African Journal of Mathematics and Computer Science Research Vol. 5(13), pp. 209-246, November, 2012
Available online at http://www.academicjournals.org/AJMCSR
DOI: 10.5897/AJMCSR12.027
ISSN 2006-9731 ©2012 Academic Journals

Full Length Research Paper

Penalty function methods using matrix laboratory (MATLAB)

Hailay Weldegiorgis Berhe

Department of Mathematics, Haramaya University, Ethiopia. E-mail: hailaywg@yahoo.com. Tel: +251913710270.

Accepted 15 May, 2012

The purpose of the study was to investigate how effectively the penalty function methods are able to solve constrained optimization problems. The approach in these methods is to transform the constrained optimization problem into an equivalent unconstrained problem and solved using one of the algorithms for unconstrained optimization problems. Algorithms and matrix laboratory (MATLAB) codes are developed using Powell's method for unconstrained optimization problems and then problems that have appeared frequently in the optimization literature, which have been solved using different techniques compared with other algorithms. It is found out in the research that the sequential transformation methods converge to at least to a local minimum in most cases without the need for the convexity assumptions and with no requirement for differentiability of the objective and constraint functions. For problems of non-convex functions it is recommended to solve the problem with different starting points, penalty parameters and penalty multipliers and take the best solution. But on the other hand for the exact penalty methods convexity assumptions and second-order sufficiency conditions for a local minimum is needed for the solution of unconstrained optimization problem to converge to the solution of the original problem with a finite penalty parameter. In these methods a single application of an unconstrained minimization technique as against the sequential methods is used to solve the constrained optimization problem.

Key words: Penalty function, penalty parameter, augmented lagrangian penalty function, exact penalty function, unconstrained representation of the primal problem.

INTRODUCTION

Optimization is the act of obtaining the best result under given circumstances. In design, construction and maintenance of any engineering system, engineers have to take many technological and managerial decisions at several stages. The ultimate goal of all such decisions is either to minimize the effort required or to maximize the desired benefit. Since the effort required or the benefit desired in any practical situation can be expressed as a function of certain decision variables, optimization can be defined as the process of finding the conditions that give the maximum or minimum value of a function. It can be taken to mean minimization since the maximum of a function can be found by seeking the minimum of the negative of the same function.

Optimization can be of constrained or unconstrained problems. The presence of constraints in a nonlinear programming creates more problems while finding the minimum as compared to unconstrained ones. Several situations can be identified depending on the effect of constraints on the objective function. The simplest situation is when the constraints do not have any influence on the minimum point. Here the constrained minimum of the problem is the same as the unconstrained minimum, that is, the constraints do not have any influence on the objective function. For simple optimization problems it may be possible to determine before hand, whether or not the constraints have any influence on the minimum point. However, in most of the practical problems, it will be extremely difficult to identify it. Thus, one has to proceed with general assumption that the constraints will have some influence on the optimum point. The minimum of a nonlinear programming problem will not be, in general, an extreme point of the feasible region and may not even be on the boundary. Also, the

problem may have local minima even if the corresponding unconstrained problem is not having local minima. Furthermore, none of the local minima may correspond to the global minimum of the unconstrained problem. All these characteristics are direct consequences of the introduction of constraints and hence, we should have general algorithms to overcome these kinds of minimization problems.

The algorithms for minimization are iterative procedures that require starting values of the design variable x. If the objective function has several local minima, the initial choice of x determines which of these will be computed. There is no guaranteed way of finding the global optimal point. One suggested procedure is to make several computers run using different starting points and pick the best. Majority of available methods are designed for unconstrained optimization where no restrictions are placed on the design variables. In these problems, the minima exist if they are stationary points (points where gradient vector of the objective function vanishes). There are also special algorithms for constrained optimization problems, but they are not easily accessible due to their complexity and specialization.

All of the many methods available for the solution of a constrained nonlinear programming problem can be classified into two broad categories, namely, the direct methods and the indirect methods approach. In the direct methods the constraints are handled in an explicit manner whereas in the most of the indirect methods, the constrained problem is solved as a sequence of unconstrained minimization problems or as a single unconstrained minimization problem. Here we are concerned on the indirect methods of solving constrained optimization problems. A large number of methods and their variations are available in the literature for solving constrained optimization problems using indirect methods. As is frequently the case with nonlinear problems, there is no single method that is clearly better than the others. Each method has its own strengths and weaknesses. The quest for a general method that works effectively for all types of problems continues. The main purpose of this research is to present the development of two methods that are generally considered for solving constrained optimization problems, the sequential transformation methods and the exact transformation methods.

Sequential transformation methods are the oldest methods also known as sequential unconstrained minimization techniques (SUMT) based upon the work of Fiacco and McCormick (1968). They are still among the most popular ones for some cases of problems, although there are some modifications that are more often used.

These methods help us to remove a set of complicating constraints of an optimization problem and give us a frame work to exploit any available methods for unconstrained optimization problems to be solved, perhaps, approximately. However, this is not without a cost. In fact, this transforms the problem into a problem of non smooth (in most cases) optimization, which has to be solved iteratively. The sequential transformation method is also called the classical approach and is perhaps the simplest to implement. Basically, there are two alternative approaches. The first is called the exterior penalty function method (commonly called penalty function method), in which a penalty term is added to the objective function for any violation of constraints. This method generates a sequence of infeasible points, hence its name, whose limit is an optimal solution to the original problem. The second method is called interior penalty function method (commonly called barrier function method), in which a barrier term that prevents the points generated from leaving the feasible region is added to the objective function. The method generates a sequence of feasible points whose limit is an optimal solution to the original problem.

Penalty function methods are procedures for approximating constrained optimization problems by unconstrained problems. The approximation is accomplished by adding to the objective function a term that prescribes a high cost for the violation of the constraints. Associated with this method is a parameter μ that determines the severity of the penalty and consequently the degree to which the unconstrained problem approximates the original problem. As $\mu \to \infty$ the approximation becomes increasingly accurate.

Thus, there are two fundamental issues associated with this method. The first has to do how well the unconstrained problem approximates the constrained one. This is essential in examining whether, as the parameter μ is increased towards infinity, the solution of the unconstrained problem converges to a solution of the constrained problem. The other issue, most important from a practical view point, is the question of how to solve a given unconstrained problem when its objective function contains a penalty term. It turns out that as μ is increased yields a good approximating problem; the corresponding structure of the resulting unconstrained problem becomes increasingly unfavorable thereby slowing the convergence rate of many algorithms that may be applied. Therefore it is necessary to device acceleration procedures that circumvent this slow convergence phenomenon. To motivate the idea of penalty function methods consider the following nonlinear programming problem with only inequality constraints:

Minimize f(x), subject to $g(x) \leq 0$ (P) $x \in X$;

Whose feasible region we denote by $S = \{\mathbf{x} \in \mathbf{X} \mid \mathbf{g}(\mathbf{x}) \leq 0\}$. Functions f: $R^n \to R$ and g: $R^n \to R^m$ are assumed to be continuously differentiable and X is a nonempty set in R^n. Let the set of minimum points of problem (P) be denoted by M (f, S), where $M(f, S) \neq \emptyset$. And we consider a real sequence $\{\mu_k\}$ such that $\mu_k \geq 0$. The number μ_k is called penalty parameter, which controls the degree of penalty for violating the constraints. Now we consider functions $\theta: X \times (R_+ \cup \{0\}) \to R$, as defined by

$$\theta(x, \mu) := f(x) + \mu p(x), (x, \mu) \in \{(X) \times (R_+ \cup \{0\})\}, \quad (1.1)$$

where $X \subseteq R^n$ and p(x) is called penalty function, to be used throughout this paper, and μp(x) is called penalty term. μ is a strictly increasing function. Throughout this paper we use penalty function methods for exterior penalty function methods

Another apparently attractive idea is to define an exact penalty function in which the minimizer of the penalty function and the solution of the constrained primal problem coincide. The idea in these methods is to choose a penalty function and a constant penalty parameter so that the optimal solution of the unconstrained problem is also a solution of the original problem. This avoids the inefficiency inherent in sequential techniques. The two popular exact penalty functions are l_1 exact penalty function and augmented Lagrangian penalty function.

More emphasis is given here for sequential transformation methods and practical examples, which appeared frequently in the optimization literature (which have been solved using different methods.) and facility locations are solved using the MATLAB code given in the appendix with special emphasis given to facility location problems.

Some important techniques in the approach of the primal problem and the corresponding unconstrained penalty problems will be discussed later. We also discuss properties of the penalty problem, convergence conditions and the structure of the Hessian objective function of the penalty problem and the methods for solving unconstrained problems; the general description of the algorithm for the penalty function problems in addition to the considerations for the implementation of the method. A major challenge in the penalty function methods is the ill-conditioning of the Hessian matrix of objective function as the penalty parameter approaches to infinity, the choice of the initial starting points, penalty parameters and subsequent values of the penalty parameters.

In the later part of this study, the exact penalty methods, the exact l_1 penalty function and the augmented Lagrangian penalty function methods will be discussed in detail. The sequential methods suffer from numerical difficulties in solving the unconstrained problem. Furthermore, the solution of the unconstrained problem approaches the solution of the original problem in the limit, but is never actually equal to the exact solution. To overcome these shortcomings, the so-called exact penalty functions have been developed.

Statement of the problem

The focus of this research paper is on investigating constraint handling to solve constrained optimization problems using penalty function methods and thereby indicating ways of revitalizing them by bringing to attention.

Objectives of the research

General objective

The purpose of the research is generally to see how the penalty methods are successful to solve constrained optimization problems.

Specific objectives

The specific objectives of the research are to:

i. Describe the essence of penalty function methods,
ii. Clearly identify the procedures in solving constrained optimization problems using penalty function methods,
iii. Develop an algorithm and MATLAB code for penalty function methods,
iv. Solve real life application problems which frequently appeared in the optimization literature and facility location problems using the investigated code and compare with other methods,
v. Compare the effectiveness of the penalty methods.

Significance of the research

i. All algorithms for constrained optimization are unreliable to a degree. Any one of them works well on one problem and fails to another. Thus, this work will be having its own contribution in bridging the gap,
ii. It will also pave way and serves as an eye opener to other researchers to carry out an extensive and/or detail study along the same or other related issue.

PENALTY FUNCTION METHODS

In this section, we are concerned with exploring the computational properties of penalty function methods. We present and prove an important result that justifies using penalty function methods as a means for solving constrained optimization problems. We also discuss some computational difficulties associated with these methods and present some techniques that should be used to overcome such difficulties. Using the special structure of the penalty function, a special purpose one-dimensional search procedure algorithm is developed. The procedure is based on Powell's method for unconstrained minimization technique together with bracketing and golden section for one dimensional search.

When solving a general nonlinear programming problem in which the constraints cannot easily be eliminated, it is necessary to balance the aims of reducing the objective function and staying inside or close to the feasible region, in order to induce global convergence (that is convergence to a local solution from

any initial approximation). This inevitably leads to the idea of a penalty function, which is a combination of the some constraints that enables the objective function to be minimized whilst controlling constraint violations (or near constraint violations) by penalizing them. The philosophy of penalty methods is simple; you give a "fine" for violating the constraints and obtain approximate solutions to your original problem by balancing the objective function and a penalty term involving the constraints. By increasing the penalty, the approximate solution is forced to approach the feasible domain and hopefully, the solution of the original constrained problem. Early penalty functions were smooth so as to enable efficient techniques for smooth unconstrained optimization to be used.

The use of penalty functions to solve constrained optimization problems is generally attributed to Courant. He introduced the earliest penalty function with equality constraint in 1943. Subsequently, Pietrgykowski (1969) discussed this approach to solve nonlinear problems. However, significant progress in solving practical problems by use of penalty function methods follows the classic work of Fiacco and McCormick under the title sequential unconstrained minimization technique (SUMT). The numerical problem of how to change the parameters of the penalty functions have been investigated by several authors.

Fiacco and McCormick (1968) and Himmelblau (1972) discussed effective unconstrained optimization algorithms for solving penalty function methods. According to Fletcher, several extensions to the concepts of penalty functions have been made; first, in order to avoid the difficulties associated with the ill-conditioning as the penalty parameter approaches to infinity, several parameter-free methods have been proposed. We will discuss some of the effective techniques in reducing these difficulties in the following sections.

The concept of penalty functions

Consider the following problem with single constraint h(**x**) = 0:

Minimize f(x),
subject to h(x) = 0.

Suppose this problem is replaced by the following unconstrained problem, where μ > 0 is a large number:

Minimize $f(x) + \mu h^2(x)$,
subject to $x \in R^n$

we can intuitively see that an optimal solution to the above problem must have $h^2(x)$ close to zero, otherwise a large penalty term $\mu h^2(x)$ will be incurred and hence $f(x) + \mu h^2(x)$ approaches to infinity which makes it difficult to minimize the unconstrained problem (Bazaraa et al. (2006).

Now consider the following problem with single inequality constraint $g(x) \leq 0$:

Minimize f(x),
subject to $g(x) \leq 0$.

It is clear that the form $f(x) + \mu g^2(x)$ is not appropriate, since a penalty will be incurred where $g(x) < 0$ or $g(x) > 0$; that is a penalty is added to the objective function whether x is inside or outside the feasible region. Needless to say, a penalty is desired only if the point is not feasible, that is, if $g(x) > 0$. A suitable unconstrained problem is therefore given by:

Minimize f(x) + μmaximum {0, g(x)},
subject to $x \in R^n$.

Note that if $g(x) \leq 0$, then maximum {0, g(x)} = 0, and no penalty is incurred on the other hand, if $g(x) > 0$, then maximum {0, g(x)} > 0, and the penalty term μg(x) is realized. However, it is observe that at points x where g(x) = 0, the forgoing objective function might not be differentiable, even though g is differentiable.

Example 1

Minimize x,
subject to $-x + 2 \leq 0$,

the constraint, $g(x) = -x + 2 \leq 0$ is active at x = 2 and the corresponding forgoing objective function is:

$$f(x) + \mu\text{maximum } \{0, g(x)\} = \begin{cases} x + \mu(-x+2), \text{if } x < 2 \\ x, \text{if } x \geq 2. \end{cases}$$

Clearly this is not differentiable at x = 2. If differentiability is desirable in such cases, then one could, for example, consider instead a penalty function term of the type $\mu(\text{maximum}\{0, g(x)\})^2$.

In general, a suitable penalty function must incur a positive penalty for infeasible points and no penalty for feasible points. If the constraints are of the form $g_i(x) \leq 0$ for i = 1. . . m, then a suitable penalty function p is defined by

$$p(x) = \sum_{i=1}^{m} \Phi(g_i(\mathbf{x})), \quad (2.1a)$$

where:

Φ is a continuous function satisfying the following properties:

$\Phi(y) = 0$ if $y \leq 0$ and $\Phi(y) > 0$ if $y > 0$. (2.1b)

Typically Φ is of the form

$\Phi(y) = (\text{maximum } \{0, y\})^q$,

where q is a nonnegative real number. Thus, the penalty function p is usually of the form p(**x**) = $\sum_{i=1}^{m}(\text{maximum}\{0, g_i(\mathbf{x})\})^q$.

Definition 1. A function p : $R^n \rightarrow R$ is called a penalty function if p satisfies

i. p(x) is continuous on R^n
ii. p(x) = 0 if g(x) $\leq$ 0 and
iii. p(x) > 0 if g(x) $>$ 0.

An often-used class of penalty functions for optimization problems with only inequality constraints is:

p(**x**) = $\sum_{i=1}^{m}(\text{maximum}\{0, g_i(\mathbf{x})\})^q$, where q ≥ 1

We refer to the function f(x) + μp(x) as the auxiliary function. Denoting θ(x, μ): = f(x) + $\mu\sum_{i=1}^{m}(\text{maximum}\{0, g_i(\mathbf{x})\})^q$, for the auxiliary function. The effect of the second term on the right side is to increase θ(x, μ) in proportion to the qth power of the amount by which the constraints are violated. Thus there will be a penalty for violating the constraints and the amount of penalty will increase at a faster rate compared to the amount of violation of a constraint (for q $>$ 1) Rao (2009). Let us see the behavior of θ(x, μ) for various values of q.

i. q = 0,

$\theta(x, \mu) = f(x) + \mu\sum_{i=1}^{m}(\text{maximum}\{0, g_i(\mathbf{x})\})^0$

$$= \begin{cases} f(\mathbf{x}) + m\mu, \text{ if all } g_i(\mathbf{x}) > 0 \\ f(\mathbf{x}), \text{ if all } g_i(\mathbf{x}) \leq 0. \end{cases}$$

This function is discontinuous on the boundary of the acceptable region and hence it will be very difficult to minimize this function.

ii. $0 < q < 1$

Here the θ-function will be continuous, but the penalty for violating a constraint may be too small. Also the derivatives of the function are discontinuous along the boundary. Thus, it will be difficult to minimize the θ-function.

iii. q = 1

In this case, under certain restrictions, it has been shown that there exists a μ_o large enough that the minimum of θ is exactly the constrained minimum of the original problem for all $\mu_k \geq \mu_o$; however, the contours of the θ-function posses discontinuous first derivatives along the boundary.

Hence, in spite of the convenience of choosing a single μ_k that yields the constrained minimum in one unconstrained minimization, the method is not very attractive from computational point of view.

iv. $q > 1$

The θ-function will have continuous first derivatives. These derivatives are given by

$$\frac{d\theta}{dx_i} = \frac{df}{dx_i} + \mu\sum_{i=1}^{m} q(\text{maximum}\{0, g_i(\mathbf{x})\})^{q-1}\frac{dg_i(x)}{dx_i}.$$

Generally, the value of q is chosen as 2 in practical computations and hence, will be used as q = 2 in the subsequent discussion of the penalty method with

p(x) = $\sum_{i=1}^{m}(\text{maximum}\{0, g_i(\mathbf{x})\})^2$.

Example 2

Consider the optimization problem in example1,
Let p(x) = $(\text{maximum}\{0, g_i(\mathbf{x})\})^2$, then,

$$p(x) = \begin{cases} 0, \text{if } x \geq 0 \\ (-x+2)^2, \text{if } x < 2. \end{cases}$$

Note that the minimum of f + μp occurs at the point 2 – $(\frac{1}{2\mu})$ and approaches the minimum point x = 2 of the original problem as μ → ∞. The penalty and auxiliary functions are as shown in Figure 1.

If the constraints are of the form $g_i(\mathbf{x}) \leq 0$ for I = 1, . . . , m and $h_i(\mathbf{x}) = 0$ for I = 1, . . . , l, then a suitable penalty function p is defined by

$$p(\mathbf{x}) = \sum_{i=1}^{m}\Phi\big(g_i(\mathbf{x})\big) + \sum_{i=1}^{l}\psi\big(h_i(\mathbf{x})\big), \qquad (2.2a)$$

where Φ and ψ are continuous functions satisfying the following properties:

Φ(y) = 0 if y ≤ 0 and Φ(y) > 0 if y > 0. (2.2b)
ψ(y) = 0 if y = 0 and ψ(y) > 0 if y ≠ 0.

Typically, Φ and ψ are of the forms

Φ(y) = (maximum {0, y}) ^{q}
ψ(y) = |y| ^{q}

where q is a nonnegative real number. Thus, the penalty function p is usually of the form

p(**x**) = $\sum_{i=1}^{m}(\text{maximum}\{0, g_i(\mathbf{x})\})^q + \sum_{i=1}^{l}|h_i(\mathbf{x})|^q$.

Definition 2

A function p : $R^n \rightarrow R$ is called a penalty function if p satisfies

i. p(x) is a continuous function on R^n
ii. p(x) = 0 if g(x) $\leq$ 0 and h(x) = 0 and
iii. p(x) > 0 if g(x) $>$ 0 and h(x) $\neq$ 0.

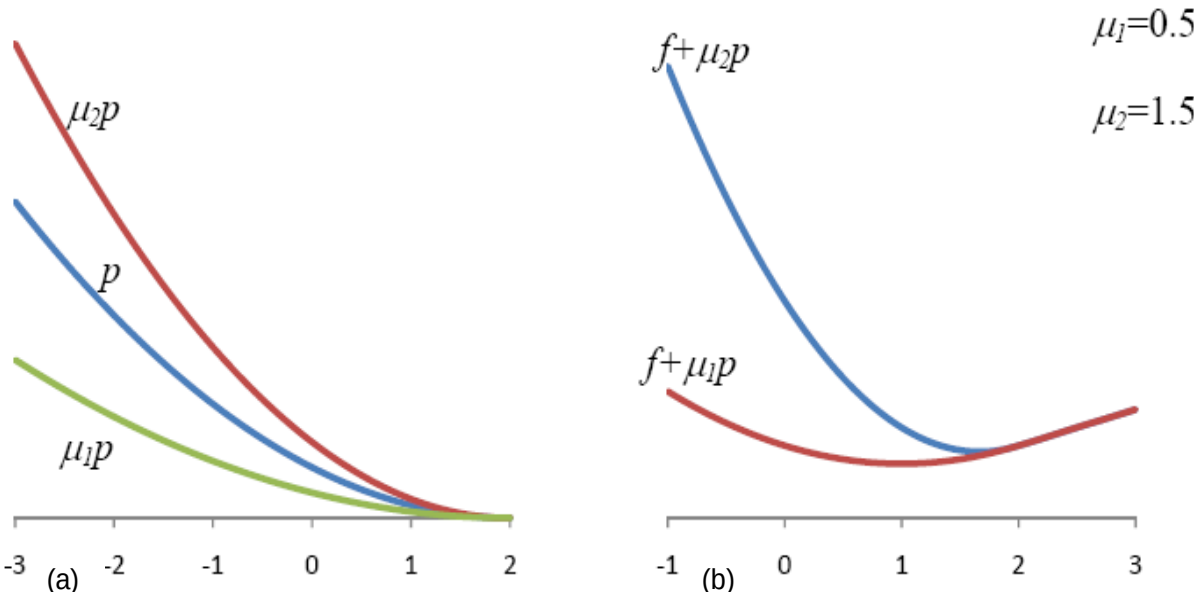

Figure 1. Penalty and auxiliary functions.

