local search to locate local minima. *Novel* has three components: exploring the search space, locating promising regions and finding local minima. In exploring the search space, the search is guided by a continuous terrain-independent *trace* that does not get trapped in local minima. In locating promising regions, *Novel* uses local gradients to attract the search to a local minimum, while at the same time relying on the trace to pull it out once little improvement can be found. Finally, *Novel* selects initial points from regions that may contain promising local minima, and uses them as starting points for a local search algorithm to find local minima.

Combining these three components, *Novel* redefines the overall dynamic system for nonlinear constrained optimization as follows:

$$\begin{aligned}
\frac{d}{dt}X(t) &= -\eta_l \nabla_X L_b(X(t), \lambda(t), \mu(t)) \\
&\quad - \eta_g * (X(t) - T_X(t)) \\
\frac{d}{dt}\lambda(t) &= \nabla_\lambda L_b(X(t), \lambda(t), \mu(t)) \\
\frac{d}{dt}\mu(t) &= \nabla_\mu L_b(X(t), \lambda(t), \mu(t))
\end{aligned} \tag{13}$$

where η_l and η_g are constants controlling the relative weights between local search and global exploration.

The dynamic system is used in both the global and the local search. There are three stages in the global search phase, each of which outputs a trajectory based on (13). In the first stage of the global search, the user-defined trace function $T_X(t)$ leads the trajectory. In the second and third stages of the global search, the trace function $T_X(t)$ in (13) is defined as the trajectory obtained in the previous stage. According to the trajectory output from the three stages, we identify a set of promising starting points and perform local Lagrangian searches from them. The final result is the best solution among all these Lagrangian searches.

5. EXPERIMENTAL RESULTS

In this section we describe our experimental results on some existing benchmarks [2]. These benchmarks are challenging because they model practical applications that have been studied extensively in the past.

We use the same set of parameters to solve all the problems, except that the search range is problem-specific. In practice, this is reasonable as search ranges are generally known. In case that the search range is not available, we use trial and error, starting from a small range and gradually increasing it until no improvement is found in the solutions.

In the global search phase of *Novel*, we use three stages that produce three trajectories. From each trajectory, we choose 100 starting points for Lagrangian search based on their Lagrangian-function values. After 300 searches, we report the best solution found.

Table 1 summarizes the results found by *Novel*. Column 1 lists the problem identifications that appear in the benchmark collection [2]. Column 2 shows the best known solutions reported in [2], and Column 3, the solutions reported by Epperly [1]. Here, symbol '–' means that the method is unable to find a solution for the corresponding problem. Column 4 shows the results obtained by *Novel* using the slack-variable method with dynamic weight adaptation [8]. Without weight adaptation, more than half of these problems cannot be solved due to divergence and oscillations described in Section 2.1. The last column shows the results obtained by *Novel* using *MaxQ*. Results in bold font are improved by *Novel* over the best known results, with improvements of up to 10%. Our results indicate that *Novel* is robust in discovering new regions and in escaping from local traps.

Table 1: Results on a collection of constrained optimization benchmarks [2] comparing *Novel* using *MaxQ*, *Novel* using the slack-variable method, and Epperly's method [1]. Improved solutions found by *MaxQ* are indicated in bold font. Symbol '–' means that the method was not able to find a solution for the corresponding problem.

Problem ID	Best Known Solutions	Epperly's Solutions	Slack Variable Solutions	MaxQ Solutions
2.1	−17.00	−17.00	−17.00	−17.00
2.2	−213.00	−213.00	−213.00	−213.00
2.3	−15.00	−15.00	−15.00	−15.00
2.4	−11.00	−11.00	−11.00	−11.00
2.5	−268.00	−268.00	−268.00	−268.00
2.6	−39.00	−39.00	−39.00	−39.00
2.7(1)	−394.75	−394.75	−394.75	−394.75
2.7(2)	−884.75	−884.75	−884.75	−884.75
2.7(3)	−8695.00	−8695.00	−8695.00	−8695.00
2.7(4)	−754.75	−754.75	−754.75	−754.75
2.7(5)	−4150.40	−4150.40	−4150.40	−4150.40
2.8	15990.00	15990.00	**15639.00**	**15639.00**
3.1	7049.25	–	7049.25	7049.25
3.2	−30665.50	−30665.50	−30665.50	−30665.50
3.3	−310.00	−310.00	−310.00	−310.00
3.4	−4.00	−4.00	−4.00	−4.00
4.3	−4.51	−4.51	−4.51	−4.51
4.4	−2.217	−2.217	−2.217	−2.217
4.5	−11.96	**−13.40**	**−13.40**	**−13.40**
4.6	−5.51	−5.51	−5.51	−5.51
4.7	−16.74	−16.74	−16.75	−16.75
5.2	1.567	–	1.567	1.567
5.4	1.86	–	1.86	1.86
6.2	400.00	400.00	400.00	400.00
6.3	600.00	600.00	600.00	600.00
6.4	750.00	750.00	750.00	750.00
7.2	56825.00	–	56825.00	56825.00
7.3	46266.00	–	46266.00	**44903.00**
7.4	35920.00	–	35920.00	35920.00

6. REFERENCES

[1] T. Epperly. *Global Optimization of Nonconvex Nonolinear Programs Using Parallel Branch And Bound*. PhD thesis, University of Wisconsin-Madison, 1995.

[2] C. A. Floudas and P. M. Pardalos. *A Collection of Test Problems for Constrained Global Optimization Algorithms*, volume 455 of *Lecture Notes in Computer Science*. Springer-Verlag, 1990.

[3] R. Horst and H. Tuy. *Global optimization: Deterministic approaches*. Springer-Verlag, Berlin, 1993.

[4] D. G. Luenberger. *Linear and Nonlinear Programming*. Addison-Wesley Publishing Company, 1984.

[5] H. E. Romeijn and R. L. Smith. Simulated annealing for constrained global optimization. *Journal of Global Optimization*, 5(2):101–126, September 1994.

[6] F. Schoen. Stochastic techniques for global optimization: A survey on recent advances. *Journal of Global Optimization*, 1(3):207–228, 1991.

[7] B. W. Wah and Y.-J. Chang. Trace-based methods for solving nonlinear global optimization problems. *J. of Global Optimization*, 10(2):107–141, March 1997.

[8] B. W. Wah, T. Wang, Y. Shang, and Z. Wu. Improving the performance of weighted Lagrange-multiple methods for constrained nonlinear optimization. In *Proc. 9th Int'l Conf. on Tools for Artificial Intelligence*, pages 224–231. IEEE, November 1997.