An often used class of penalty functions for this is:

$p(x) = \sum_{i=1}^{m} (\text{maximum}\{0, g_i(\mathbf{x})\})^q + \sum_{i=1}^{l} |h_i(\mathbf{x})|^q$, where $q \geq 1$.

We note the following:

If q = 1, p(x) is called "linear penalty function." This function may not be differentiable at points where $g_i(x) = 0$ or $h_i(x) = 0$ for some i.

Setting q = 2 is the most common form that is used in practice and is called the "quadratic penalty function".

We focus here mainly on the quadratic penalty function and investigate how penalty function methods are useful to solve constrained optimization problems by changing into the corresponding unconstrained optimization problems.

Penalty function methods for mixed constraints

Consider the following constrained optimization problem:

Minimize f(x) (P)
subject to $g_i(\mathbf{x}) \leq 0$, I = 1, . . . , m
$h_i(\mathbf{x}) = 0$, I = 1, . . . , l,

where functions f, $h_i(x)$, I = 1, . . . , l and g_i, I = 1, . . . , m are continuous and usually assumed to posses continuous partial derivatives on R^n. For notational simplicity, we introduce the vector-valued functions $h = (h_1, h_2, \ldots, h_l)^T \in R^l$ and $g = (g_1, g_2, \ldots, g_m)^T \in R^m$ and rewrite (P) as:

Minimize f(x)
subject to $g(x) \leq 0$
$h(x) = 0$ (P)
$x \in X$,

whose feasible region we denote by S: $= \{\mathbf{x} \in \mathbf{X}, \mathbf{g}(\mathbf{x}) \leq \mathbf{0}, \mathbf{h}(\mathbf{x}) = \mathbf{0}\}$. The constraints h(x) = 0 and g(x) ≤ 0 are referred to as functional constraints, while the constraint $x \in X$ is a set constraint. The set X is a nonempty set in R^n and might typically represent simple constraints that could be easily handled explicitly, such as lower and upper bounds on the variables. We emphasize the set constraint, assuming in most cases that either X is in the whole R^n or that the solution to (P) is in the interior of X.

By converting the constraints "$h_i(x) = 0$" to "$h_i(x) \leq 0, -h_i(\mathbf{x}) \leq 0$" or considering only problems with inequality constraints we can assume that (P) is of the form:

Minimize f(x) (P)
subject to $g(x) \leq 0$
$x \in X$,

whose feasible region we denote by S: $= \{\mathbf{x} \in \mathbf{X} \mid \mathbf{g}(\mathbf{x}) \leq 0\}$. We then consider solving the following penalty problem,

θ(μ): Minimize f(x) + μp(x)
subject to $x \in X$,

and investigate the connection between a sequence $\{\mathbf{x}_\mu\}, \mathbf{x}_\mu \in M(f, X)$, a minimum point of θ and a solution of

the original problem (P), where the set of minimum points of θ is denoted by M(f, X) and the set of all minimum points of (P) is denoted by M(f, S).

The representation of penalty methods above has assumed either that the problem (P) has no equality constraints, or that the equality constraints have been converted into inequality constraints. For the latter, the conversion is easy to do, but the conversion usually violates good judgments in that it unnecessarily complicates the problem. Furthermore, it can cause the linear independence condition to be automatically violated for every feasible solution. Therefore, instead let us consider the constrained optimization problem (P) with both inequality and equality constraints since the above can be easily verified from this. To describe penalty methods for problems with mixed constraints, we denote the penalty parameter by $l(\mu) = \mu \geq 0$, which is a monotonically increasing function and the penalty function $P(\mathbf{x}) = \sum_{i=1}^{m} \Phi(g_i(\mathbf{x})) + \sum_{i=1}^{l} \psi(h_i(\mathbf{x}))$, satisfying the properties given in (2.2b) and then consider the following Primal and Penalty problems:

Primal problem

Minimize f(x)
subject to g(x)
$\leq$ 0 h(x) = 0 (P) x ∈ X.

Penalty problem

The basic penalty function approach attempts to solve the following problem:

Maximize θ(μ)
subject to μ ≥ 0,

where θ(μ) = inf{f(x) + μp(x) : x ∈ X}. The penalty problem consists of maximizing the infimum (greatest lower bound) of the function {f(x) + μp(x): x ∈ X}; therefore, it is a max-min problem. Therefore the penalty problem can be formulated as:

Find $\sup_{\mu \geq 0} \inf_{x \in X}\{f(\mathbf{x}) + \mu p(\mathbf{x})\}$ which is equivalent to the form;

Find $\inf_{x \in X} \sup_{\mu \geq 0}\{f(\mathbf{x}) + \mu p(\mathbf{x})\}$. We remark here that, strictly speaking, we should write the penalty problem as sup{θ(μ), μ ≥ 0}, rather than maximum{θ(μ), μ ≥ 0}, since the maximum may not exist. The main theorem of this section states that:

inf{f(x) : x ∈ S} = $\sup_{\mu \geq 0} \theta(\mu) = \lim_{\mu \to \infty} \theta(\mu)$.

From this result, it is clear that we can get arbitrarily close to the optimal objective value of the original problem by computing θ(μ) for a sufficiently large μ.This result is established in Theorem 2. First, the lemma theorem is needed.

Lemma 1 (**Penalty Lemma)**

Suppose that f, $g_1, \ldots, g_m$, $h_1, \ldots, h_l$ are continuous functions on R^n, and let X be a nonempty set in R^n. Let p be a continuous function on R^n as given by definition 1, and suppose that for each μ, there exists an $x_\mu \in X$, which is a solution of θ(μ), where θ(μ) := $f(x_\mu) + \mu p(x_\mu)$.
Then, the following statements hold:

1. $p(x_\mu)$ is a non-increasing function of μ.
2. $f(x_\mu)$ is a non-decreasing function of μ.
3. θ(μ) is a non-decreasing function of μ.
4. nf{f(x) : x ∈ S} ≥ $\sup_{\mu \geq 0} \theta(\mu)$, where θ(μ) = inf{f(x) + μp(x) : x ∈ X}, and g, h are vector valued functions whose components are $g_1, g_2, \ldots, g_m$ and $h_1, h_2, \ldots, h_l$ respectively.

Proof: Assume that μ and λ are penalty parameters such that λ < μ.

1. By the definition of θ(λ), x_λ is a solution of θ(λ) such that,

θ(λ) = $f(x_\lambda) + \lambda p(x_\lambda) \leq$ inf{f(x) + λp(x), for all x ∈ X}, which follows

$f(x_\lambda) + \lambda p(x_\lambda) \leq f(x_\mu) + \lambda p(x_\mu)$, since $x_\mu \in X$. (2.3a)

Again by the definition of θ(μ)

θ(μ) = $f(x_\mu) + \mu\, p(x_\mu) \leq$ inf{f(x) + μ p(x), for all x ∈ X} which follows that

$f(x_\mu) + \mu p(x_\mu) \leq f(x_\lambda) + \mu p(x_\lambda)$, since $\mathbf{x}_\lambda \in X$. (2.3b)

Adding equation (2.3a) and (2.3b) holds:

$f(x_\lambda) + \lambda p(x_\lambda) + f(x_\mu) + \mu p(x_\mu) \leq f(x_\mu) + \lambda p(x_\mu) + f(x_\lambda) + \mu p(x_\lambda)$

and simplifying like term, we get

$\lambda p(x_\lambda) + \mu\, p(x_\mu) \leq \lambda p(x_\mu) + \mu p(x_\lambda)$,

which implies by rearranging that

$(\lambda - \mu)[p(x_\lambda) - p(x_\mu)] \leq 0$.

Since λ - μ ≤ 0 by assumption, $p(x_\lambda) - p(x_\mu) \geq 0$. Then, $p(x_\lambda) \geq p(x_\mu)$.

Therefore, $p(x_\mu)$ is a non increasing function of μ.

2. By (2.3a) above

$f(x_\lambda) + \lambda p(x_\lambda) \le f(\mathbf{x}_\mu) + \lambda p(\mathbf{x}_\mu)$.

Since $p(x_\lambda) \ge p(\mathbf{x}_\mu)$ by part 1, we concluded that

$f(x_\lambda) \le f(\mathbf{x}_\mu)$.

3. $\theta(\lambda) = f(x_\lambda) + \lambda p(x_\lambda) \le f(\mathbf{x}_\mu) + \lambda p(\mathbf{x}_\mu)$
$\le f(\mathbf{x}_\mu) + \mu p(\mathbf{x}_\mu) = \theta(\mu)$.

4. Suppose $\mathbf{x}^*$ be any feasible solution to problem (P) with

$g(\mathbf{x}^*) \le 0$, $h(\mathbf{x}^*) = 0$ and $p(\mathbf{x}^*) = 0$, where $\mathbf{x}^* \in X$. Then,
$f(\mathbf{x}^*) + \mu p(\mathbf{x}^*) = \inf\{f(x), x \in S\}$ which implies that
$f(\mathbf{x}^*) = \inf\{f(x), x \in S\}$. (2.3c)

By the definition of $\theta(\mu)$

$\theta(\mu) = f(\mathbf{x}_\mu) + \mu p(\mathbf{x}_\mu) \le f(\mathbf{x}^*) + \mu p(\mathbf{x}^*) = \inf\{f(x), x \in S\}$, for all $\mu \ge 0$.

Therefore, $\sup_{\mu \ge 0} \theta(\mu) \le \{\inf\{f(x), x \in S\}\}$.

The next result concerns convergence of the penalty method. It is assumed that f(x) is bounded below on the (nonempty) feasible region so that the minimum exists.

Theorem 2 (Penalty convergence theorem)

Consider the following Primal problem:

Minimize f(x)
subject to $g(x) \le 0$
$h(x) = 0$ (P)
$x \in X$,

where f, g, h are continuous functions on R^n and X is a nonempty set in R^n. Suppose that the problem has a feasible solution denoted by $\mathbf{x}^*$, and p is a continuous function of the form (2.2). Furthermore, suppose that for each μ, there exists a solution $\mathbf{x}_\mu \in X$ to the problem to minimize $\{f(x) + \mu p(x)$ subject to $x \in X\}$, and $\{x_\mu\}$ is contained in a compact subset X then,

$\inf\{f(x) : x \in S\} = \sup_{\mu \ge 0} \theta(\mu) = \lim_{\mu \to \infty} \theta(\mu)$,

where $\theta(\mu) = \inf\{f(x) + \mu p(x) : x \in X\} = f(\mathbf{x}_\mu) + \mu p(\mathbf{x}_\mu)$. Furthermore, the limit $\bar{\mathbf{x}}$ of any convergent subsequence of $\{\mathbf{x}_\mu\}$ is an optimal solution to the original problem, and $\mu p(\mathbf{x}_\mu) \to 0$ as $\mu \to \infty$.

Proof

We first show that $p(\mathbf{x}_\mu) \to 0$ as $\mu \to \infty$. Let y be any feasible point and $\varepsilon > 0$.

Let x_1 be an optimal solution to the problem minimize $\{f(x) + \mu p(x), x \in X\}$, for $\mu = 1$.
If we choose $\mu \ge \frac{1}{\varepsilon}|f(y) - f(x_1)| + 2$, then by part 2 of Lemma 1 we have $f(\mathbf{x}_\mu) \ge f(x_1)$.
We now show that $p(\mathbf{x}_\mu) \le \varepsilon$. By contradiction, suppose $p(x_\mu) > \varepsilon$. Noting part 4 of lemma 1, we get

$\inf\{f(x), x \in S\} \ge \theta(\mu) = f(x_\mu) + \mu p(x_\mu) \ge f(x_1) + \mu p(x_\mu)$
$> f(x_1) + (\frac{1}{\varepsilon}|f(y) - f(x_1)| + 2)\varepsilon$
$= f(x_1) + |f(y) - f(x_1)| + 2\varepsilon > f(y)$

it follows that $\inf\{f(x), x \in S\} > f(y)$. This is not possible in the view of feasibility of y.

Thus, $p(\mathbf{x}_\mu) \le \varepsilon$ for all $\mu \ge \frac{1}{\varepsilon}|f(y) - f(x_1)| + 2$. Rearranging the above we get $\varepsilon \ge \frac{1}{\mu}|f(y) - f(x_1))| + \frac{2\varepsilon}{\mu}$, since $\varepsilon > 0$ is arbitrary, $p(x_\mu) \to 0$ as $\mu \to \infty$.
To show $\inf\{f(x) : x \in S\} = \sup_{\mu \ge 0} \theta(\mu) = \lim_{\mu \to \infty} \theta(\mu)$.
Let $\{\mathbf{x}_{\mu_k}\}$ be any arbitrary convergent sequence of $\{x_\mu\}$, and let $\bar{\mathbf{x}}$ be its limit. Then,

$\sup_{\mu \ge 0} \theta(\mu) \ge \theta(\mu_k) = f(\mathbf{x}_{\mu_k}) + \mu_k p(\mathbf{x}_{\mu_k}) \ge f(\mathbf{x}_{\mu_k})$.

Since $\mathbf{x}_{\mu_k} \to \bar{\mathbf{x}}$ and f is continuous function with $\lim_{\mu_k \to \infty} f(\mathbf{x}_{\mu_k}) = f(\bar{\mathbf{x}})$, then the above inequality implies that

$\sup_{\mu \ge 0} \theta(\mu) \ge f(\bar{\mathbf{x}})$. (2.4)

Since $p(\mathbf{x}_\mu) \to 0$ as $\mu \to \infty$, then $p(\bar{\mathbf{x}}) = 0$, that is, $\bar{\mathbf{x}}$ is a feasible solution to the original problem (P) which follows that $\inf\{f(x) : x \in S\} = f(\bar{\mathbf{x}})$.

By part 3 of Lemma 1 $\theta(\mu)$ is a nondecreasing function of μ, then

$\sup_{\mu \ge 0} \theta(\mu) = \lim_{\mu \to \infty} \theta(\mu)$. (2.5a)

$\mathbf{x}^*$ is an optimal solution to (P) by assumption implies that

$\inf\{f(x) : x \in S\} = f(\mathbf{x}^*)$ (2.5b)

and by part 4 of the Lemma 1 above

$\sup_{\mu \ge 0} \theta(\mu) \le \inf\{f(x) : x \in S\}$. (2.5c)

Equating (2.4), (2.5a), (2.5b) and (2.5c), we get
$\inf\{f(x) : x \in S\} = \sup_{\mu \ge 0} \theta(\mu) = \lim_{\mu \to \infty} \theta(\mu)$

To show $\mu p(\mathbf{x}_\mu) \to 0$ as $\mu \to \infty$.
$\theta(\mu) = f(\mathbf{x}_\mu) + \mu p(\mathbf{x}_\mu)$
$\mu p(\mathbf{x}_\mu) = \theta(\mu) - f(\mathbf{x}_\mu)$.

Taking the limit as $\mu \to \infty$ to both sides

$\lim_{\mu\to\infty} \mu p(\mathbf{x}_\mu) = \lim_{\mu\to\infty} \theta(\mu) - \lim_{\mu\to\infty} f(\mathbf{x}_\mu)$
$= \sup_{\mu\geq 0} \theta(\mu) - f(\bar{\mathbf{x}})$
$= f(\bar{\mathbf{x}}) - f(\bar{\mathbf{x}})$
$= 0$.

So that $\mu p(\mathbf{x}_\mu) \to 0$ as $\mu \to \infty$.

Note: It is interesting to observe that this result is obtained in the absence of differentiability or Karush Kuhn-Tucker regularity assumptions.

Corollary 3

If $p(\mathbf{x}_\mu) = 0$ for some μ, then $\mathbf{x}_\mu$ is an optimal solution to the original problem (P)

Proof

If $p(\mathbf{x}_\mu) = 0$,then $\mathbf{x}_\mu$ is a feasible solution to the problem (P). Furthermore, since
$\inf\{f(x), x \in S\} \geq \theta(\mu) = f(\mathbf{x}_\mu) + \mu p(\mathbf{x}_\mu) = f(\mathbf{x}_\mu)$ it follows that
$\inf\{f(x), x \in S\} \geq f(\mathbf{x}_\mu)$

it immediately follows that $\mathbf{x}_\mu$ is an optimal solution to (P). Note the significance of the assumption that $\{\mathbf{x}_\mu\}$ is contained in a compact subset X. obviously, this assumption holds if X is compact. Without this assumption, it is possible that the optimal objective values of the primal problem and the penalty problems are not equal. This assumption is not restricted in most practical cases, since the variables usually lie between finite lower and upper bounds.

From the above theorem, it follows that the optimal solution $\mathbf{x}_\mu$ to the problem to minimize $f(x) + \mu p(x)$ subject to $x \in X$ can be made arbitrarily close to the feasible region by choosing μ large enough. Furthermore, by choosing μ large enough, $f(\mathbf{x}_\mu) + \mu p(\mathbf{x}_\mu)$ can be made arbitrarily close to the optimal objective value of the original problem. One popular scheme for solving the penalty problem is to solve a sequence of problems of the form:

Minimize $f(x) + \mu p(x)$
subject to $x \in X$,

for an increasing sequence of penalty parameters. The optimal points $\{\mathbf{x}_\mu\}$ are generally infeasible as seen in proof of the Theorem 2, as the penalty parameter μ is made large, the points generated approach an optimal solution from outside the feasible region.

Hence, as mentioned earlier, this technique is also referred to as an exterior penalty function method.

Karush Kuhn Tucker multipliers at optimality

Under certain conditions, we can use the solutions to the sequence of penalty problems to recover the KKT Lagrange multipliers associated with the constraints at optimality. Suppose $X = R^n$ for simplicity and consider the primal problem (P) and the penalty function given in (2.2). In the penalty methods we solved, for various values of μ, the unconstrained problem is

Minimize $f(x) + \mu p(x)$ (2.6)
subject to $x \in X$.

Most algorithms require that the objective function has continuous first partial derivatives. Hence we shall assume that $f, g, h \in C^1$. It is natural to require, that the penalty function $p \in C^1$. As we explained earlier, the derivative of maximum $\{0, g_i(x)\}$ is usually discontinuous at points where $g_i(x) = 0$ and thus, some restrictions must be placed on Φ in order to guarantee $p \in C^1$. We assume that the functions Φ and ψ are continuously differentiable and satisfy:

$\Phi'(y) = 0$ if $y \leq 0$ and $\Phi'(y) \geq 0$ for all y. (2.7)

In view of this assumption p is differentiable whenever f, g, h are differentiable, that is, $f, g, h \in C^1$ implies $p \in C^1$ and we can write

$\nabla p(x) = \sum_{i=1}^{m} \Phi'(g_i(x_\mu)) + \sum_{i=1}^{l} \psi'(h_i(x_\mu))$.

Assuming that the conditions of Theorem 2 hold true, since $\mathbf{x}_\mu$ solves the problem to minimize $\{f(x) + \mu p(x), x \in X\}$, the gradient of the objective function of this penalty problem must vanish at $\mathbf{x}_\mu$. This gives

$\nabla f(\mathbf{x}_\mu) + \nabla p(\mathbf{x}_\mu) = 0$ for all μ,

that is,

$\nabla f(x_\mu) + \mu \sum_{i=1}^{m} \Phi'(g_i(x_\mu)) \nabla g_i(\mathbf{x}_\mu) + \sum_{i=1}^{l} \psi'(h_i(x_\mu)) \nabla h_i(\mathbf{x}_\mu) = 0$.

Now let $\bar{\mathbf{x}}$ be an accumulation point of the generated sequence $\{\mathbf{x}_\mu\}$. Without loss of generality, assume that $\{\mathbf{x}_\mu\}$ itself converges to $\bar{\mathbf{x}}$ and so $\bar{\mathbf{x}}$ is an optimal solution

to (P).

Denoted by:

$I = \{I : g_i(\bar{\mathbf{x}}) = 0\}$ to be the set of inequality constraints that are binding at $\bar{\mathbf{x}}$ and
$N = \{I : g_i(\bar{\mathbf{x}}) < 0\}$ for all constraints not binding at $\bar{\mathbf{x}}$.

Since $g_i(\bar{x}) < 0$ for all elements of N then by Theorem 2.2, We have $g_i(x_\mu) < 0$ for sufficiently large μ which results $\Phi'(g_i(x_\mu)) = 0$ (by assumption). Hence, we can write the foregoing identity as

$$\nabla f(x_\mu) + \mu \sum_{i\in I}^{m} (u_\mu)_i \nabla g_i(x_\mu) + \sum_{i=1}^{m} (v_\mu)_i \nabla h_i(x_\mu) = 0, \quad (2.8a)$$

for all μ large enough, where u_μ and v_μ are vectors having components

$$(u_\mu)_i = \mu\Phi'(g_i(x_\mu)) \geq 0 \text{ for all } I \in I \text{ and } (v_\mu)_i = \mu\psi'(h_i(x_\mu)) \text{ for all } I = 1, \ldots, l. \quad (2.8b)$$

Let us now assume that $\bar{x}$ is a regular solution such that $\nabla g_i(x_\mu)$ and $\nabla h_i(x_\mu)$ are linearly independent then, we know that there exist unique scalars $\bar{u}_i \geq 0$, $I \in I$ and $\bar{v}_i$, $I = l, \ldots, l$ such that

$$\nabla f(x_\mu) + \sum_{i\in I}^{m} \bar{u}_i \nabla g_i(x_\mu) + \sum_{i=1}^{m} \bar{v}_i \nabla h_i(x_\mu) = 0.$$

Since g, h, Φ, ψ are all continuously differentiable and since $\{x_\mu\} \to \bar{x}$, which is a regular point, we must then have in (2.8) that

$(u_\mu)_i \to \bar{u}_i$, for all $I \in I$ and $(v_\mu)_i \to \bar{v}_i$, for all $I = 1, \ldots, l$.

For sufficiently large values of μ, the multipliers given in (2.8) can be used to estimate KKT Lagrange multipliers at optimality and so we can interpret u_μ and v_μ as a sort of vector of Karush-Kuhn-Tucker multipliers. The result stated in next lemma insures that $u_\mu \to \bar{u}$ and $v_\mu \to \bar{v}$.

Lemma 4

Suppose Φ(y) and ψ(y) are continuously differentiable and satisfy (2.7), and that f, g, h are differentiable. Let (u_μ, v_μ) be defined by (2.8). Then, if $x_\mu \to \bar{x}$, and $\bar{x}$ satisfies the linear independence condition for gradient vectors of active constraints ($\bar{x}$ is a regular solution), then $(u_\mu, v_\mu) \to (\bar{u}, \bar{v})$, where $(\bar{u}, \bar{v})$ are vectors of KKT multipliers for the optimal solution $\bar{x}$ of (P).

Proof: From the Penalty Convergence Theorem, $\bar{x}$ is an optimal solution of (P).

Let

$I = \{I \mid g_i(\bar{x}) = 0\}$ and

$N = \{I \mid g_i(\bar{x}) < 0\}$.

For $I \in N$, $g_i(x_\mu) < 0$ for all μ sufficiently large, so $(u_\mu)_i = 0$ for all μ sufficiently large, whereby $\bar{u}_i = 0$ for $I \in N$.

From (2.8b) and the definition of a penalty function, it follows that $(u_\mu)_i \geq 0$ for $I \in I$, for all μ sufficiently large.

Suppose $(u_\mu, v_\mu) \to (\bar{u}, \bar{v})$ as $\mu \to \infty$. Then $\bar{u}_i = 0$ for $I \in N$. From the continuity of all functions involved,

$\nabla f(x_\mu) + \mu \sum_{i\in I}^{m} (u_\mu)_i \nabla g_i(x_\mu) + \sum_{i=1}^{l} (v_\mu)_i \nabla h_i(x_\mu) = 0$, implies $\nabla f(\bar{x}) + \sum_{i\in I}^{m} \bar{u}_i \nabla g_i(\bar{x}) + \sum_{i=1}^{m} \bar{v}_i \nabla h_i(\bar{x}) = 0$.

From the above remarks, we also have $\bar{u} \geq 0$ and $\bar{u}_i = 0$ for all $I \in N$. Thus $(\bar{u}, \bar{v})$ are vectors of Karush-Kuhn-Tucker multipliers. It therefore remains to show $(u_\mu, v_\mu) \to (\bar{u}, \bar{v})$ as $\mu \to \infty$ for some unique $(\bar{u}, \bar{v})$.

Suppose $\{(u_\mu, v_\mu)\}_{\mu=0}^{\infty}$ has no accumulation point, then $||(u_\mu, v_\mu)|| \to \infty$. But then define $(W_\mu, \lambda_\mu) = \left(\frac{(u_\mu, v_\mu)}{||(u_\mu, v_\mu)||}\right)$, and then $||(W_\mu, \lambda_\mu)|| = 1$ for all μ, and so the sequence $\{(u_\mu, v_\mu)\}_{\mu=0}^{\infty}$ has some accumulation point $(\bar{w}, \bar{\lambda})$ point. For all $I \in N$, $(w_\mu)_i = 0$ for all μ large, where by $\bar{w}_i = 0$ for all $I \in N$, and

$$\sum_{i\in I}^{m} (w_\mu)_i \nabla g_i(x_\mu) + \sum_{i=1}^{l} (\lambda_\mu)_i \nabla h_i(x_\mu) = \sum_{i=1}^{m} (w_\mu)_i \nabla g_i(x_\mu) + \sum_{i=1}^{l} (\lambda_\mu)_i \nabla h_i(x_\mu)$$
$$= \sum_{i=1}^{m} \frac{(u_\mu)_i}{||(u_\mu, v_\mu)||} \nabla g_i(x_\mu) + \sum_{i=1}^{l} \frac{(v_\mu)_i}{||(u_\mu, v_\mu)||} \nabla h_i(x_\mu)$$
$$= -\frac{\nabla f(x_\mu)}{||(u_\mu, v_\mu)||}$$

for μ large. As $\mu \to \infty$, we have $x_\mu \to \bar{x}$, $(W_\mu, \lambda_\mu) \to (\bar{w}, \bar{\lambda})$, and $||(u_\mu, v_\mu)|| \to \infty$ by assumption, and so the above equation becomes;

$\sum_{i\in I}^{m} (w_\mu)_i \nabla g_i(x_\mu) + \sum_{i=1}^{l} (\lambda_\mu)_i \nabla h_i(x_\mu) = 0$, and $||(W_\mu, \lambda_\mu)|| = 1$, which violates the linear independence condition. Therefore $\{(u_\mu, v_\mu)\}$ is bounded sequence, and so has at least one accumulation point.

Now suppose that $\{(u_\mu, v_\mu)\}$ has two accumulation points, $(\bar{u}, \bar{v})$ and $(\hat{u}, \hat{v})$. Note $\bar{u}_i = 0$ and $\hat{u}_i = 0$ for $I \in N$, and so

$$\sum_{i\in I} \bar{u}_i \nabla g_i(\bar{x}) + \sum_{i=1}^{l} \bar{v}_i \nabla h(\bar{x}) = -\nabla f(\bar{x}) = \sum_{i\in I} \hat{u}_i \nabla g_i(\bar{x}) + \sum_{i=1}^{l} \hat{v}_i \nabla h(\bar{x}),$$

so that

$$\sum_{i\in I} (\bar{u}_i - \hat{u}_i) \nabla g_i(\bar{x}) + \sum_{i=1}^{l} (\bar{v}_i - \hat{v}_i) \nabla h(\bar{x}) = 0.$$

But by the linear independence condition, $\bar{u}_i - \hat{u}_i = 0$ for all $I \in I$, and $\bar{v}_i - \hat{v}_i$. This implies that $(\bar{u}, \bar{v}) = (\hat{u}, \hat{v})$.

Remark: The quadratic penalty function satisfies the condition (2.7), but the linear penalty function does not satisfy.

As a final observation we note that in general if $x_\mu \to \bar{x}$, then since $(u_\mu)_i = \mu\Phi'(g_i(x_\mu)) \to \bar{u}_i$ and $(v_\mu)_i = \mu\psi'(h_i(x_\mu)) \to \bar{v}_i$, the sequence x_μ approaches $\bar{x}$ from outside the constraint region. Indeed as $x_\mu \to \bar{x}$ all

constraints that are active at $\bar{\mathbf{x}}$ and have positive Lagrange multipliers will be violated at x_μ because the corresponding $\Phi'(g_i(\mathbf{x}_\mu))$ are positive. Thus, if we assume that the active constraints are non degenerate (all Lagrange multipliers are strictly positive), every active constraint will be approached from outside of the feasible region.

Consider the special case if p is the quadratic penalty function given by:

$p(x) = \sum_{i=1}^{m}(\text{maximum}\{0, g_i(\mathbf{x})\})^2 + \sum_{i=1}^{m}(h_i(\mathbf{x}))^2$, then $p(y) = \sum_{i=1}^{m}(\text{maximum}\{0, y_i\})^2 + \sum_{i=1}^{m} y_i^2$, $\Phi'(y) = 2\text{maximum}\{0, y\}$ and $\psi'(y) = 2y$. Hence, from (2.8), we obtain

$(\mathbf{u}_\mu)_i = 2\mu(\text{maximum}\{0, g_i(\mathbf{x}_\mu)\})$, for all $I \in I$, and
$(\mathbf{v}_\mu)_i = 2\mu h_i(x_\mu)$ for all $I = 1, \ldots, l$. (2.9)

In particular, observe that if $\bar{u}_i > 0$ for some $I \in I$, then $(\mathbf{u}_\mu)_i > 0$ for μ large enough and then $g_i(x_\mu) > 0$ and by our assumption $\Phi'(g_i(\mathbf{x}_\mu)) > 0$ for $g_i(x_\mu) > 0$. This means that $g_i(x) \le 0$ is violated all along the trajectory leading to $\bar{\mathbf{x}}$, and in the limit $g_i(\bar{\mathbf{x}}) = 0$. Hence, if $\bar{u}_i > 0$ for some $I \in I$, $\bar{v}_i \neq 0$ for all I, then all the constraints binding at $\bar{\mathbf{x}}$ are violated along the trajectory $\{\mathbf{x}_\mu\}$ leading to $\bar{\mathbf{x}}$.

Example 3

Consider the following optimization problem:

Minimize $x_1^2 + x_2^2$
subject to $x_1 + x_2 = 1$

and the corresponding penalty problem:

Minimize $x_1^2 + x_2^2 + \mu(x_1 + x_2 - 1)^2$
subject to $(x_1, x_2) \in R^2$,

where μ is a large number.

$\mathbf{X}_\mu = [\frac{\mu}{2\mu+1}, \frac{\mu}{2\mu+1}]$ is the solution for the penalty problem.
And
$h(\mathbf{x}_\mu) = \frac{\mu}{2\mu+1} + \frac{\mu}{2\mu+1} - 1 = \frac{-\mu}{2\mu+1}$; and so,

$(\mathbf{v}_\mu)_i = (v_\mu) = 2\mu h(\mathbf{x}_\mu) = 2\mu(\frac{-1}{2\mu+1})$

Implies

$v_\mu = \frac{-2\mu}{2\mu+1}$ from (2.8)

As $\mu \to \infty$, $v_\mu \to -1$, which is the optimal value of the Lagrange multiplier for this example.

Example 4

Minimize x
subject to $-x + 2 \le 0$.

The corresponding penalty problem is:

Minimize $x + \mu(-x + 2)^2$
subject to $x \in R$,
$\nabla_x \theta(\mathbf{x}_\mu) = 1 + 2\mu[\text{maximum}\{0, -\mathbf{x}_\mu + 2\}](-1) = 0$, for $x < 2$
which implies that $1 + 2\mu\mathbf{x}_\mu - 4\mu = 0$.
Therefore, $x_\mu = 2 - \frac{1}{2\mu}$, and
$(u_\mu)_i = 2\mu(g_i(x_\mu))$, for $I \in I$
$= 2\mu(-(\frac{4\mu+1}{2\mu}) + 2)$
$= 1$
It follows that $U_\mu = 1$.

Note that, as $\mu \to \infty$, $u_\mu = 1$, the optimal value of the Lagrange multiplier for the primal problem.

Example 5

Minimize $x_1^2 + 2x_2^2$
subject to $-x_1 - x_2 + 1 \le 0$, $x \in R^2$.

For this problem the Lagrangian is given by
$L(x, u) = x_1^2 + 2x_2^2 + u(-x_1 - x_2 + 1)$. The KKT conditions yield:

$\frac{\partial L(x,u)}{\partial x_1} = 2x_1 - u = 0$ and $\frac{\partial L(x,u)}{\partial x_2} = 4x_2 - u = 0$; $u(-x_1 - x_2 + 1) = 0$.

Solving these results in $x_1^* = 2/3$; $x_2^* = 2/3$; $\bar{u} = 4/3$; ($u = 0$ yields an infeasible solution).
To consider this example using penalty method, define the penalty function

$$p(x) = \begin{cases} (-x_1 - x_2 + 1)^2, \text{if } g_i(\mathbf{x}) > 0 \\ 0, \text{if } g_i(\mathbf{x}) \le 0. \end{cases}$$

The unconstrained problem is then,

minimize $x_1^2 + 2x_2^2 + \mu p(\mathbf{x})$.
If $p(x) = 0$, then the optimal solution is $x^* = (0, 0)$ which is infeasible.

Therefore, $p(x) = (-x_1 - x_2 + 1)^2$, $\theta(\mathbf{x}, \mu) = x_1^2 + 2x_2^2 + \mu(-x_1 - x_2 + 1)^2$, and the necessary conditions for the optimal solution yield the following:

$\frac{\partial \theta(x,\mu)}{\partial x_1} = 2x_1 + 2\mu((-x_1 - x_2 + 1)(-1)$, and
$\frac{\partial \theta(x,\mu)}{\partial x_2} = 2x_1 + 2\mu(-x_1 - x_2 + 1)(-1) = 0$.

Thus, $x_{\mu_1} = \frac{2\mu}{(1+3\mu)}$ and $x_{\mu_2} = \frac{\mu}{(2+3\mu)}$ for any fixed μ.
When μ → ∞, this converges to the optimum solution of $x^* = (\frac{2}{3}, \frac{1}{3})$.
Now suppose we use (2.8) to define

$(u_\mu)_i = 2\mu(\text{maximum } \{0, g_i(x_\mu)\})$, then

$u_\mu = 2\mu(1 - \{2\mu/(2+3\mu)\} - \{\mu/(2+3\mu)\})$
$= 2\mu(1 - \{3\mu/(2+3\mu)\})$
$= (4\mu/(2+3\mu))$.

Then it is readily seen that $\lim_{\mu\to\infty}(4\mu/(2+3\mu)) = 4/3 = \bar{u}$ (the optimal Lagrangian multiplier for this example). Therefore the above Lemma 4 is true under some regularity conditions.

III-Conditioning of the Hessian matrix

Since the penalty function method must, for various (large) values of μ, solve the unconstrained problem:

Minimize f(**x**) + μp(**x**)
subject to x ∈ **X**,

It is important, in order to evaluate the difficulty of such a problem, to determine the eigenvalue structure of the Hessian of this modified objective function. The motivation for this is that the eigenvalue structure of the Hessian of the objective function determines the natural rates of convergence for algorithms designed for unconstrained optimization problems. We show here that the structure of the eigenvalue of the corresponding unconstrained problem becomes increasingly unfavorable as μ increases. Although, one usually insists for computational as well as theoretical purposes that the function p ∈ c^1, one usually does not insist that p ∈ c^2. In particular, the most popular penalty function p(x) = (maximum {0, y})^2, has discontinuity in its second derivative at any point where the component of g is zero, that is, $\Phi'(y) = 2(\text{maximum}\{0, y\})$, but $\Phi''(y)$ would have been undefined at y = 0 (as shown below). Hence, the Hessian of the unconstrained problem would be undefined at points having binding inequality constraints. At first this might appear to be a serious drawback, since it means the Hessian is discontinuous at the boundary of the constraint region-right where, in general, the solution is expected to lie.

However, as pointed out above, the penalty method generates points that approach a boundary solution from the outside the constraint region. Thus, except for some possible chance occurrences, the sequence will, as $\mathbf{x}_\mu \to \bar{\mathbf{x}}$, be at points where the Hessian is well-defined. Furthermore, in iteratively solving the above unconstrained problem with a fixed μ, a sequence will be generated that converges to $\mathbf{x}_\mu$ which is (for most value of μ) a point where the Hessian is well-defined, and the standard type of analysis will be applicable to the tail of such a sequence (Luenberger, 1974).

Consider the constrained optimization problem:

Minimize {f(x), x ∈ S}

whose feasible region we denote by S: = {x ∈ X | g(x) ≤ 0, h(x)} and the corresponding unconstrained problem:

$\theta(\mathbf{x}, \mu) = f(x) + \mu p(x)$, $p(x) = \sum_{i=1}^{m} \Phi(g_i(\mathbf{x})) + \sum_{i=1}^{l} \psi(h_i(\mathbf{x}))$,

where f, g, h, Φ, ψ are assumed to be twice continuously differentiable at $\mathbf{x}_\mu$. Then denoting by ∇ and ∇^2 the gradient and the Hessian operators for the functions Q, f, g, h, respectively, and denoting the first and second derivatives of Φ and ψ as Φ′, ψ′ and Φ″,ψ″ (all with respect to x) we have,

$\nabla_x \theta(\mathbf{x}_\mu, \mu) = \nabla f(\mathbf{x}_\mu) + \mu \sum_{i=1}^{m} \Phi'(g_i(x_\mu)) \nabla g_i(\mathbf{x}_\mu) + \mu \sum_{i=1}^{l} \psi'(h_i(x_\mu)) \nabla h_i(\mathbf{x}_\mu)$
And

$Q(x, \mu) = \nabla_x^2 \theta(x_\mu) = [\nabla^2 f(x_\mu) + \mu \sum_{i=1}^{m} \Phi'(g_i(x_\mu)) \nabla^2 g_i(x_\mu) + \sum_{i=1}^{l} \psi'(h_i(x_\mu)) \nabla^2 h_i(x_\mu)] + \mu \sum_{i=1}^{m} \Phi''(g_i(x_\mu)) \nabla g_i(x_\mu) \nabla(g_i(x_\mu))^t + \mu \sum_{i=1}^{l} \psi''(h_i(x_\mu)) \nabla h_i(x_\mu) \nabla h_i(x_\mu)^t$. (2.10)

To estimate the convergence rate of algorithms designed to solve the modified objective function let us examine the eigenvalue structure of (2.10) as μ → ∞, and under the conditions of Theorem 2.2, as x ≡ $\mathbf{x}_\mu \to \bar{\mathbf{x}}$, an optimum solution to the given problem. Assuming that $\{\mathbf{x}_\mu\} \to \bar{\mathbf{x}}$ and $\bar{\mathbf{x}}$ is a regular solution, we have from (2.8) that,

$\mu\Phi'(g_i(\mathbf{x}_\mu)) \to \bar{u}_i \geq 0$ for i ∈ I and $\mu\psi'(h_i(\mathbf{x}_\mu)) \to \bar{v}_i$, i = 1, . . . , l,

where the optimal Lagrange multipliers associated with the i^{th} constraint. Hence, the term in [.] approaches the Hessian of the Lagrangian function of the original problem as $\mathbf{x}_\mu \to \bar{\mathbf{x}}$, which is

$\nabla^2 L(\bar{\mathbf{x}}) = \nabla^2 f(\bar{\mathbf{x}}) + \sum_{i=1}^{m} \bar{u}_i \nabla^2 h_i(\bar{\mathbf{x}}) + \sum_{i=1}^{m} \bar{v}_i \nabla^2 h_i(\bar{\mathbf{x}})$,

and has a limit that is independent of μ. The other term in (2.10), however, is strongly tied in with μ, and is potentially explosive.

For example, if $\Phi(y) = (\text{maximum}\{0, y\})^2$ and ψ(y) = y^2, as the popular quadratic penalty functions for the inequality and equality constraints and considering a primal problem with equality or inequality constraints separately we have two matrices.

$$\Phi''(g_i(\mathbf{x}_\mu)) = 2\begin{bmatrix} e_1 & 0 & . & . & . & 0 \\ 0 & e_2 & . & . & . & 0 \\ . & . & . & . & . & . \\ . & . & . & . & . & . \\ . & . & . & . & . & . \\ 0 & 0 & . & . & . & e_m \end{bmatrix}$$

where

$$e_i = \begin{cases} 1 \text{ if } g_i(\mathbf{x}_\mu) > 0 \\ 0 \text{ if } g_i(\mathbf{x}_\mu) < 0 \\ \text{undefined if } g_i(\mathbf{x}_\mu) = 0. \end{cases}$$

Thus,

$$\mu\sum_{i=1}^{m} \Phi''(g_i(x_\mu))\, \nabla g_i(\mathbf{x}_\mu) \nabla g_i(\mathbf{x}_\mu)^t = 2\mu\sum_{i=1}^{m} \nabla g_i(\mathbf{x}_\mu) \nabla g_i(\mathbf{x}_\mu)^t,$$

which is 2μ times a matrix that approaches $\sum_{i=1}^{m} \nabla g_i(\bar{\mathbf{x}})\, \nabla g_i(\bar{\mathbf{x}})^t$. This matrix has rank equal to the rank of the active constraints at $\bar{\mathbf{x}}$ (Luenberger, 1974).

Assuming that there are r_1 active constraints at the solution $\bar{\mathbf{x}}$, then for well behaved Φ the matrix Q(x, μ) with only inequality constraints has r_1 eigenvalues that tend to ∞ as μ → ∞, but the n – r_1 eigenvalues, though varying with μ, tend to finite limits. These limits turnout to be the eigenvalues of $L(\bar{\mathbf{x}})$ restricted to the tangent subspace M of the active constraints. The other matrix $2\mu\sum_{i=1}^{l} \nabla h_i(\mathbf{x}_\mu) \nabla h_i(\mathbf{x}_\mu)^t$ with l equality constraints has rank l. As μ → ∞, $\mathbf{x}_\mu \rightarrow \bar{\mathbf{x}}$, the matrix Q(x, μ) with only equality constraints has l eigenvalues that approach infinity while the n - l eigenvalues approach some finite limits. Consequently, we can expect a severely ill-conditioned Hessian matrix for large values of μ.

Considering equation (2.10) with both equality and inequality constraints we have as $\mathbf{x}_\mu \rightarrow \bar{\mathbf{x}}$, is a local solution to the constrained minimization problem (P) and that it satisfies

$\mathbf{h}(\bar{\mathbf{x}}) = 0$ and $\mathbf{g}_A(\bar{\mathbf{x}}) = 0$ and $\mathbf{g}_I(\bar{\mathbf{x}}) < 0$,

where g_A and g_I, is the induced partitioning of g into r_1 active and r_2 inactive constraints, respectively. Assuming that the l gradients of h and the r_1 gradients of g_A evaluated at $\bar{\mathbf{x}}$ together are linearly independent, then $\bar{\mathbf{x}}$ is said to be regular. It follows from this expression that, for large μ and for $\mathbf{x}_\mu$ close to the solution of (P) the matrix Q has l + r_1, eigenvalues of the order of μ. Consequently, we can expect a severely ill-conditioned Hessian matrix for large values of μ. Since the rate of convergence of the method of steepest descent applied to a functional is determined by the ratio of the smallest to the largest eigenvalues of the Hessian of that functional, it follows in particular that the steepest descent method applied to θ converges slowly for large μ.

In examining the structure of Q is therefore; first, as μ is increased, the solution of the penalty problem approaches the solution of the original problem, and, hence, the neighborhood in which attention is focused for convergence analysis is close to the true solution. This means that the structure of the Lagrangian in the neighborhood of interest is close to that of Lagrangian at the true solution. Secondly, we conclude that, for large μ, the matrix Q is positive definite. For any μ, Q must be at least positive semi definite at the solution to the penalty problem: it is indicated that a stronger condition holds for large μ.

Example 6

Consider the auxiliary function, $\theta(x, \mu) = x_1^2 + 2x_2^2 + \mu(-x_1^2 - x_2^2 + 1)^2$, of example 2.5. The Hessian is:

$$H = \begin{bmatrix} 2 + 2\mu & 2\mu \\ 2 & 4 + 2\mu \end{bmatrix}.$$

Suppose we want to find its eigenvalues by solving det |H – λI| = 0,

$|H - \lambda I| = (2 + 2\mu - \lambda)(4 + 2\mu - \lambda) - 4\mu^2$

$= \lambda^2 - (6 + 4\mu)\lambda + 8 + 12\mu.$

This quadratic equation yields

$\lambda = (3 + 2\mu) \pm \sqrt{4\mu^2 + 1}$,

$\lambda_1 = (3 + 2\mu) - \sqrt{4\mu^2 + 1}$ and $\lambda_2 = (3 + 2\mu) + \sqrt{4\mu^2 + 1}$.

Note that $\lambda_2 \rightarrow \infty$ as μ → ∞, while λ_1 is finite; and, hence, the condition number of H approaches ∞ as μ → ∞. Taking the ratio of the largest and the smallest eigenvalue yields

It should be clear that as μ → ∞, the limit of the preceding ratio also goes to ∞. This indicates that as the iterations proceed and we start to increase the value of μ, the Hessian of the unconstrained function that we are minimizing becomes increasingly ill-conditioned. This is a common situation and is especially problematic if we are using a method for the unconstrained optimization that requires the use of the Hessian.

Unconstrained minimization techniques and penalty function methods

In this we mainly concentrate on the problems of efficiently solving the unconstrained problems with a penalty method. The main difficulty as explained above is the extremely unfavorable eigenvalue structure. Certainly straight forward application of the method of steepest descent is out of the question.

Newton's method and penalty function methods

One method for avoiding slow convergence for the problems is to apply Newton's method (or one of its variations), since the order two convergence of Newton's method is unaffected by the poor eigenvalue structure.

In applying the method, however, special care must be devoted to the manner by which the Hessian is inverted, since it is ill-conditioned. Nevertheless, if second order information is easily available, Newton's method offers an extremely attractive and effective method for solving modest size penalty and barrier optimization problems.

When such information is not readily available, or if data handling and storage requirements of Newton's method are excessive, attention naturally focuses on zero order or first order methods.

Conjugate gradients and penalty function methods

According to Luenberger (1984) the partial conjugate gradient method for solving unconstrained problems is ideally suited to penalty and barrier problems having only a few active constraints. If there are l active constraints, then taking cycles of 1+1 conjugate gradient steps will yield a rate of convergence that is independent of μ. For example, consider the problem having only equality constraints:

Minimize f(x) (P)
subject to h(x) = 0,

where $x \in R^n$, $h(x) \in R^l$, $l < n$. Applying the standard quadratic penalty method, we solve instead the unconstrained problem:

minimize $f(x) + \mu|h(\mathbf{x})|^2$,

for large μ. The objective function of this problem has a Hessian matrix that has l eigenvalues that are of order μ in magnitude, while the remaining n – l eigenvalues are close to the eigenvalues of the matrix L_M, corresponding to the primal problem (P). Thus, letting $x_{\mu+1}$ be determined from x_μ by making l + 1 steps of a (nonquadratic) conjugate gradient method, and assuming $x_\mu \rightarrow \bar{\mathbf{x}}$, a solution to $\theta(\mu)$, the sequence $\{f(x_\mu)\}$ converges linearly to $f(\bar{\mathbf{x}})$ with a convergence ratio equal to approximately

$$\left(\frac{\beta-\alpha}{\beta+\alpha}\right)^2$$

where α and β are, respectively, the smallest and largest eigenvalues of $L_M(\bar{\mathbf{x}})$. This is an extremely effective technique when l is relatively small. The method can be used for problems having inequality constraints as well but it is advisable to change the cycle length, depending on the number of constraints active at the end of the previous cycle.

Here we will use Powell's method which is the zero order method. Powell's method is an extension of the basic pattern search methods. It is the most widely used direct search method and can be proved to be a method of conjugate directions. This is as effective as the first order methods like the gradient method for solving unconstrained optimization problems. The reason why we use it here is:

a. First, it is assumed that the objective and constraint functions be continuous and smooth (continuously differentiable). Experience has shown this to be a more theoretical than practical requirement and this restriction is routinely violated in engineering design and in some facility location problems. Therefore it is better to develop a general code that solves both differentiable and non-differentiable problems.
b. The input of the derivative if it exists is tiresome for problems with large number of variables. In spite of its advantages, Newton's method for example is not generally used in practice due to the following features of the method:

i. It requires the storing of the n×n Hessian matrix of the objective function,
ii. It becomes very difficult and sometimes, impossible to compute the elements of the Hessian matrix of the objective function,
iii. It requires the inversion of the Hessian matrix of the objective function at each step,
iv. It requires the evaluation of the product of inverse of the Hessian matrix of the objective function and the negative of the gradient of the objective function at each step.

Because of the above reasons I do not prefer first and second order methods and I did not give more emphasis on these methods and their algorithms.

Finally, we should not use second-order gradient methods (e.g., pure Newton's method) with the quadratic loss penalty function for inequality constraints, since the Hessian is discontinuous (Belegundu and, Chandrupatla, 1999). To see this clearly, consider:

Minimize f(x) = 100/x
subject to g = x -5 ≤ 0,

with f(x) being a monotonically decreasing function of x. At the optimum $\bar{x} = 5$, the gradient of p(x) is 2μmax(0, x – 5). Regardless of whether we approach $\bar{x}$ from the left or right, the value of $\frac{dp}{dx}$ at $\bar{x}$ is zero. So, p(x) is first-order differentiable. However, $\frac{\partial p^2}{\partial x^2} = 0$ when approaching $\bar{x}$ from the left while $\frac{\partial p^2}{\partial x^2} = 2\mu$ when approaching from the right. Thus, the penalty function is not second-order differentiable at the optimum.

Powell's method and penalty function methods

Powell's method is a zero-order method, requiring the evaluation of f(x) only. If the problem involves n design variables, the basic algorithm is (Kiusalaas, 2005):

Choose a point x_0 in the design space.

Choose the starting vectors v_i , i = 1, 2, . . . , n(the usual choice is $v_i = e_i$, where e_i is the unit vector in the x_i-coordinate direction).
Cycle
do with i = 1, 2, . . . , n
Minimize f(x) along the line through x_{i-1} in the direction of v_i. Let the minimum point be x_i.

end
do $v_{n+1} \leftarrow x_n - x_0$ (this vector is conjugate to v_{n+1} produced in the previous loop).
Minimize f(x) along the line through x_0 in the direction of v_{n+1}. Let the minimum point be x_{n+1}.
if $|x_{n+1} - x_0| < \varepsilon$ exit loop
do with i = 1, 2, . . . , n
$v_i \leftarrow v_{i+1}$ (v_1 is discarded, the other vectors are reused) end do end cycle.

Powell (1997) demonstrated that the vectors $\mathbf{v}_{n+1}$ produced in successive cycles are mutually conjugate, so that the minimum point of a quadratic surface is reached in precisely n cycles. In practice, the merit function is seldom quadratic, but as long as any function can be approximated locally by quadratic function, Powell's method will work. Of course, it usually takes more than n cycles to arrive at the minimum of a non quadratic function. Note that it takes n line minimizations to construct each conjugate direction.

Powell's method does have a major flaw that has to be remedied; if f(x) is not a quadratic, the algorithm tends to produce search directions that gradually become linearly dependent, thereby ruining the progress towards the minimum. The source of the problem is the automatic discarding of v_1 at the end of each cycle. It has been suggested that it is better to throw out the direction that resulted in the largest decrease of f(x), a policy that we adopt. It seems counter intuitive to discard the best direction, but it is likely to be close to the direction added in the next cycle, thereby contributing to linear dependence. As a result of the change, the search directions cease to be mutually conjugate, so that a quadratic form is not minimized in n cycles any more. This is not a significant loss since in practice f(x) is seldom a quadratic anyway. Powell suggested a few other refinements to speed up convergence. Since they complicate the bookkeeping considerably, we did not implement them.

General description of the penalty function method algorithm

The detail of this and a MATLAB computer program for implementing the penalty method using Powell's method of unconstrained minimization is given in the appendix.

Algorithm 1 (Algorithm for the penalty function method)

To solve the sequence of unconstrained problems with monotonically increasing values of μ_k, let $\{\mu_k\}$, k = 1, . . . be a sequence tending to infinity such that $\mu_k \geq 0$ and $\mu_{k+1} > \mu_k$. Now for each k we solve the problem

Minimize $\{\theta(x, \mu_k), x \in X\}$. (2.11)

To obtain x_k, the optimum it is assumed that problem (2.11) has a solution for all positive values of μ_k. A simple implementation known as the sequential unconstrained minimization technique (SUMT) is given below.

Step 0: (Initialization) Select a growth parameter $\beta > 1$ and a stopping parameter $\varepsilon > 0$ and an initial value of the penalty parameter μ_0. Choose a starting point $\mathbf{x}_0$ that violates at least one constraint and formulate the augmented objective function $\theta(x, \mu_k)$. Let k = 1.

Step 1: Iterative - Starting from x_{k-1}, use an unconstrained search technique to find the point that minimizes $\theta(x, \mu_{k-1})$ and call it x_k .

Step 2: Stopping Rule - If the distance between x_{k-1} and x_k is smaller than ε, that is, $\| x_{k-1} - x_k \| < \varepsilon$ or the difference between two successive objective function values is smaller than ε, that is, $|f(x_{k-1}) - f(x_k)| < \varepsilon$, stop with x_k an estimate of the optimal solution otherwise, put $\mu_k = \beta\mu_{k-1}$, and formulate the new $\theta(x, \mu_k)$ and put k = k+1 and return to the iterative step.

Considerations for implementation of the penalty function method

Starting point x_1

First in the solution step is to select a starting point. A good rule of thumb is to start at an infeasible point. By design then, we will see that every trial point, except the last one, will be infeasible (exterior to the feasible region). A reasonable place to start is at the unconstrained minimum. Always we should ensure that the penalty does not dominate the objective function during initial iterations of penalty function method.

Selecting the initial penalty parameter (μ_0)

The initial penalty parameter μ_0 should be fixed so that the magnitude of the penalty term is not much smaller than the magnitude of objective function. If an imbalance exists, the influence of the objective function could direct the algorithm to head towards an unbounded minimum even in the presence of unsatisfied constraints. Because the exterior penalty method approach seems to work so well, it is natural to conjecture that all we have to do is set μ to a very large number and then optimize the resulting augmented objective function $\theta(\mathbf{x}, \mu_k)$ to obtain the

solution to the original problem. Unfortunately, this conjecture is not correct. First, “large” depends on the particular model. It is almost always impossible to tell how large μ must be to provide a solution to the problem without creating numerical difficulties in the computations. Second, in a very real sense, the problem is dynamically changing with the relative position of the current value of **x** and the subset of the constraints that are violated. The third reason why the conjecture is not correct is associated with the fact that large values of μ create enormously steep valleys at the constraint boundaries. Steep valleys will often present formidable if not insurmountable convergence difficulties for all preferred search methods unless the algorithm starts at a point extremely close to the minimum being sought. Fortunately, there is a direct and sound strategy that will overcome each of the difficulties mentioned above. All that needs to be done is to start with a relatively small value of μ. The most frequently used initial penalty parameters in the literature are 0.01, 0.1, 2, 5, and 10. This will assure that no steep valleys are present in the initial optimization of $\theta(\mathbf{x}, \mu_k)$. Subsequently, we will solve a sequence of unconstrained problems with monotonically increasing values of μ chosen so that the solution to each new problem is “close” to the previous one. This will preclude any major difficulties in finding the minimum of $\theta(x, \mu_k)$ from one iteration to the next.

Subsequent values of the penalty parameter

Once the initial value of the μ_k is chosen, the subsequent values of μ_k have to be chosen such that $\mu_{k+1} > \mu_k$.

For convenience, the value of μ_k is chosen according to the relation:

$\mu_{k+1} = \beta\mu_k$.
where $\beta > 1$. The value of β can be taken as in most literatures 2, 5, 10,100 etc.

Various approaches to selecting the penalty parameter sequence exist in the literature. The simplest is to keep it constant during all iterations and we consider here the penalty parameter as same for all constraints.

Normalization of the constraints

An optimization may also become ill-conditioned when the constraints have widely different magnitudes and thus badly affect the convergence rate during the minimization of θ-function. Much of the success of SUMT depends on the approach used to solve the intermediate problems, which in turn depends on their complexity. One thing that should be done prior to attempting to solve a nonlinear programming using a penalty function method is, to scale the constraints so that the penalty generated by each is about the same magnitude. This scaling operation is intended to ensure that no subset of the constraints has an undue influence on the search process. If some constraints are dominant, the algorithm will steer towards a solution that satisfies those constraints at the expense of searching for the minimum. In either case, convergence may be exceedingly slow. Discussion on how to normalize constraints is given on barrier function methods.

Test problems (Testing practical examples)

As discussed in previous sections, a number of algorithms are available for solving constrained nonlinear programming problems. In recent years, a variety of computer programs have been developed to solve engineering optimization problems. Many of these are complex and versatile and the user needs a good understanding of the algorithms/computer programs to be able to use them effectively. Before solving a new engineering design optimization problem, we usually test the behavior and convergence of the algorithm/computer program on simple test problems. Eight test problems are given in this section. All these problems have appeared in the optimization and on facility location literature and most of them have been solved using different techniques.

Example 1

Consider the optimization problem:

Minimize $f(x) = (x_1 - 5)^2 + (x_2 - 6)^2$
subject to $x_1^2 - 4 \le 0$
$e^{-x_1} - x_2 \le 0$
$x_1 + 2x_2 - 4 \le 0$
$x_1 \ge 0$
$x_2 \ge 0$

We consider the sequence of problems:

$$\theta(\mathbf{x}, \mu) = f(x) + \mu[(\text{maximum}\{0, x_1^2 - 4\})^2 + (\text{maximum}\{0, e^{-x_1} - x_2\})^2 + (\text{maximum}\{0, -x_1\})^2] + (\text{maximum}\{0, -x_2\})^2 + (\text{maximum}\{0, x_1 + 2x_2 - 4\})^2]$$

Optimum solution point using Mathematica is x = (1.67244, 1.21942) and Optimum solution is at f^* = 34.1797
Optimum solution point using MATLAB is x = (2.000000003129, 1.000000123435) and
Optimum solution is at f^* = 34.0000050125

The graph of the feasible region and steps of a computer program (Deumlich, 1996) with the contours of the objective function are shown in Figure 2.

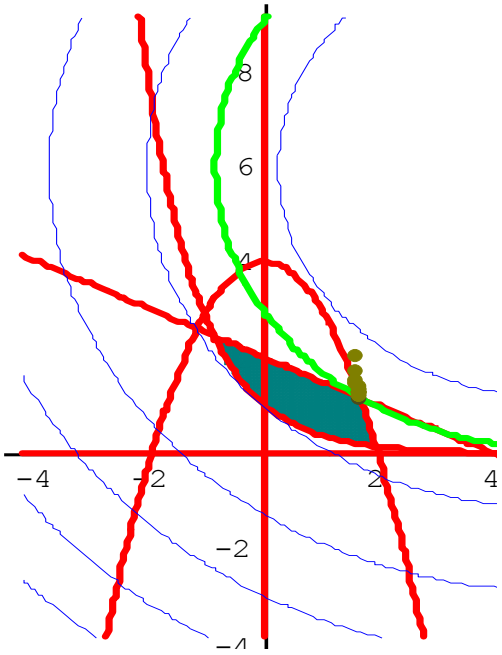

Figure 2. The sequence of unfeasible results from outside the feasible region. And the iteration step using MATLAB for penalty and the necessary data are given.

Table 1. The iteration step using MATLAB.

μ	x_{min}	f_{min}	aug_{min}
1.00	(-10.000000000000,-10.000000000000)	481.00	485615721.72568649
10.00	(2.055936255098, 1.977625505064)	24.847007912	65.8104676667
100.00	(2.003470085541, 1.120258484331)	32.791068788	38.7633368822
1000.00	(2.000316074133, 1.012311179827)	33.875143422	34.4986676000
10000.00	(2.000031285652, 1.001234049950)	33.987473310	34.0500992010
100000.00	(2.000003125478, 1.000123433923)	33.998746923	34.0050122192
1000000.00	(2.000000312552, 1.000012343760)	33.999874687	34.0005012538
10000000.00	(2.000000031250, 1.000001234352)	33.999987469	34.0000501229
100000000.00	(2.000000003129, 1.000000123435)	33.999998747	34.0000050125

And the iteration step using MATLAB for penalty method and the necessary data are given as follows:

Initial:

x_1 = [2; 5];
μ = 1; beta = 10;
tol = 1.0e-4; tol1 = 1.0e-6; h = 0.1;N = 10 (Table 1).

Example 2

Consider the optimization problem:

Minimize f(x) = $(x_1 - 3)^2 + (x_2 - 4)^2$

subject to $x_1^2 - x_2 \leq 0$

$e^{-x_1} - x_2 \leq 0$
$-x_1 + 2x_2 - 2 \leq 0$

We consider the sequence of problems:
$\theta(\mathbf{x}, \mu) = f(x) + \mu[(\text{maximum}\{0, x_1^2 - x_2\})^2 + (\text{maximum}\{0, e^{-x_1} - x_2\})^2 + (\text{maximum}\{0, -x_1 + 2x_2 - 2\})^2]$
Optimum solution point using Mathematica is x = (1.33271, 1.7112) and Optimum solution is at f^* = 8.26363
Optimum solution point using MATLAB is x = (1.280776520285, 1.640388354297) and
Optimum solution is at f^* = 8.5235020151.

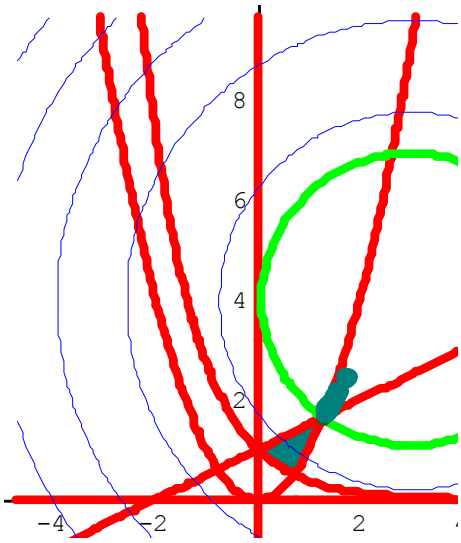

Figure 3. The sequence of unfeasible results.

Table 2. The iteration step using MATLAB.

μ	x_{min}	f_{min}	aug_{min}
1.00	(10.00000000000, 10.00000000000)	85.000000000	8249.000000000
10.00	(1.762313021963, 2.438395531128)	3.9704775728	20.8446729268
100.00	(1.376888913083, 1.774431551334)	7.5876445202	12.0187470067
1000.00	(1.291940370698, 1.655287557366)	8.4151441359	8.9534555697
10000.00	(1.281912903137, 1.641896920522)	8.5124734059	8.5675584019
100000.00	(1.280890263154, 1.640539267595)	8.5223932351	8.5279146690
1000000.00	(1.280787794313, 1.640403311676)	8.5233871397	8.5239394231
10000000.00	(1.280777545189, 1.64038971403)	8.5234865508	8.5235417778
100000000.00	(1.280776520285, 1.64038835429)	8234964917	8.5235020150

The graph of the feasible region and steps of a computer program (based on Mathematica) with the contours of the objective function are shown in Figure 3.

And the iteration step using MATLAB for penalty method and the necessary data are given as follows:

Initial:

x = [10; 10];
μ = 1; beta = 10;
tol = 1.0e-3; tol1 = 1.0e-5; h = 0.1; N = 10 (Table 2).

Example 3

Consider the optimization problem:

Minimize f(x) = $-\ln(\sqrt{x_1} + \sqrt{x_2})$
subject to $x_1 \geq 0$
$x_2 \geq 0$
$2x_1 + 3x_2 \leq 6$

Solution

The θ-function of the corresponding unconstrained problem is:

$$\theta(\mathbf{x}, \mu) = f(x) + \mu[(\text{maximum}\{0, -x_1\})^2 + (\text{maximum}\{0, -x_2\})^2 + (\text{maximum}\{0, 2x_1 + 3x_2 - 6\})^2]$$

The Exterior penalty function method, coupled with the Powell method of unconstrained minimization and golden

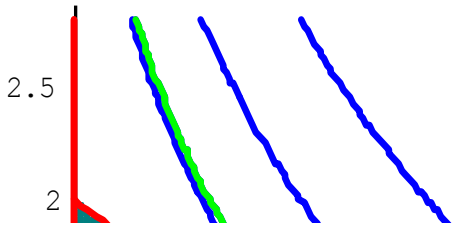

Figure 4. The sequence of unfeasible results from outside the feasible region.

Table 3. The iteration step using MATLAB.

μ	x_{min}	f_{min}	aug_{min}
1.00000	(100.0000000000, 100.0000000000)	-2.9957322736	244033.0042677264
1.50000	(1.812414390011, 0.805517491028)	-0.8081555566	-0.805586944500
2.25000	(1.812414390377, 0.805517490784)	-0.8081555566	-0.804302638400
3.37500	(1.812414395786, 0.805517487178)	-0.8081555566	-0.802376179003
5.06250	(1.803696121437, 0.801642712659)	-0.8057446020	-0.804976156000

bracket and golden search method of one-dimensional search, is used to solve this problem.
Optimum solution point using Mathematica is **x** = (1.80125, 0.800555) and
Optimum solution is at f^* = -0.804892

Optimum solution point using MATLAB is **x** = (1.803696121437, 0.801642712659) and
Optimum solution is at f^* = -0.8057446020
The graph of the feasible region and steps of a computer program (Deumlich, 1996) with the contours of the objective function are shown in Figure 4. And the iteration step using MATLAB for penalty method and the necessary data are given as follows:

Initial:

x_1 = [100; 100];
μ = 1; beta = 1.5;
tol = 1.0e-9; tol1 = 1.0e-3; h = 0.1;N = 10; Table 3

Example 4

Consider the optimization problem:

Minimize f(**x**) = $(x_1 - 2)^2$ - 5ln($(x_2 - 2)^2 + 1$)
subject to $x_1^2 + x_2 - 4 \leq 0$
$e^{-x_1} - x_2 \leq 0$.

We consider the sequence of problems:

$\theta(\mathbf{x}, \mu) = (x_1 - 2)^2 - 5\ln((x_1 - 2)^2 + 1) + \mu[(\text{maximum}\{0, x_1^2 + x_2 - 4\})^2 + (\text{maximum}\{0, e^{-x_1} - x_2\})^2]$.

We can solve this problem numerically. Since the function f is not convex we can expect local minimum points depending on the choice of the initial point.

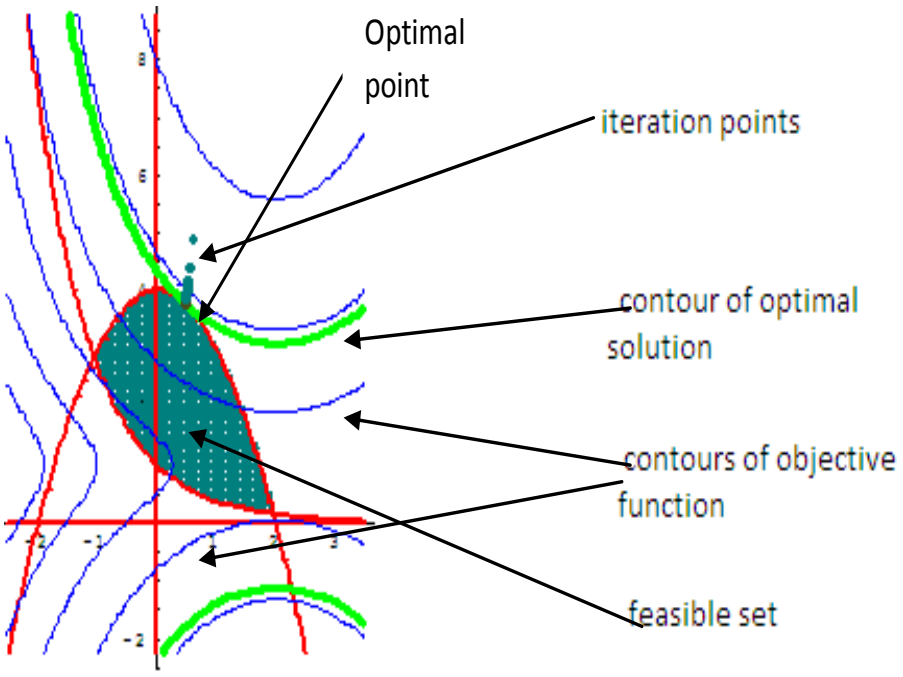

Figure 5. The sequence of unfeasible results from outside the feasible region.

Table 4. The iteration step using MATLAB.

μ	x_{min}	f_{min}	aug_{min}
1.0	(2 .000000000000, 3.000000000000)	-8.0471895622	0.9528104378
10.0	(0.584383413070, 4.869701406514)	-11.851026032	2.8191586927
100.0	(0.491210873054, 3.912290440909)	-8.9702340739	-6.6115965813
1000.0	(0.478587312848, 3.786849015552)	-8.5400072779	-8.2873616072
10000.0	(0.477272005299, 3.773806675110)	-8.4944671131	-8.4690191379
100000.0	(0.477139883021, 3.772497114149)	-8.4898858321	-8.4873391868
1000000.0	(0.477126692178, 3.772366078499)	-8.4894274297	-8.4891727436
10000000.0	(0.477125354746, 3.772352991728)	-8.4893815863	-8.4893561184

Optimum solution point using Mathematica is x = (0.472776, 3.80538) and
Optimum solution is at f^* = -8.52761
Optimum solution point using MATLAB is x = (0.477125354746, 3.772352991728) and
Optimum solution is at f^* = -8.4893815863

The graph of the feasible region and steps of a computer program (Deumlich 1996) with the contours of the objective function are shown in Figure 5. And the iteration step using MATLAB for penalty method and the necessary data are given as follows:

Initial:
$\mathbf{x}_1$ = [2; 3];
μ = 1; beta = 10;
tol = 1.0e-4; tol1 = 1.0e-6; h = 0.1;N = 10 (Table 4).

Example 5

A new facility is to be located such that the sum of its distance from the four existing facilities is minimized. The four facilities are located at the points (1, 2), (-2, 4), (2, 6), and (-6,-3). If the coordinates of the new facility are x_1 and x_2, suppose that x_1 and x_2 must satisfy the restrictions $x_1 + x_2 = 2$, $x_1 \geq 0$, and $x_2 \geq 0$.

Formulate the problem
Solve the problem by a penalty function method using a suitable unconstrained optimization technique.

Minimize f(x) = $\sqrt{(x_1-1)^2+(x_2-2)^2} + \sqrt{(x_1-2)^2+(x_2-6)^2} + \sqrt{(x_1+2)^2+(x_2-4)^2} + \sqrt{(x_1+6)^2+(x_2+3)^2}$

subject to $x_1 + x_2 = 2$

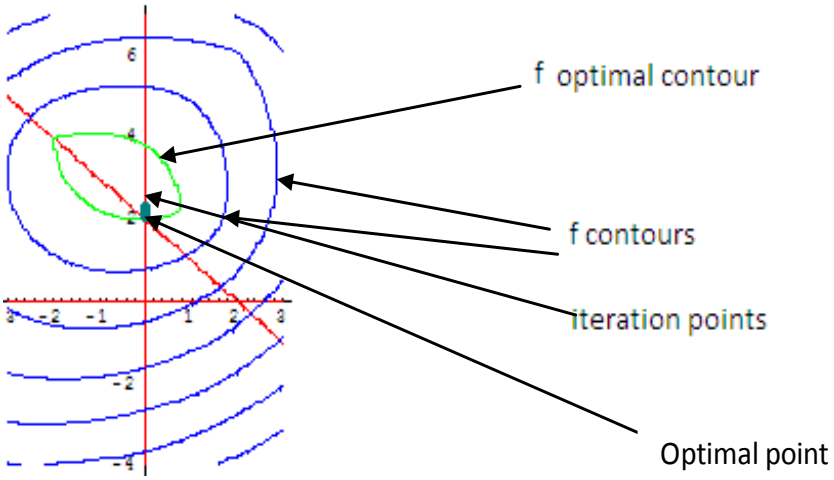

Figure 6. The sequence of unfeasible results from outside the feasible region.

Table 5. The iteration step using MATLAB.

μ	x_{min}	f_{min}	aug_{min}
0.1	(-100000.0000000,-100000.0000000)	565688.2534900	6000645688.6535
1.0	(-0.504816443491, 2.941129889320)	5.6534514664	16.0986605310
10.0	(-0.235317980540, 2.465609185550)	15.7843894418	16.8684753525
100.0	(-0.043496067432, 2.086197027712)	16.0316892765	16.4032172656
1000.0	(-0.004877163871, 2.009622311363)	16.1014887806	16.1477919328
10000.0	(0.000981076620, 2.000347630256)	16.1105064270	16.1281610467
100000.0	(0.003030519805, 1.997102981748)	16.1136901197	16.1154723862
1000000.0	(0.003236508228, 1.996776847061)	16.1140109620	16.1141893257

$x_1 \geq 0$
$x_2 \geq 0$

The corresponding unconstrained optimization problem

is:

$\theta(\mathbf{x}, \mu) = f(x) + \mu[\frac{(\text{maximum}\{0, x_1 + x_2 - 2\})^2 + (\text{maximum}\{0, -x_1\})^2 + (\text{maximum}\{0, -x_2\})^2}{}]$.

Optimum solution point using Mathematica is x $=\{3.087x10^{-32}, 2.02328\}$
Optimum solution is at f^* = 16.0996.
Optimum solution point using MATLAB is x = (0.003236508228, 1.996776847061) and
Optimum solution is at f^* = 16.1140109620.
The graph of the feasible region and steps of a computer program (Deumlich 1996) with the contours of the objective function are shown in Figure 6.
And the iteration step using MATLAB for penalty method and the necessary data are given as follows:

Initial:
x = [-100000; -100000]; μ = 0.1; beta = 10;
tol = 1.0e-6; tol1 = 1.0e-3; h = 0.1;N = 10 (Table 5).

Example 6

A new facility is to be located such that the sum of its distance from the four existing facilities is minimized. The four facilities are located at the points (1, 2), (-2, 4), (2, 6), and (-6,-3). If the coordinates of the new facility are x_1 and x_2, suppose that x_1 and x_2 must satisfy the restrictions $x_1 + x_2 = 2$, $x_1^2 + x_2^2 \leq 2$, $-x_1^2 - 2x_2^2 \leq -3$, $x_1 \geq 0$, and $x_2 \geq 0$.

Formulate the problem
Solve the problem by a penalty function method using a suitable unconstrained optimization technique.

Minimize $f(\mathbf{x}) = \sqrt{(x_1-1)^2+(x_2-2)^2} + \sqrt{(x_1-2)^2+(x_2-6)^2} + \sqrt{(x_1+2)^2+(x_2-4)^2} + \sqrt{(x_1+6)^2+(x_2+3)^2}$
subject to $x_1 + x_2 = 2$
$x_1^2 + x_2^2 \leq 2$
$-x_1^2 - 2x_2^2 \leq -3$
$-x_1 \leq 0$
$-x_2 \leq 0$

The corresponding unconstrained optimization

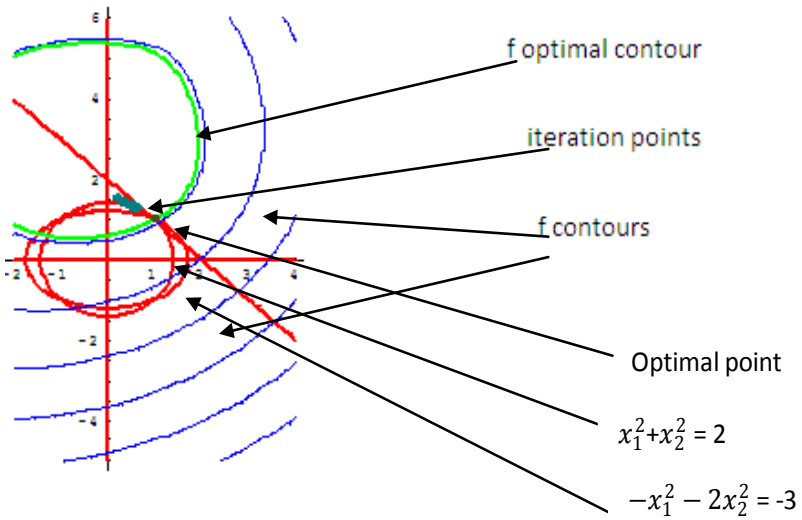

Figure 7. The sequence of unfeasible results from outside the feasible region.

Table 6. The iteration step using MMATLAB.

μ	x_{min}	f_{min}	aug_{min}
0.01	(-100.00000000000,-100.0000000000)	568.62244178640	4000376.7024418
0.10	(-0.204074540511, 2.488615905901)	15.7748455724	17.5805067419
1.00	(-0.005676255570, 1.87710924673)	16.2384983466	18.5763296826
10.0	(0.208853398412, 1.519794991416)	16.7514115057	18.7366197410
100.0	(0.541258580758, 1.337453329403)	17.2205864696	19.3598462433
1000.0	(0.774045018932, 1.191529581681)	17.7152204663	19.2571015793
10000.0	(0.894859224075, 1.097008320463)	18.0550540479	18.8928463977
100000.0	(0.951465363246, 1.046723818830)	18.2363926667	18.6484042954
1000000.0	(0.977561265017, 1.022043406130)	18.3250429734	18.5208297859
10000000.0	(0.989607061234, 1.01030727942)	18.3670763153	18.4588852580

problem is:

$$\theta(\mathbf{x}, \mu) = f(\mathbf{x}) + [(x_1 + x_2 - 2)^2 + (\text{maximum}\{0, -x_1\})^2 + (\text{maximum}\{0, -x_2\})^2 + (\text{maximum}\{0, x_1^2 + x_2^2 - 2\})^2 + (\text{maximum}\{0, -x_1^2 - 2x_2^2 + 3\})^2]$$

Optimum solution point using Mathematica is **x** = {0.624988, 1.28927} and
Optimum solution is at $f^* = 17.579$.
Optimum solution point using MATLAB is **x** = (0.989607061234, 1.010307279416) and
Optimum solution is at $f^* = 18.3670763153$.
The graph of the feasible region and steps of a computer program (Deumlich 1996) with the contours of the objective function are shown in Figure 7.

And the iteration step using MATLAB for penalty method and the necessary data are given as follows:

Initial:

$\mathbf{x_1}$ = [-100; -100];

μ = 0.01; beta = 10;
tol = 1.0e-6; tol1 = 1.0e-3; h = 0.1; N = 10; Table 6

Example 7

The detail of this location problem is given in example 1 of the barrier method.

Minimize

$$3600\sqrt{(x_1)^2 + (x_2 - 2)^2} + 2500\sqrt{(x_1 - 2)^2 + (x_2 - 4)^2} + 1800\sqrt{(x_1 - 5)^2 + (x_2 - 6)^2} + 2200\sqrt{(x_1 - 5)^2 + (x_2 - 10)^2} + 1000\sqrt{(x_1 - 7)^2 + (x_2 - 15)^2} + 4500\sqrt{(x_1 - 10)^2 + (x_2 - 15)^2} + 5600\sqrt{(x_1 - 12)^2 + (x_2 - 10)^2} + 1400\sqrt{(x_1 - 12)^2 + (x_2 - 6)^2} + 1800\sqrt{(x_1 - 15)^2 + (x_2 - 4)^2} + 3000\sqrt{(x_1 - 20)^2 + (x_2 - 2)^2}$$

subject to $x_1^2 + x_2^2 \leq 25$

Table 7. The iteration step using MMATLAB.

μ	x_{min}	fmin	aug_{min}
10.0	(-100.000000,-100.0000000)	4201299.48370	3994823869.4837
100.0	(5.18664531200, 5.613234562710)	217725.1471	335935.8823
1000.0	(3.874734526, 3.94912345612000)	243622.3686	306218.6111
10000.0	(4.28341223, 2.646232345145123)	256672.3825	399517.2275
100000.0	(4.175642351, 0.46232345145100)	293524.5216	342434.4427
1000000.0	(4.01825125, 0.047812341001001)	302445.1718	307672.3810
10000000.0	(4.001831234, 0.00479123410010)	303387.2635	303913.3990
100000000.0	(4.000181234, 0.00047912341001)	303481.9797	303534.6277
1000000000.0	(4.000018123, 0.00004791234100)	303491.4564	303496.7216
10000000000.0	(4.0000018123, 0.0000047912340)	303492.4042	303492.9307
100000000000.0	(4.0000001812, 0.0000004790123)	303492.4989	303492.5516
100000000000.0	(4.000000018, 0.00000004812331)	303492.5084	303492.5137
10000000000000.0	(4.0000000018, 0.0000000048123)	303492.5100	303492.5099
100000000000000.0	(4.000000000185, 0.00000000048)	303492.5095	303492.5095
1000000000000000.0	(4.00000000002, 0.00000000005)	303492.5095	303492.5095

$x_1 + x_2 = 4$
$x_1 - x_2 = 4$
$-x_1 \le 0$
$-x_2 \le 0$

The corresponding unconstrained optimization problem is:

$\theta(\mathbf{x}, \mu)$ = f(x) + $\mu[(\max imum\{0, -x_1\})^2 + (\max imum\{0, -x_2\})^2 + (maximum\{0, x_1^2 + x_2^2 - 25\})^2 + (x_1 + x_2 - 4)^2 + (x_1 - x_2 - 4)^2]$.
Optimum solution point using Mathematica is x = $(4, 1x10^{-12})$
Optimum solution is at f^* = 303493.0.
Optimum solution point using MATLAB is x = (4.000000000018, 0.000000000048) and
Optimum solution is at f^* = 303492.50947.
And the iteration step using MATLAB for penalty method and the necessary data are given as follows:

Initial:

x = [-100; -100];
μ = 10; beta = 10;
tol = 1.0e-6; tol1 = 1.0e-6; h = 0.1; N = 20 (Table 7).

Example 8

Here, we test the well studied welded beam design problem, which has been solved by using a number of classical optimization methods and by using Genetic Algorithms [Deb, 128 to 129]. The welded beam is designed for minimum cost subject to constraints on shear stress in weld (τ), bending stress in the beam (σ), buckling load on the bar (Pc), end deflection of the beam (δ), and side constraints. It has four design variables $\mathbf{x} = (h, l, t, b)$.

Design vector: $\begin{pmatrix} h \\ l \\ t \\ b \end{pmatrix} = \begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{pmatrix}$

Objective function: f (x) = $1.10471x_1x_2 + 0.04811x_3x_4 (14.0 + x_2)$

Constraints:
$g_1(x) = \tau(x) - \tau_{max} \le 0$
$g_2(x) = \sigma(x) - \sigma_{max} \le 0$
$g_3(x) = x_1 - x_4 \le 0$
$g_4(x) = 0.10471x_1 + 0.04811x_3x_4 (14.0 + x_2) - 5.0 \le 0$
$g_5(x) = 0.125 - x_1 \le 0$
$g_6(x) = \delta(x) - \delta_{max} \le 0$
$g_7(x) = P - Pc(x) \le 0$
$g_8(x)$ to $g_{11}(x)$: $0.1 \le x_i \le 2.0$, i = 1, 4
$g_{12}(x)$ to $g_{15}(x)$: $0.1 \le x_i \le 10.0$, i = 2, 3
where

$$\tau(\mathbf{x}) = \sqrt{(\tau'(x))^2 + (\tau''(x))^2 + l\tau'(x)\tau''(x)/\sqrt{0.25[l^2 + (h+t)^2]}}$$

$$\sigma(x) = \frac{504000}{t^2 b},$$

$$p_c(\mathbf{x}) = 64746.022(1 - 0.0282346t)tb^3,$$

$$\delta(\mathbf{x}) = \frac{2.1952}{t^3 b},$$

$$\tau'(\mathbf{x}) = \frac{6000}{\sqrt{2}hl},$$

$$\tau''(\mathbf{x}) = \frac{6000(14 + 0.5l)\sqrt{0.25[l^2 + (h+t)^2]}}{2\left\{0.707hl\left[\frac{l^2}{12} + 0.25(h+t)^2\right]\right\}}$$

Table 8. The iteration step using MMATLAB.

μ	x_{min}	f_{min}	aug_{min}
0.01	(2.00000, 3.00000, 0.100000, 0.050000)	13.2606093500	10160035228635306
0.02	(0.3634872, 2.7826082, 10.558957, 0.232105300)	2.3849380	2.39159576770
0.04	(0.3634872, 2.7826082, 10.55895716, 0.2321053)	2.3849380	2.39825371460
0.08	(0.3738517, 2.8145296, 10.10980684, 0.2340900)	2.3490380	2.35157651770
0.16	(0.375852754, 2.8212375, 10.0249324, 0.234488)	2.3426482	2.34679527470

P = 6000 lb, τ_{max} =13,600 psi, σ_{max} = 30,000 psi, and δ_{max} = 0.25 in.

Starting and optimum solutions:

$x_{start} = \begin{pmatrix} h \\ l \\ t \\ b \end{pmatrix} = \begin{pmatrix} 10 \\ 10 \\ 10 \\ 10 \end{pmatrix}$, f_{start} = 5.398 and **X*** = $\begin{pmatrix} 0.2415 \\ 6.3568, \\ 8.25539 \\ 0.24651 \end{pmatrix}$ and

fsolution^* = $2.3810

Optimum solution point given by Rao (2009) is x = (0.2444, 6.2177, 8.2915, 0.2444) and Optimum solution is at f^* = 2.3810. Optimum solution point using MATLAB is x = (0.375852754, 2.8212375, 10.0249324, 0.234488) and Optimum solution is at f^* = 2.3467952747.

And the iteration step using MATLAB for penalty method and the necessary data are given as follows:

Initial:

x_1 = [2; 3; 0.1; 0.05];
μ = 0.01; beta = 2;
tol = 1.0e-2; tol1 = 1.0e-6; h = 0.1; N = 5 (Table 8).

Using other starting point we have different solution but the difference is not significant as given below.

Initial:

$\mathbf{x}_1$ = [0.4; 6; 0 .01; 0.05];
μ = 0.1; beta = 2;
tol = 1.0e-2; tol1 = 1.0e-6; h = 0.1;N = 30 (Table 9).

EXACT PENALTY FUNCTION METHODS

In this chapter, we analyze two important extensions of the transformation methods, which are called exact penalty functions and have been most frequently used. In these methods a single unconstrained minimization problem, with a reasonable sized penalty parameter can yield an optimum solution to the original problem. This suggests an algorithm which attempts to locate the optimum value of θ whilst keeping μ finite and so avoids the ill-conditioning in the limit μ goes to infinity that we face in penalty function methods.

For the types of penalty functions considered thus far, we have seen that we need to make the penalty parameter infinitely large in a limiting sense to recover an optimal solution. This can cause numerical difficulties and ill-conditioning effects. To alleviate the computational difficulties associated with having to take the penalty parameter to infinity in order to recover an optimal solution to the original problem, we present below two penalty functions that possess this property and are known as exact penalty functions. These are exact absolute value (l_1 penalty function) and augmented Lagrangian penalty function method.

The exact absolute value or l_1 penalty function

An attractive approach to nonlinear programming is to attempt to determine an exact penalty function θ by which is meant a function defined in terms of the objective function and constraints. This holds out the possibility that the solution can be found by a single application of an unconstrained minimization technique to θ, as against the sequential processes described above cannot be used. Consider problem (P) to minimize f(x) subject to $g_i(\mathbf{x}) \leq 0$, i = 1, . . . , m, and $h_i(\mathbf{x}) = 0$, i = 1, . . . , l, and a penalty parameter μ > 0.

Roughly speaking, an exact penalty function for problem (P) is a function $\theta_E(\mathbf{x}, \mu)$, where μ > 0 is the penalty parameter, with the property that there exists a lower bound $\bar{\mu} > 0$ such that for $\mu \geq \bar{\mu}$ any local minimizer of (P) is also a local minimizer of the penalty problem. Exact penalty functions can be divided into two classes: continuously differentiable and non-differentiable exact penalty functions. Continuously differentiable exact penalty functions were introduced by Fletcher (1987) for equality constrained problems and by Gland and Polak (1979) for problems with inequality constraints; further contributions have been assumed in Di Pillo. Non-differentiable exact penalty functions were introduced by Zangwill (1967); Pietrgykowski (1969). The most frequently used type of exact penalty function is the l_1 exact penalty function. This function has been researched widely, for example by Pietrgykowski (1969); Coleman and Conn (1982); in nonlinear programming applications amongst others. Unfortunately the many effective techniques for smooth minimization cannot adequately be used because of its non-differentiability

and the best way of using this penalty function is currently being researched. A more realistic approach is to use this function as a criterion function to be used in conjunction with other iterative methods for nonlinear programming. The most satisfactory approach of all is to apply methods of non-smooth optimization.

A class of non-differentiable exact penalty functions associated to (P) for X = R^n was analyzed by Charalambous in 1978. It is assumed by

$$\theta_q(x, \alpha, \beta) = f(x) + q\left(\sum_{i=1}^{m} [\beta_i g_i^+(\mathbf{x})]^q + \sum_{i=1}^{l} [\alpha_i |h_i(\mathbf{x})|]^q \right)^{\frac{1}{q}},$$

where $q \geq 1$, $\beta_i, \alpha_i > 0$, i = 1, . . . , m and i = 1, . . . , l. For q = 1 and considering all the penalty parameters equal to μ; we have the l_1 penalty function, introduced by Pietrgykowski (1969),

$$\theta_E(\mathbf{x}, \mu) = f(x) + \mu \sum_{i=1}^{m} [\max\{0, g_i(\mathbf{x})\}] + \sum_{i=1}^{l} |h_i(\mathbf{x})|, \quad (4.1)$$

Where:
$p(\mathbf{x}) = \sum_{i=1}^{m} \Phi(g_i(\mathbf{x})) + \sum_{i=1}^{l} \psi(h_i(\mathbf{x}))$
$= \sum_{i=1}^{m} [\max\{0, g_i(\mathbf{x})\}] + \sum_{i=1}^{l} |h_i(\mathbf{x})|$, is the penalty function.

Pietrgykowski (1969), has shown that function (4.1) is exact in the sense that there is a finite μ > 0 such that any regular local minimizer of (P) is also a local minimizer of the penalized unconstrained problem. In 1970, Luenberger showed that, under convex assumptions, there is a lower bound for μ, equal to the largest Lagrange multiplier in absolute value, associated to the nonlinear problem. In 1978, Charalambous generalized the result of Luenberger for the l_1 penalty function (4.1), assuming the second-order sufficient conditions for (P). The following result shows that, under suitable convexity assumptions, there does exist a finite value of μ that will recover an optimum solution to (P) via the minimization of θ_E. Alternatively, it can be shown that if $\bar{\mathbf{x}}$ satisfies the second-order sufficiency conditions for a local minimum of (P) (the Hessian is positive definite). Then, for μ at least as large as the theorem below, $\bar{\mathbf{x}}$ will also be a local minimum of θ_E.

Theorem 4

Consider the following primal problem:

Minimize f(x)
subject to $g(x) \leq 0$
h(x) = 0. (P)

Let $\bar{\mathbf{x}}$ be a KKT point with Lagrangian multipliers $\bar{u}_i$, i ∈ I , and $\bar{v}_i$, i = 1, . . . , l associated with the inequality and equality constraints, respectively, where I = {i ∈ {1, . . . , m} : $g_i(\bar{\mathbf{x}}) = 0$} is the index set of active constraints.

Furthermore, suppose that f and g_i, i ∈ I are convex functions and that h_i, i = 1, . . . , l are affine functions. Then, for μ ≥ maximum {$\bar{u}_i$, i ∈ I, $|\bar{v}_i|$, i = 1, . . . , l}, $\bar{\mathbf{x}}$ also minimizes the exact l_1 penalized objective function θ_E defined by (4.1).

Proof

Since $\bar{\mathbf{x}}$ is a KKT point to (P), it is feasible to (P) and satisfies

$\nabla f(\bar{\mathbf{x}}) + \sum_{i \in I}^{m} \bar{u}_i \nabla g_i(\bar{\mathbf{x}}) + \sum_{i=1}^{m} \bar{v}_i \nabla h_i(\bar{\mathbf{x}}) = 0$, $\bar{u}_i \geq 0$ for i ∈ I (.4.2) (Moreover $\bar{\mathbf{x}}$ solves (P).)

Now, consider the problem of minimizing $\theta_E(\mathbf{x}, \mu)$ over x ∈ R^n. This can equivalently be restated as follows, for any μ ≥ 0:

Minimize $f(x) + \mu\left[\sum_{i=1}^{m} y_i + \sum_{i=1}^{l} z_i\right]$ (4.3a)

subject to $y_i \geq g_i(x)$ and $y_i \geq 0$ for i = 1, . . . , m (4.3b)

$z_i \geq h_i(x)$ and $z_i \geq -h_i(x)$ for i = 1, . . . , l. (4.3c)

The equivalence follows easily by observing that for any x ∈ R^n, the maximum value of the objective function in (4.3a), subject to (4.3b) and (4.3c), is realized by taking y_i = maximum {0, $g_i(\bar{\mathbf{x}})$} for i = 1, . . . , m and $z_i = |h_i(\mathbf{x})|$ for i = 1, . . . , l. In particular, given $\bar{\mathbf{x}}$, define $\bar{y}_i$ = maximum {0, $g_i(\bar{\mathbf{x}})$} for i = 1, …, m and $\bar{z}_i = |h_i(\bar{\mathbf{x}})| \equiv 0$ for i = 1, …, l.

Note that, of the inequalities $y_i \geq g_i(\mathbf{x})$, i = 1, . . . , m, only those corresponding to i ∈ I are binding, while all the other inequalities in (4.3) are binding at $(\bar{\mathbf{x}}, \bar{\mathbf{y}}, \bar{\mathbf{z}})$. Hence, for $(\bar{\mathbf{x}}, \bar{\mathbf{y}}, \bar{\mathbf{z}})$ to be a KKT point for (4.3), we must find Lagrangian multipliers u_i^+, u_i^-, i = 1, . . . , m, and v_i^+, v_i^-, i = 1, . . . , l, associated with the respective pairs of constraints in (4.3b) and (4.3c) such that

$\nabla f(\bar{\mathbf{x}}) + \sum_{i \in I}^{m} u_i^+ \nabla g_i(\bar{\mathbf{x}}) + \sum_{i=1}^{m} (v_i^+ - v_i^-) \nabla h_i(\bar{\mathbf{x}}) = 0$,
$\mu - u_i^+ - u_i^- = 0$ for i = 1, . . . , m,
$\mu - v_i^+ - v_i^- = 0$ for i = 1, . . . , l,
$(u_i^+, u_i^-) \geq 0$ for i = 1, . . . , m,
$(v_i^+, v_i^-) \geq 0$ for i = 1, . . . , l, $u_i^+ = 0$ for i ∉ I.

Assumed that μ ≥ maximum {$\bar{u}_i$, i ∈ I, $|\bar{v}_i|$, i = 1, . . . , l}, we then have, using (4.2), that $u_i^+ = \bar{u}_i$ for all i ∈ I, $u_i^+ = 0$ for i ≠ I, $u_i^- = \mu - u_i^+$ for all i = 1, . . . , m, and $v_i^+ = \frac{(\mu + \bar{v}_i)}{2}$ and $v_i^- = \frac{(\mu - \bar{v}_i)}{2}$ for i = 1, . . . , l satisfy the forgoing KKT conditions. By stated convexity assumptions, it follows that $(\bar{\mathbf{x}}, \bar{\mathbf{y}}, \bar{\mathbf{z}})$ solves (4.3), and, so, $\bar{\mathbf{x}}$ minimizes θ_E. This completes the proof. We proof it as follows in detail:

Lemma 5

Suppose (P) is a convex program for which the Karush-Kuhn-Tucker conditions are necessary. Suppose that

$p(x) = \sum_{i=1}^{m} g_i(x)^+ + \sum_{i=1}^{l} |h_i(x)|$.

Then as long as μ is chosen sufficiently large, the sets of optimal solutions of $\theta_E(\mu)$ and (P) coincide. In fact, it suffices to choose μ > maximum $\{\bar{u}_i, i \in I, |\bar{v}_i|, i = 1, \ldots, l\}$, where $(\bar{u}, \bar{v})$ is a vector of Karush-Kuhn-Tucker multipliers.

Proof

Suppose $\hat{\mathbf{x}}$ solves (P). For any $\mathbf{x} \in R^n$ we have:

$$\theta_E(\mathbf{x}, \mu) = f(\mathbf{x}) + \mu\left(\sum_{i=1}^{m} g_i(\mathbf{x})^+ + \sum_{i=1}^{l} |h_i(\mathbf{x})|\right)$$
$$\geq f(\mathbf{x}) + \sum_{i=1}^{m} \bar{u}_i g_i(\mathbf{x})^+ + \sum_{i=1}^{l} |\bar{v}_i h_i(\mathbf{x})|$$
$$\geq f(\mathbf{x}) + \sum_{i=1}^{m} \bar{u}_i g_i(\mathbf{x}) + \sum_{i=1}^{l} \bar{v}_i h_i(\mathbf{x})$$

$$\geq f(\mathbf{x}) + \sum_{i=1}^{m} \bar{u}_i \left(g_i(\hat{\mathbf{x}}) + (\nabla g_i(\hat{\mathbf{x}}))^T(\mathbf{x} - \hat{\mathbf{x}})\right) +$$
$$\sum_{i=1}^{l} \bar{v}_i (h_i(\hat{\mathbf{x}}) + (\nabla h_i(\hat{\mathbf{x}}))^T(\mathbf{x} - \hat{\mathbf{x}}))$$
$$= f(\mathbf{x}) + \left(\sum_{i=1}^{m} \bar{u}_i \nabla g_i(\hat{\mathbf{x}}) + \sum_{i=1}^{l} \bar{v}_i \nabla h_i(\hat{\mathbf{x}})\right)^T(\mathbf{x} - \hat{\mathbf{x}})$$
$$= f(\mathbf{x}) - \nabla f(\mathbf{x})^T(\mathbf{x} - \hat{\mathbf{x}}) \geq f(\hat{\mathbf{x}})$$
$$= f(\hat{\mathbf{x}}) + \mu\left(\sum_{i=1}^{m} g_i(\hat{\mathbf{x}})^+ + \sum_{i=1}^{l} |h_i(\hat{\mathbf{x}})|\right) = \theta_E(\hat{\mathbf{x}}, \mu).$$

Thus $\theta_E(\hat{\mathbf{x}}, \mu) \leq \theta_E(\mathbf{x}, \mu)$ for all $\mathbf{x}$, and therefore $\hat{\mathbf{x}}$ solves $\theta_E(\mathbf{x}, \mu)$.
Next suppose that $\bar{\mathbf{x}}$ solves $\theta_E(\mathbf{x}, \mu)$. Then if $\hat{\mathbf{x}}$ solves (P), we have:

$$f(\bar{\mathbf{x}}) + \mu\left(\sum_{i=1}^{m} g_i(\bar{\mathbf{x}})^+ + \sum_{i=1}^{l} |h_i(\bar{\mathbf{x}})|\right) \leq f(\hat{\mathbf{x}}) +$$
$$\left(\sum_{i=1}^{m} g_i(\hat{\mathbf{x}})^+ + \sum_{i=1}^{l} |h_i(\hat{\mathbf{x}})|\right) = f(\hat{\mathbf{x}})$$

and so

$$f(\bar{\mathbf{x}}) \leq f(\hat{\mathbf{x}}) - \mu\left(\sum_{i=1}^{m} g_i(\bar{\mathbf{x}})^+ + \sum_{i=1}^{l} |h_i(\bar{\mathbf{x}})|\right). \quad (4.3.1)$$

However, if $\bar{\mathbf{x}}$ is not feasible for (P), then

$$f(\bar{\mathbf{x}}) \geq f(\hat{\mathbf{x}}) + \nabla f(\hat{\mathbf{x}})^T(\bar{\mathbf{x}} - \hat{\mathbf{x}})$$
$$= f(\hat{\mathbf{x}}) - \sum_{i=1}^{m} \bar{u}_i \nabla g_i(\bar{\mathbf{x}} - \hat{\mathbf{x}})) - \sum_{i=1}^{l} \bar{v}_i \nabla g_i(\bar{\mathbf{x}} - \hat{\mathbf{x}})$$
$$\geq f(\hat{\mathbf{x}}) + \sum_{i=1}^{m} \bar{u}_i (g_i(\hat{\mathbf{x}}) - g_i(\bar{\mathbf{x}})) + \sum_{i=1}^{l} \bar{v}_i (h_i(\hat{\mathbf{x}}) - h_i(\bar{\mathbf{x}}))$$
$$= f(\hat{\mathbf{x}}) + \sum_{i=1}^{m} \bar{u}_i g_i(\bar{\mathbf{x}}) - \sum_{i=1}^{l} \bar{v}_i h_i(\bar{\mathbf{x}})$$
$$> f(\hat{\mathbf{x}}) - \left(\sum_{i=1}^{m} g_i(\hat{\mathbf{x}})^+ + \sum_{i=1}^{l} |h_i(\hat{\mathbf{x}})|\right),$$

which contradicts (4.3.1). Thus $\bar{\mathbf{x}}$ is feasible for (P). That being the case,

$$f(\bar{\mathbf{x}}) \leq f(\hat{\mathbf{x}}) - \mu\left(\sum_{i=1}^{m} g_i(\bar{\mathbf{x}})^+ + \sum_{i=1}^{l} |h_i(\bar{\mathbf{x}})|\right) = f(\hat{\mathbf{x}})$$

from (4.3.1.) and so $\bar{\mathbf{x}}$ solves (P). Therefore they have the same optimal value.

Example 1

Minimize $x_1^2 + x_2^2$
subject to $x_1 + x_2 - 1 = 0$

$\bar{\mathbf{x}} = (\frac{1}{2}, \frac{1}{2})^t$ is the KKT point with the Lagrangian multiplier associated with this point is found as:

$\nabla f(\bar{\mathbf{x}}) + \bar{v} \nabla h(\bar{\mathbf{x}}) = 0$, which follows that $2x_1 + \bar{v} = 0$ and $2x_2 + \bar{v} = 0$.

Equating the two we have $\bar{v} = -2\bar{x} = -2(\frac{1}{2}) = -1$.
The function θ_E defined by (4.1) for μ ≥ 0 is:
$\theta_E(\mathbf{x}, \mu) = x_1^2 + x_2^2 + \mu|x_1 + x_2 - 1|$.

If $\mu = 0$, $\theta_E(\mathbf{x}, \mu)$ is minimized at (0, 0). For μ > 0, minimizing $\theta_E(\mathbf{x}, \mu)$ is equivalent to:

Minimizing $x_1^2 + x_2^2 + \mu z$
subject to $-z + x_1 + x_2 - 1 \leq 0$ (4.3.2)
$-z - x_1 - x_2 + 1 \leq 0$

For $(\bar{\mathbf{x}}, \bar{z})$ to be a KKT point for (4.3.2) above, we must find Lagrange multipliers v^+ and v^-, associated with the respective constraints such that:

$$\begin{pmatrix} 2x_1 + (v^+ - v^-) \\ 2x_2 + (v^+ - v^-) \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \end{pmatrix} \quad (4.3.3)$$
$\mu - v^+ - v^- = 0$
$v^+(-z + x_1 + x_2 - 1) = 0$
$v^-(-z - x_1 - x_2 + 1) = 0$
and, moreover, optimality dictates that $z = |x_1 + x_2 - 1|$.

Now let us consider the cases,

Case 1: if $(x_1 + x_2) < 1$, then
$v^+(-z + x_1 + x_2 - 1) = 0$, from this $v^+ = 0$ since $-z + x_1 + x_2 - 1 < 0$.
And, hence (4.3.3) is:

$$\begin{pmatrix} 2x_1 + (-v^-) \\ 2x_2 + (-v^-) \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \end{pmatrix}$$

$\mu = v^+ + v^- = v^-$
$2x_1 - v^- = 0$, and
$2x_2 - v^- = 0$
It follows that $x_1 = \frac{v^-}{2} = \frac{\mu}{2} = x_2$.

This is a KKT point, provided that $0 \leq \mu < 1$.

Case 2: if $x_1 + x_2 = 1$, then $z = |x_1 + x_2 - 1| = 0$.

By (4.3.3)
$$\begin{pmatrix} 2x_1 + (v^+ - v^-) \\ 2x_2 + (v^+ - v^-) \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \end{pmatrix}$$
$\mu - v^+ - v^- = 0$, then

$2x_1 = \frac{(v^- - v^+)}{2} = 2x_2 \quad x_1 = \frac{(v^- - v^+)}{2} = \frac{1}{2} = x_2$.
From this we have;

$v^+ = \mu - v^- = \mu - [v^+ + 1]$ with $v^+ = \frac{\mu-1}{2}$ and $v^- = \frac{\mu+1}{2}$.
This is a KKT point, provided that $\mu \geq 1$.

Case 3: if $(x_1 + x_2) > 1$, so that (4.3.3) is:

$$\begin{pmatrix} 2x_1 \\ 2x_2 \end{pmatrix} + v^+ \begin{pmatrix} 1 \\ 1 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \end{pmatrix} \quad v^- = 0,$$

which implies that $x_1 = \frac{-v^+}{2} = x_2$, and $v^+ = \mu$. Hence, this means that $x_1 + x_2 = -\mu > 1$, a contradiction to $\mu > 0$. Consequently, as μ increases from 0, the minimum of θ_E occurs at $(\frac{\mu}{2}, \frac{\mu}{2})$ until μ reaches the value 1, after which it remains at $(\frac{1}{2}, \frac{1}{2})$, which is the optimum to the original problem.

Augmented lagrangian penalty function (ALAG)

As we have seen in the above discussion, most “smooth” penalty functions (such as quadratic penalty function) never generate exact solutions to the constrained minimization problem. Therefore, we would need to solve the (penalized) unconstrained problems with very large values of the constant μ in order to obtain solutions that are close to being feasible and optimal. (In theory, we need to let $\mu \rightarrow \infty$ to obtain a solution.) This is unfortunate, since the unconstrained optimization problems one encounters in implementing penalty methods tend to become ill-conditioned when μ increases, and therefore, it will be hard to solve each of the unconstrained problems required by the algorithm. Alternatively, one could employ an exact penalty method, that is, a method that guarantees termination at an optimal solution provided that the value of μ is sufficiently large (but finite). As we have established, linear penalty function is an exact penalty function; unfortunately, it is not differentiable at points at the boundary of the feasible region, and therefore poses difficulties in solving corresponding unconstrained problems.

Motivated by our discussion of exact penalty functions, it is natural to raise the question whether we can design a penalty function that not only recovers an exact optimum for finite penalty parameter values but also enjoys the property of being differentiable. The Augmented Lagrangian Penalty Function (ALAG), also known as the multiplier penalty function, is one such exact penalty function. This approach uses both a Lagrangian multiplier term and a penalty term in the auxiliary function. This approach was independently proposed by Hestenes (1969); Powell (1997). The original proposal of this method may be viewed as a significant milestone in the recent history of the constrained optimization area. As described by Hestenes, augmented Lagrangian methods are not only practically important in their own right, but have also served as the starting point for a chain of research developments centering around the use of penalty functions, Lagrange multiplier iterations, and Newton's method for solving the system of necessary optimality conditions. Again, the motivation here is to avoid the ill-conditioning difficulties encountered by the classical approach as the penalty parameter approaches to infinity.

For simplicity, let us begin by discussing the case with only equality constraints, for which augmented Lagrangians are first introduced, and then readily extend the discussion to include inequality constraints as well.

ALAG penalty function for equality constrained problems

Consider Problem (P) of minimizing f(x) subject to $h_i(x) = 0$ for i = 1, . . . , l. we have seen if we employ the quadratic penalty function problem to minimize $f(x) + \sum_{i=1}^{m} h_i^2(\mathbf{x})$, then we typically need to $\mu \rightarrow \infty$ to obtain a constrained minimum for (P). We might then be curious whether, if we were to shift the origin of the penalty term to $\boldsymbol{\theta} = (\theta_i, i = 1, \ldots, l)$ and consider the penalized objective function $f(x) + \sum_{i=1}^{l} [h_i(\mathbf{x}) - \theta_i]^2$ with respect to the problem in which the constraint right-hand sides are perturbed to $\boldsymbol{\theta}$ from 0, it can be shown (Theorem 4.1 below) that if the Lagrange multipliers are fixed at their optimum values $\bar{v}_i$, the minimization of $F_{ALAG}(x, v, \mu)$ gives the solution of the original problem (P) in one step for any value of μ. In such a case there is no need to minimize the function F_{ALAG} for an increasing sequence of values of μ. In expanded form, this latter objective function is

$f(x) - \sum_{i=1}^{m} 2\mu\theta_i h_i(\mathbf{x}) + \mu \sum_{i=1}^{l} h_i^2(\mathbf{x}) + \mu \sum_{i=1}^{m} \theta_i^2$.
Denoting $v_i = -2\mu\theta_i$ for i = 1, . . . , l and dropping the final constant term (independent of x), this can be written as

$$F_{ALAG}(x, v) = f(x) + \sum_{i=1}^{m} v_i h_i(\mathbf{x}) + \mu \sum_{i=1}^{l} h_i^2(\mathbf{x}), \qquad (4.4)$$

where $v \in R^l$ is some vector of multipliers, that can be either kept constant or updated as we proceed with the penalty algorithm. (Compare this to the usual Lagrangian function $L(x, v) = f(x) + \sum_{i=1}^{m} v_i h_i(\mathbf{x})$.) The usage of this function as a penalty function can be partially motivated by the following observation: suppose that $\bar{\mathbf{x}}$ is the optimal solution of (P), and $\bar{\mathbf{v}}$ is the vector of corresponding multipliers. Taking the (partial) gradient of the function $F_{ALAG}(\bar{\mathbf{x}}, \bar{\mathbf{v}})$, we obtain

$$\nabla_x F_{ALAG}(\bar{\mathbf{x}}, \bar{\mathbf{v}}) = \left[\nabla f(\bar{\mathbf{x}}) + \sum_{i=1}^{l} \bar{v}_i \nabla h_i(\bar{\mathbf{x}})\right] + 2\mu \sum_{i=1}^{l} h_i(\bar{\mathbf{x}}) \nabla h_i(\bar{\mathbf{x}}) = 0 \qquad (4.5)$$

For all values of μ; whereas this was not necessary the case with the quadratic penalty function, unless $\nabla f(\bar{\mathbf{x}})$ was

itself zero. Hence, whereas we need to take $\mu \to \infty$ to recover $\bar{\mathbf{x}}$ in a limiting sense using the quadratic penalty function, it is possible that we only need to make μ large enough (under suitable regularity conditions as enunciated below) for the critical point $\bar{\mathbf{x}}$ of $F_{ALAG}(\cdot, \bar{\mathbf{v}})$ to turn out to be its (local) minimizer. In this respect, the last term in (4.5) turns out to be a local convexifier of the overall function.

Observe that the function (4.5) is the ordinary Lagrangian function augmented by the quadratic penalty term; hence the name augmented Lagrangian penalty function. Accordingly, (4.5) can be viewed as the usual quadratic penalty function with respect to the following problem that is equivalent to (P):

$$\text{Minimize } \{f(\mathbf{x}) + \textstyle\sum_{i=1}^{m} v_i h_i(\mathbf{x}) : h_i(\mathbf{x}) = 0 \text{ for } i = 1, l\}. \quad (4.6)$$

Alternatively, (4.4) can be vied as a Lagrangian function for the following problem which is equivalent to (P):

$$\text{Minimize } \{f(\mathbf{x}) + \mu \textstyle\sum_{i=1}^{l} h_i^2(\mathbf{x}) : h_i(\mathbf{x}) = 0 \text{ for } i = 1, l\}. \quad (4.7)$$

inclusion of a “multiplier based term” in the quadratic penalty objective function; it is also sometimes called a multiplier penalty function. These view points lead to a reach theory and algorithmic felicity that is not present in the pure quadratic penalty function.

The following result provides the basis by virtue of which the ALAG penalty function can be classified as an exact penalty function. Namely, if the vector of multipliers $\bar{\mathbf{v}}$ is known, one can hope that under some regularity assumptions, the point $\bar{\mathbf{x}}$ is the local minimizer of $F_{ALAG}(\mathbf{x}, \bar{\mathbf{v}})$ for large (but finite) values of μ.

Theorem 6 (ALAG Theorem)

Consider problem (P) to minimize $f(\mathbf{x})$ subject to $h_i(\mathbf{x}) = 0$ for $i = 1, \ldots, l$, and let the KKT solution $(\bar{\mathbf{x}}, \bar{\mathbf{v}})$ satisfy the second- order sufficiency conditions for a local minimum (the Hessian is positive definite.) Then, there exists a $\bar{\mu}$ such that for $\mu \geq \bar{\mu}$, $F_{ALAG}(\cdot, \bar{\mathbf{v}})$ also achieves a strict local minimum at $\bar{\mathbf{x}}$. In particular, if f is convex and h_i are affine, then any minimizing solution $\bar{\mathbf{x}}$ for (P) also minimizes $F_{ALAG}(\cdot, \bar{\mathbf{v}})$ for all $\mu \geq 0$.

Proof

Since $(\bar{\mathbf{x}}, \bar{\mathbf{v}})$ is a KKT solution, we have, from (4.5), that $\nabla_x F_{ALAG}(\bar{\mathbf{x}}, \bar{\mathbf{v}}) = 0$.

Furthermore, letting $G(\bar{\mathbf{x}})$ denote the Hessian of $F_{ALAG}(\cdot, \bar{\mathbf{v}})$ at $x = \bar{\mathbf{x}}$, we have

$$G(\bar{\mathbf{x}}) = \nabla^2 f(\bar{\mathbf{x}}) + \textstyle\sum_{i=1}^{l} \bar{v}_i \nabla^2 h_i(\mathbf{x}) + 2\mu \sum_{i=1}^{l} [h_i(\bar{\mathbf{x}}) \nabla^2 h_i(\bar{\mathbf{x}}) + \nabla h_i(\bar{\mathbf{x}}) \nabla h_i(\bar{\mathbf{x}})^t]$$
$$= \nabla^2 L(\bar{\mathbf{x}}) + 2\mu \textstyle\sum_{i=1}^{l} \nabla h_i(\bar{\mathbf{x}}) \nabla h_i(\bar{\mathbf{x}})^t \quad (4.8)$$

where $\nabla^2 L(\bar{\mathbf{x}})$ is the Hessian of the Lagrangian function for (P) with a multiplier vector $\bar{\mathbf{v}}$ at $x = \bar{\mathbf{x}}$. From the second-order sufficiency conditions, we know that $\nabla^2 L(\bar{\mathbf{x}})$ is positive definite on the cone

$$C = \{d \neq 0 : \nabla h_i(\bar{\mathbf{x}})^t d = 0 \text{ for } i = 1, \ldots, l\}.$$

Now, on the contrary, if there does not exist a $\bar{\mu}$ such that $G(\bar{\mathbf{x}})$ is positive definite for $\mu \geq \bar{\mu}$, then it must be the case that, given any $\mu_k = k$, $k = 1, \ldots, l$, there exists a $\mathbf{d}_k$ with $\|\mathbf{d}_k\| = 1$ such that

$$\mathbf{d}_k^t G(\bar{\mathbf{x}}) \mathbf{d}_k = \mathbf{d}_k^t \nabla^2 L(\bar{\mathbf{x}}) \mathbf{d}_k + 2k \textstyle\sum_{i=1}^{l} [\nabla h_i(\bar{\mathbf{x}})^t \mathbf{d}_k]^2 \leq 0. \quad (4.9)$$

Since, $\|\mathbf{d}_k\| = 1$ for all k, there exists a convergent subsequence for $\{\mathbf{d}_k\}$ with limit point $\bar{\mathbf{d}}$, where $\|\bar{\mathbf{d}}\| = 1$. Over this subsequence, since the first term in (4.9) approaches $\bar{\mathbf{d}}^t \nabla^2 L(\bar{\mathbf{x}}) \bar{\mathbf{d}}$, a constant, we must have $\nabla h_i(\bar{\mathbf{x}})^t \bar{\mathbf{d}} = 0$ for all $i = 1, \ldots, l$ for (4.9) to hold for all k. Hence, $\bar{\mathbf{d}} \in C$. Moreover, since $\mathbf{d}_k^t \nabla^2 L(\bar{\mathbf{x}}) \mathbf{d}_k \leq 0$ for all k by (4.9), we have $\bar{\mathbf{d}}^t \nabla^2 L(\bar{\mathbf{x}}) \bar{\mathbf{d}} \leq 0$. This contradicts the second-order sufficiency conditions. Consequently, $G(\bar{\mathbf{x}})$ is positive definite for μ exceeding some value $\bar{\mu}$, and so, $\bar{\mathbf{x}}$ is a strict local minimum for $F_{ALAG}(\cdot, \bar{\mathbf{v}})$.

Finally, suppose that f is convex and h_i are affine, and $\bar{\mathbf{x}}$ is optimal to (P). There exists a set of Lagrange multipliers $\bar{\mathbf{v}}$ such that $(\bar{\mathbf{x}}, \bar{\mathbf{v}})$ is a KKT solution. As before, we have $\nabla_x F_{ALAG}(\bar{\mathbf{x}}, \bar{\mathbf{v}}) = 0$, and since for $F_{ALAG}(\cdot, \bar{\mathbf{v}})$ is convex for any $\mu \geq 0$, this completes the proof.

We remark here that without the second-order sufficiency conditions of Theorem 4.3, there might not exist any finite value μ that will recover an optimum $\bar{\mathbf{x}}$ for problem (P), and it might be that we need to take $\mu \to \infty$ for this to occur. The following example from (1987) illustrates this point.

Example 2

Consider the following optimization problem:

$$\text{Minimize } f(x) = x_1^4 + x_1 x_2$$
$$\text{subject to } x_2 = 0$$

$\bar{\mathbf{x}} = (0, 0)^t$ is the optimal solution. From the KKT conditions, $\nabla f(\bar{\mathbf{x}}) + \bar{v} \nabla h_i(\bar{\mathbf{x}}) = 0$ and we get $\bar{v} = 0$ as the unique Lagrange multiplier. Note that:

$L(\bar{\mathbf{x}}) = f(\bar{\mathbf{x}}) + \bar{v}_i h_i(\bar{\mathbf{x}}) = f(\bar{\mathbf{x}})$. Then,

$$\nabla L(\bar{\mathbf{x}}) = \begin{pmatrix} 4x_1^3 + x_2 \\ x_1 \end{pmatrix} \text{ and}$$

$$\nabla^2 L(\bar{\mathbf{x}}) = \begin{bmatrix} 12x_1^2 & 1 \\ 1 & 1 \end{bmatrix}$$

$$\nabla^2 L(0,0) = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} = H$$

The eigenvalues of H are found by solving: |H-λI| = 0, With $\lambda = 1$ Or $\lambda = -1$. Therefore $\nabla^2 L(0,0)$ is indefinite. This shows the second-order sufficiency condition does not hold at $(\bar{x}, \bar{v})$. Now, consider

$F_{ALAG}(\bar{x}, \bar{v}, \mu) = F_{ALAG}(\bar{x}, \mathbf{0}, \mu) = f(\bar{x}) + \sum_{i=1}^{m} \bar{v}_i h_i(\bar{x}) + \mu \sum_{i=1}^{l} h_i^2(\bar{x})$
$= x_1^4 + x_1 x_2 + 0 + \mu x_2^2$.

Note that for any μ > 0

$$\nabla F_{ALAG}(x) = \begin{pmatrix} 4x_1^3 + x_2 \\ x_1 + 2\mu x_2 \end{pmatrix},$$

vanishes at $\bar{x} = (0,0)^t$ and $\hat{x} = (\frac{1}{\sqrt{8\mu}}, \frac{-1}{2\mu\sqrt{8\mu}})^t$. Furthermore,

$\nabla^2 F_{ALAG}(x) = \begin{bmatrix} 12x^2 & 1 \\ 1 & 2\mu \end{bmatrix}$ and $\nabla^2 F_{ALAG}(\bar{x}) = \begin{bmatrix} 0 & 1 \\ 1 & 2\mu \end{bmatrix}$,

is indefinite and, hence, $\bar{x}$ is not a local minimizer for any μ > 0. Hence worth it is assumed that second order sufficient conditions hold and μ is sufficiently large.

However,

$\nabla^2 F_{ALAG}(\hat{x}) = \begin{bmatrix} \frac{3}{2\mu} & 1 \\ 1 & 2\mu \end{bmatrix}$,

The eigenvalues of $\nabla^2 F_{ALAG}(\hat{x})$ are all positive for μ > 0 which shows that $\nabla^2 F_{ALAG}(\hat{x})$ is positive definite, and $\hat{x}$ is in fact the minimizer of F_{ALAG} for all μ > 0. Moreover, as μ → ∞, $\hat{x}$ approaches the constrained minimum for problem (P).

It is demonstrated in the following examples that if the optimum Lagrange multipliers are known, then the solution of this unconstrained problem corresponds to the solution of the original problem regardless of the value of the penalty parameter.

Example 3

Consider the optimization problem (P) in example 4.1.

$\bar{x} = \left(\frac{1}{2}, \frac{1}{2}\right)^t$, with $\bar{v}$ = -1 is the unique KKT point and optimum for this problem. Furthermore, $\nabla^2 L(\bar{x}) = \nabla^2 f(\bar{x})$ is positive definite, and thus second-order sufficiency condition holds at $(\bar{x}, \bar{v})$. Moreover from equation (4.4),

$F_{ALAG}(\bar{x}, \bar{v}) = x_1^2 + x_2^2 - (x_1 + x_2 - 1) + \mu(x_1 + x_2 - 1)^2$
$= (x_1 - \frac{1}{2})^2 + (x_2 - \frac{1}{2})^2 + \mu(x_1 + x_2 - 1)^2 + \frac{1}{2}$,

which is clearly uniquely minimized at $\bar{x} = \left(\frac{1}{2}, \frac{1}{2}\right)^t$ for all μ ≥ 0. Hence, both assertions of Theorem 4.3 are verified.

Example 4

Consider the following optimization problem:

Minimize $f(x) = \frac{1}{2}x^2 + xy^2$

subject to $x + y^2 = 10$

using KKT conditions, it is easy to compute the optimal solutions as follows (This is computed using Mathematica):

KTSolution[$\frac{1}{2}x^2 + xy^2$,{ $x + y^2$ -10= 0},{x,y}];
***** Lagrangian→$\frac{1}{2}x^2 + xy^2$ + [-10+ $x + y^2$]v_1
***** Valid KT point(s) *****
f → 50
x → 10
y → 0 v_1 → -10
Optimum: x = 10, y = 0 Lagrange multiplier, v = -10

In the augmented Lagrangian approach, the unconstrained function is defined by adding exterior penalty term to the Lagrangian of the original problem. Thus we have the following unconstrained function.

$F_{ALAG} = \frac{1}{2}x^2 + xy^2 + v(x + y^2 - 10) + \mu(x + y^2 - 10)^2$

The necessary conditions for the minimum of this function give the following equations:

$$\begin{pmatrix} v + x + y^2 - 20\mu + 2x\mu + 2x^2\mu = 0 \\ 2vy + 2yx - 40\mu + 4xyv + 4y^3v = 0 \end{pmatrix}$$

If v is set to the optimum value of the Lagrange multiplier, we get the following equations:

$$\begin{pmatrix} -10 + x + y^2 - 20\mu + 2x\mu + 2y^2\mu = 0 \\ -20y + 2yx - 40\mu + 4xy\mu + 4y^3\mu = 0 \end{pmatrix}$$

The second equation can be written as follows:

$y(-20 + 2x - 40\mu + 4x\mu + 4y^2\mu)) = 0.$

Thus, y = 0 satisfies this equation for any value μ. Substituting y = 0 in the first equation, we get

$-10 + x - 20\mu + 2x\mu = 0$
or
$(-10 + x)(1 + 2\mu) = 0.$

Thus, x = 10 satisfies this equation. Thus, the Lagrangian penalty function has the property that the optimum solution of the original problem is recovered, if we know the optimum values of the Lagrange multipliers. Therefore, in this sense it is an exact penalty function.

Obviously, when we are solving a problem we don't know the optimum Lagrange multipliers. (If they were known we wouldn't need to spend time in developing new algorithms. We could simply use them with the KKT conditions to get a solution). However, the presence of Lagrange multipliers makes the choice of penalty parameter less critical. In a computational procedure based on the augmented Lagrangian penalty function

method, we start with arbitrary values of Lagrange multipliers and develop a procedure that moves the Lagrange multipliers closer to their optimum values. Thus, near the optimum, the function is not as sensitive to the value of μ_i and the procedure converges to the true optimum.

Therefore, to make use of the above result, one attempts to estimate the multipliers by updating the vector v after solving each (or some) unconstrained minimizations of F_{ALAG}. The outline of such an algorithm is given in the following section.

Schema of an algorithm using augmented Lagrangian penalty functions

Method of multipliers

The method of multipliers is an approach for solving nonlinear programming problems by using the augmented Lagrangian penalty function in a manner that combines the algorithmic aspects of both Lagrangian duality methods and penalty function methods. However, this is accomplished while gaining from both these concepts without being impaired by their respective shortcomings. The method adopts a dual ascent step similar to the sub-gradient optimization scheme for optimizing the Lagrangian dual; but, unlike the latter approach, the overall procedure produces both primal and dual solutions. The primal solution is produced via a penalty function minimization; but because of the properties of the ALAG penalty function, this can usually be accomplished without having to make the penalty parameter infinitely large and, hence, having to contend with the accompanying ill-conditioning effects. Moreover, we can employ efficient derivative based methods in minimizing the penalized objective function. The fundamental scheme of this algorithm is as follows.

Schema of the algorithm for equality constraints

Consider the problem of minimizing f(x) subject to the equality constraints $h_i(x) = 0$ for i = 1, . . . , l. (The extension to include inequality constraints is relatively straight forward and is addressed in the following subsection). Below, we outline the procedure first, and then provide some interpretations, motivations, and implementation comments. As is typically the case, the augmented Lagrangian function employed is of the form (4.4), except that each constraint is assigned its own specific penalty parameter μ_i, instead of a common parameter μ. Hence, constraint violations, and consequent penalizations, can be individually monitored. Accordingly, we replace (4.4) by

$$F_{ALAG}(\mathbf{x}, \mathbf{v}) = f(\mathbf{x}) + \sum_{i=1}^{l} v_i h_i(\mathbf{x}) + \sum_{i=1}^{l} \mu_i h_i^2(\mathbf{x}).$$

Although, there are different algorithms to solve this kind of problems the algorithm due to Powell (1997) is given below and ensures global convergence. The outline of such an algorithm is as follows.

Algorithm 1: Algorithm for ALAG with equality constraints

Initialization: Select some initial Lagrangian multipliers $v = \bar{\mathbf{v}}$ usually 0 and positive values $\mu_1, \ldots, \mu_l$ for the penalty parameters. Let x_o be a null vector, and denote $VIOL(x_o) = \infty$, where for any $x \in R^n$, $VIOL(x) = \text{maximum}\{|h_i(x)| : i = 1, \ldots, l\}$ is a measure of constraint violations. Put k = 1 and proceed to the "inner loop" of the algorithm.

Inner loop (**Penalty function minimization):** Solve minimize $F_{ALAG}(\mathbf{x}, \bar{\mathbf{v}})$ subject to $\mathbf{x} \in R^n$ and let $\mathbf{x}_k$ denote the optimal solution obtained. If $VIOL(x_k) = 0$, stop with x_k as a KKT point. (Practically, one would terminate if $VIOL(x_k)$ is less than some tolerance $\varepsilon > 0$). Otherwise, if $VIOL(x_k) \leq 0.25VIOL(x_{k-1})$, proceed to the outer loop. On the other hand, if $VIOL(x_k) > 0.25VIOL(x_{k-1})$ then, for each constraint i = 1, . . . , l for which $|h_i(x_k)| > 0.25VIOL(x_{k-1})$, replace the corresponding penalty parameter μ_i by $10\mu_i$ and repeat this inner loop step.

Outer loop (**Lagrange multiplier update):** Replace $\bar{\mathbf{v}}$ by $\bar{\mathbf{v}}_{new}$,
Where,

$$(\bar{v}_{new})_i = \bar{v}_i + 2\mu_i h_i(\mathbf{x}_k) \text{ for } i = 1, \ldots, l. \quad (4.10)$$

Increment k by 1, and return to the inner loop.

The inner loop of the forgoing method is concerned with the minimization of the augmented Lagrangian penalty function. For this purpose, we can use x_{k-1} (for $k \geq 2$) as a starting solution and employ Newton's method (with line searches) in case the Hessian is available, or else use a quasi-Newton method if only gradients are available, or use some conjugate gradient method for relatively large-scale problems. If $VIOL(x_k) = 0$, then x_k is feasible, and, moreover,

$$\nabla_x F_{ALAG}(\mathbf{x}_k, \bar{\mathbf{v}}) = \nabla f(\mathbf{x}_k) + \sum_{i=1}^{l} \bar{v}_i \nabla h_i(\mathbf{x}_k) + 2\sum_{i=1}^{l} \mu_i h_i(\mathbf{x}_k) \nabla h_i(\mathbf{x}_k) = 0 \quad (4.11)$$

implies that $\mathbf{x}_k$ is a KKT point. Whenever the revised iterate $\mathbf{x}_k$ of the inner loop does not improve the measure for constraint violations by selected factor 0.25, the penalty parameter is increased by a factor of 10. Hence, the outer loop will be visited after a finite number of iterations when the tolerance $\varepsilon > 0$ is used in the inner

loop, since, as in Theorem 2.2 we have $h_i(\mathbf{x}_k) \to 0$ as $\mu_i \to \infty$ for i = 1, . . . , l.

Observe that the forgoing argument holds regardless of the dual multiplier update scheme used in the outer loop, and that it is essentially related to using the standard quadratic penalty function approach on the equivalent problem (4.6). In fact, if we adopt this view point, then the Lagrange multiplier estimate associated with the constraints in (4.6) is assumed by $2\mu_i h_i(\mathbf{x}_k)$ for i = 1, . . . , l, as (2.8). Since the relationship between the Lagrange multipliers of the original problem (P) and its primal equivalent from (4.6) with $\mathbf{v} = \bar{\mathbf{v}}$ is that the Lagrange multiplier vector for (P) equals $\bar{v}$ plus the Lagrange multiplier vector for (4.6), equation (4.10) then gives the corresponding estimate for the Lagrange multiplier associated with the constraints for (P).

This observation can be reinforced more directly by the following interpretation. Note that having minimized $F_{ALAG}(\mathbf{x}, \bar{v})$, we have (4.11) holding true. However, for $\mathbf{x}_k$ and $\bar{v}$ to be a KKT solution, we want $\nabla_x L(\mathbf{x}_k, \bar{v}) = 0$, where L(x, v) = f(x) + $\sum_{i=1}^{l} v_i h_i(\mathbf{x})$ is the Lagrangian function for (P). Hence, we can choose to revise $\bar{\mathbf{v}}$ to $\bar{\mathbf{v}}_{new}$ in a manner such that

$$\nabla f(\mathbf{x}_k) + \sum_{i=1}^{l} (\bar{v}_{new})_i \nabla h_i(\mathbf{x}_k) = 0.$$

Super imposing this identity on (4.11), we get

$$\nabla_x F_{ALAG}(\mathbf{x}_k, \bar{\mathbf{v}}) = \nabla_x L(\mathbf{x}_k, \bar{\mathbf{v}}),$$

which follows that

$$\nabla f(\mathbf{x}_k) + \sum_{i=1}^{l} \bar{v}_i \nabla h_i(\mathbf{x}_k) + 2\sum_{i=1}^{l} \mu_i h_i(\mathbf{x}_k) \nabla h_i(\mathbf{x}_k) = \nabla f(\mathbf{x}_k) + \sum_{i=1}^{l} (\bar{\mathbf{v}}_{new})_i \nabla h_i(\mathbf{x}_k)$$

by eliminating like terms, we get

$$\sum_{i=1}^{l} [\bar{v}_i + 2\mu_i h_i(x_k)] \nabla h_i(\mathbf{x}_k) = \sum_{i=1}^{l} (\bar{v}_{new})_i \nabla h_i(\mathbf{x}_k).$$

From this,

$(\bar{v}_{new})_i = \bar{v}_i + 2\mu_i h_i(\mathbf{x}_k)$, which is the update scheme in (4.10).

Hence, from the view point of problem (4.6), convergence is obtained above in one of the two ways. First, we might finitely determine a KKT point as is frequently the case. Alternatively, viewing the forgoing algorithm as a one of applying the standard quadratic penalty function approach, in sprit, to the equivalent sequence of problems of the type (4.6), each having particular estimates of the Lagrangian multipliers in the objective function, convergence is achieved by letting the penalty parameters approach infinity. In the latter case, the inner loop problems become increasingly ill-conditioned and second-order methods become imperative.

Example 4.5

Consider the optimization problem of Example 4.1. Given any v, the inner loop of the method of multipliers evaluates $\theta(\mathbf{v}) = \min_x\{F_{ALAG}(\mathbf{x}, \mathbf{v})\}$,
Where $F_{ALAG}(\mathbf{x}, \mathbf{v}) = x_1^2 + x_2^2 + v(x_1 + x_2 - 1) + (x_1 + x_2 - 1)^2$.
Solving $\nabla_x F_{ALAG}(\mathbf{x}, \mathbf{v}) = 0$ yields
$x(v) = \frac{2\mu - v}{2(1+2\mu)}$. (The KKT point)
The outer loop then updates the Lagrangian multiplier according to
$(\bar{v}_{new})_i = \bar{v}_i + 2\mu_i h_i(\mathbf{x}_k)$
which gives $\mathbf{v}_{new} = v + 2\mu[x_1(v) + x_2(v) - 1] = \frac{v - 2\mu}{1+2\mu}$. Note that, as $\mu \to \infty$, $v_{new} \to -1$, the optimal Lagrange multiplier value.
Hence, if we start the algorithm with $\bar{v} = 0$, and $\mu = 1$, the inner loop will determine

$x(0) = \frac{2-0}{2(3)} = (\frac{1}{3}, \frac{1}{3})^t$ with $VIOL[x(0)] = x_1 + x_2 - 1 = \frac{1}{3} + \frac{1}{3} - 1 = \frac{-1}{3}$,

and the outer loop will find $v_{new} = 0 + 2(1)(\frac{-1}{3}) = \frac{-2}{3}$. Next, at the second iteration, the inner loop solution will be obtained as

$x(v) = \frac{2\mu - v}{2(1+2\mu)} = \frac{2(1) - (\frac{-2}{3})}{2(1+2(1))} = \frac{4}{9}$
it follows $x(\frac{-2}{3}) = (\frac{4}{9}, \frac{4}{9})^t$ with $VIOL(x(\frac{-2}{3})) = h[x(\frac{-2}{3})] = \frac{1}{9} > (\frac{1}{4})(\frac{-2}{3})$ with VIOL(x(1)) > VIOL(x(0)).

Hence we will increase μ to 10, and recompute the revised

$\mathbf{x}(\frac{-2}{3}) = \frac{2\mu - v}{2(1+2\mu)} = \frac{2(10) - (\frac{-2}{3})}{2(1+2(10))} = (\frac{31}{63}, \frac{31}{63})^t$ with $VIOL(\mathbf{x}(\frac{-31}{63})) = \frac{-1}{63}$.
The outer loop will then revise the Lagrange multiplier $\bar{v} = \frac{-2}{3}$ to $v_{new} = \frac{-2}{3} + 2(10)(\frac{-1}{63}) = \frac{-62}{63}$.

The iteration will progress in this fashion, using the forgoing formulas, until the constraint violation at the inner loop solution is acceptably small.

ALAG penalty function for problems with mixed constraints

Consider problem (P) to minimize f(**x**) subject to the constraints $g_i(\mathbf{x}) \le 0$ for i = 1, . . . , m and $h_i(\mathbf{x}) = 0$ for i = 1, . . . , l (Bhatti (2000). The extension of the forgoing theory of augmented Lagrangians and the method of multipliers to this case, which also includes inequality constraints, is readily accomplished by equivalently writing the inequalities as the equations $g_i(x) + s_i^2 = 0$ for i = 1, . . . , m. Now suppose that $\bar{\mathbf{x}}$ is a KKT point for problem (P)

with optimal Lagrange multipliers $\bar{u}_i$, i = 1, . . . , m, and $\bar{v}_i$, i = 1, . . . , l, associated with the inequality and equality constraints, respectively, and such that the strict complementary slackness condition holds, namely, that $\bar{u}_i g_i(\bar{x}) = 0$ for all i = 1, . . . , m, with $\bar{u}_i > 0$ for each $i \in I(\bar{x}) = \{i : g_i(\bar{x}) = 0\}$. Furthermore, suppose that the second-order sufficiency condition holds at $(\bar{x}, \bar{u}, \bar{v})$, namely, that $\nabla^2 L(\bar{x})$ is positive definite over the cone

$C = \{d \neq 0 : \nabla g_i(\bar{x})^t \mathbf{d} = 0 \text{ for all } i \in I(\bar{x}), \nabla h_i(\bar{x})^t \mathbf{d} = 0 \text{ for } i = 1, \ldots, l\}$.

Then it can be verified that the conditions of Theorem 4.3 are satisfied for problem $\mathbb{P}'$ to minimize f(x) subject to $g_i(x) + s_i^2 = 0$ for i = 1, . . . , m, and $h_i(x) = 0$ for i = 1, . . . , l, at the solution $(\bar{x}, \bar{s}, \bar{u}, \bar{v})$, where $s_i^2 = - g_i(x)$ for i = 1, . . . , m. Hence, for μ large enough, the solution $(\bar{x}, \bar{s})$ will turn out to be a strict local minimizer for the following ALAG penalty function at (u, v) = $(\bar{u}, \bar{v})$:

$$f(x) + \sum_{i=1}^{m}(g_i(\mathbf{x}) + s_i^2) + \sum_{i=1}^{l} v_i h_i(\mathbf{x}) + \mu\left[\sum_{i=1}^{m}(g_i(\mathbf{x}) + s_i^2)^2 + \sum_{i=1}^{l} h_i^2(\mathbf{x})\right]. \quad (4.12)$$

The augmented Lagrangian function defined in (4.12) includes slack variables for inequality constraints. Their presence increases the number of variables in the problem. It is possible to remove these variables by writing the necessary conditions for the minimum with respect to s. Before proceeding it is convenient to combine the two terms involving the slack variables by noting that;

$$\mu\left[g_i(x) + s_i^2 + \frac{u_i}{2\mu}\right]^2 = \mu\left[(g_i(\mathbf{x}) + s_i^2)^2 + \frac{u_i^2}{4\mu} + 2(g_i(\mathbf{x}) + s_i^2)\left(\frac{u_i}{2\mu}\right)\right].$$

Rearranging the terms gives,

$$\mu[g_i(\mathbf{x}) + s_i^2]^2 + u_i(g_i(x) + s_i^2) = \mu[g_i(x) + s_i^2 + \frac{u_i}{2\mu}]^2 - \frac{u_i^2}{4\mu}.$$

For a given penalty parameter μ > 0, let $\theta(\mathbf{u}, \mathbf{v})$ represent the minimum of (4.12) over (x, s) for any given set of Lagrange multipliers (u, v). Now let us rewrite (4.12) more conveniently as follows:

$$f(x) + \mu\sum_{i=1}^{m}(g_i(\mathbf{x}) + s_i^2 + \frac{u_i}{2\mu})^2 - \sum_{i=1}^{m}\frac{u_i^2}{4\mu} + \sum_{i=1}^{l} v_i h_i(\mathbf{x}) + \mu\sum_{i=1}^{l} h_i^2(\mathbf{x})]. \quad (4.13)$$

Hence, in computing $\theta(\mathbf{u}, \mathbf{v})$, we can minimize (4.13) over (x, s) by first minimizing $[g_i(\mathbf{x}) + s_i^2 + \frac{u_i}{2\mu}]$ over s_i^2 in terms of x for each i = 1, . . . , m and then minimizing the resulting expression over the $x \in R^n$. The former task is accomplished by writing the necessary conditions for the minimum of θ with respect to the slack variables, we get

$\frac{\partial \theta}{\partial s_i} = 0$ implies that

$$2(g_i(\mathbf{x}) + s_i^2 + \frac{u_i}{2\mu})(2s_i) = s_i(g_i(\mathbf{x}) + s_i^2 + \frac{u_i}{2\mu}) = 0, \text{ i} = 1, \ldots, \text{m}.$$

These conditions state that either

$s_i = 0$ Or $g_i(\mathbf{x}) + s_i^2 + \frac{u_i}{2\mu} = 0$ which follows

$s_i = 0$ Or $s_i^2 = -(g_i(\mathbf{x}) + \frac{u_i}{2\mu}) \geq 0$.

Using this $\theta(\mathbf{u}, \mathbf{v})$ can be written as:

$$\theta(\mathbf{u}, \mathbf{v}) = \frac{\text{minimum}_x\{f(\mathbf{x}) + \mu\sum_{i=1}^{m}(\text{maximum}\{g_i(\mathbf{x}) + \frac{u_i}{2\mu}, 0\})^2 - \sum_{i=1}^{m}\frac{u_i^2}{4\mu} + \sum_{i=1}^{l} v_i h_i(\mathbf{x}) + \mu\sum_{i=1}^{l} h_i^2(\mathbf{x})\}}{= \text{minimum}_x\{F_{ALAG}(\mathbf{x}, u, v)\}, \text{ say.}} \quad (4.14)$$

Similar to (4.14), the function $F_{ALAG}(\mathbf{x}, \mathbf{u}, \mathbf{v})$ is sometimes referred to as the ALAG penalty function itself in the presence of both equality and inequality constraints. In particular, in the context of the method of multipliers, the inner loop evaluates $\theta(\mathbf{u}, \mathbf{v})$, measures the constraint violations, and revises the penalty parameter(s) in an identical fashion as before.

In order the augmented Lagrangian penalty function to solve constrained optimization problems, we need to determine a procedure that, starting from arbitrary values, leads to near optimum values of Lagrange multipliers. A simple procedure is based on comparing the necessary conditions for the minimum of the Lagrangian function and the augmented Lagrangian penalty function for problem only with equality constraint. In the presence of inequality constraints, the above analysis does not work out as clearly as for the equality case. In practice, the following rule based on similarity with the equality is adopted:

$(\bar{u}_{new})_i = \bar{u}_i + \text{maximum}\{2\mu g_i(\mathbf{x}_k), -\bar{u}_i\}$.

If $\mathbf{x}_k$ minimizes (4.14), then the sub-gradient component to u_i at (u, v) = $(\bar{u}, \bar{v})$ is found at

$\nabla_{u_i}\theta(\mathbf{u}, \mathbf{v}) = 2\mu\sum_{i=1}^{m}\text{maximum}\left\{g_i(\mathbf{x}) + \frac{u_i}{2\mu}, 0\right\}\frac{1}{2\mu} - 2\sum_{i=1}^{m}\frac{u_i}{4\mu}$

and is

$\text{maximum}\left\{g_i(\mathbf{x}) + \frac{u_i}{2\mu}, 0\right\} - \frac{u_i}{2\mu}$.

Adopting the fixed step length of 2μ along this sub-gradient direction as for the equality constraint case revises u_i to

$$(\bar{u}_{new})_i = \bar{u}_i + 2\mu[\text{maximum}\left\{g_i(\mathbf{x}) + \frac{u_i}{2\mu}, 0\right\} - \frac{u_i}{2\mu}]$$
$$= 0 + \text{maximum}\{2\mu g_i(\mathbf{x}_k) + \bar{u}_i, 0\}$$
$$= \bar{u}_i + \text{maximum}\{2\mu g_i(\mathbf{x}_k), -\bar{u}_i\} \text{ for i} = 1, \ldots, \text{m}. \quad (4.15)$$

To start the process, arbitrary values, usually zero, are assigned to all multipliers. Also, the multiplier updating is done only after a substantial decrease in constraint violation is achieved. The following algorithm from [6] is used in most literatures.

Algorithm 2: Algorithm for ALAG with mixed constraints

Set iteration counter k = 0. Set multipliers u_i = 0, i = 1, …, m and v_i = 0, i = 1,…,l. Set multiplier update counter *l* = 0. Choose a penalty parameter μ and a factor $\beta > 1$ to increase the penalty parameter value during iterations. Typically μ = 10 and β = 2.
Set up the unconstrained minimization problem.

$$\theta(u,v) = \frac{\text{minimum}_x\{f(\mathbf{x}) + \mu\sum_{i=1}^{m}\left(\text{maximum}\left\{g_i(\mathbf{x}) + \frac{u_i}{2\mu}, 0\right\}\right)^2 - \sum_{i=1}^{m}\frac{u_i^2}{4\mu} + \sum_{i=1}^{l} v_i h_i(\mathbf{x}) + \mu\sum_{i=1}^{l} h_i^2(\mathbf{x})\}}{}$$

use a suitable unconstrained minimization problem to find $\mathbf{x}_k$ the minimum. The derivative of θ with respect to x_j are evaluated according to the following:

$$\frac{\partial\theta}{\partial x_j} = \frac{\partial f}{\partial x_j} + 2\mu\sum_{i=1}^{m} r_i\frac{\partial g_i}{\partial x_j} + 2\mu\sum_{i=1}^{m} h_i\frac{\partial h_i}{\partial x_j} + \sum_{i=1}^{m} v_i\frac{\partial h_i}{\partial x_j} + 2\mu\sum_{i=1}^{m}\frac{u_i}{2\mu}, \; j = 1, \ldots, n.$$

Check for convergence: A simple convergence criterion is to stop if all constraints are satisfied and the objective function is not changing much between successive iterations. Thus, stop if the following two criteria are satisfied. Otherwise continue to step (iv).

If $\text{Abs}[(\frac{f(x_k)-f(x_{k-1})}{f(x_k)})] < \varepsilon$

$\text{VIOL}(x_k) < \varepsilon$

$\text{VIOL}(x_k)$ = maximum{Abs($h_i(x)$, i = 1, . . . , l), $g_i(x), i = 1, \ldots, m$} is the maximum constraint violation.
Update the multiplier and the penalty parameter:

If $\text{VIOL}(x_k) \leq 0.25\text{VIOL}(x_{k-1})$ then update the multipliers
$(u_{l+1})_i = (u_l)_i$ + maximum $\{2\mu g_i(\mathbf{x}_k), -\bar{u}_i\}$ i = 1, …, m
$(v_{l+1})_i = (v_l)_i + 2\mu_i h_i(\mathbf{x}_k)$ $i = 1, \ldots, l$
Set *l* = *l* + 1.
Else update the penalty parameter

$\mu_{k+1} = 10\mu_k$.

(v) Update the iteration counter k = k +1 and go back to step (ii).

The ALAG has several advantages. As stated earlier, the penalty parameter need not be increased to infinity for convergence. The starting design vector, x_0, need not be feasible. Finally, it is possible to achieve $g_i(\mathbf{x})$ = 0 and $h_i(\mathbf{x})$ = 0 precisely and the nonzero values of the Lagrange multipliers ($u_i \neq 0$) identify the active constraints automatically. It is to be noted the function F_{ALAG}, assumed by (4.14), is continuous and has continuous first derivatives but has discontinuous second derivatives with respect to x at $g_i(\mathbf{x}) = -\frac{u_i}{2\mu}$. Hence, a second-order methods cannot be used to minimize the function F_{ALAG} (Rao, 2009).

SUMMARY AND CONCLUSION

The intent of this section is to point out some of the key points discussed in the previous chapters; and based on that it aims to draw some conclusions.

As discussed in the previous sections all algorithms for constrained optimization are unreliable to a degree. This fact also holds true in the penalty and function methods.

Penalty methods are among the most powerful class of algorithms available for attacking general nonlinear optimization problems. This statement is supported by the fact that these techniques will converge to at least a local minimum in most cases, regardless of the convexity characteristics of the objective function and constraints. They work well even in the presence of cusps and similar anomalies that can stymie other approaches. Penalty methods approximate a constrained problem that assigns high cost to points that are far from the feasible region. As the approximation is made more exact (by letting) the penalty parameter μ tend to infinity) the solution of the unconstrained penalty problem approaches the solution to the original constrained problem from outside of the active constraints. This method is not used in cases where feasibility must be maintained, for example, if the objective function is undefined or ill-conditioned outside the feasible region.

Penalty methods are quite different than other algorithms that they are not iterative in nature. The definition of $\mathbf{x}_{k+1}$ in no way depends on that of $\mathbf{x}_k$. From this point of view, if one decides to terminate the sequence at the Nth term corresponding to μ_N, obtaining x_N, the calculation of the previous vectors x_1, x_2, …, x_{N-1} is irrelevant, since x_N could have been calculated directly by solving a single unconstrained problem. Indeed, this is the generally the manner that penalty functions are employed; one selects a large value of μ, solves the unconstrained problem, and takes the resulting solution as the final approximate answer to the original problem. It is sometimes recognized, however, that selecting a single large value of μ can lead to difficulty. First, exactly what is a large value relative to a given problem may not be known in advance and consequently an initial trial may produce a solution point that is not close enough to the feasible region in which case μ must be increased. Second, large values of μ yield, as shown in the above sections, ill-conditioned Hessians which in turn imply slow convergence for many algorithms.

A partial remedy to these difficulties is obtained by noting that the search for $\mathbf{x}_{k+1}$ can be initiated from $\mathbf{x}_k$, a starting point that may be fairly close to $\mathbf{x}_{k+1}$. Solution of the k +1th problem will then probably require less time than if the search were initiated from an arbitrary point x_0. For this reason the penalty methods are often regarded as truly iterative algorithms. It has never been determined, however, that solving a sequence of unconstrained problems for increasing value of μ leads to a computational saving over just solving the corresponding to the largest value of μ directly. Indeed,

indications are that it does not. The Hessian matrix of $\theta(x, \mu)$ becomes increasingly ill-conditioned as $\mu \rightarrow \infty$ and the minimization becomes more difficult. That's why the parameter μ should not be increased too quickly and the previous iterate should be used as a starting point. As $\mu \rightarrow \infty$ the Hessian (at the solution) is equal to the sum of L, the Hessian of the Lagrangian associated with the original constrained problem, and a matrix of rank r that tends to infinity (where r is the number of active constraints). This is the fundamental property of these methods.

Though penalty functions are old methods for solving constrained optimization problems, it is, nevertheless, worthy of noticing to recognize the wrong assumption and generalization that everything which is old method is nonsense. We have to be very careful not to trivialize old methods for solving constrained optimization problems and erroneously assume it to be as synonymous to backwardness, as some might misconceive it. In fact, this sequential methods needs to be modified in one way or another so that they would serve for the ever-changing and growing demands of algorithms for certain optimization problems. Though these methods suffer from some computational disadvantages, in the absence of alternative software especially for no-derivative problems they are still recommended. They work well for zero order methods like Powell's method with some modifications and taking different initial points and monotonically increasing parameters.

Finally, In spite of their great initial success, their slow rates of convergence due to ill-conditioning of the associated Hessian led researchers to pursue other approaches. With the advent of interior point methods for linear programming, algorithm designers have taken a fresh look at penalty methods and have been able to achieve much greater efficiency than previously thought possible (Nash and Sofer, 1993).

Exact transformation methods are newer and less well-established as sequential transformation methods and are called the newly established modern penalty methods. Exact transformation methods avoid this long sequence by constructing penalty functions that are exact in the sense that the solution of the penalty problem yields the exact solution to the original problem for a finite value of the penalty parameter. However, it can be shown that such exact functions are not differentiable in most cases. Great consideration should be assumed to the convexity assumption and second-order conditions in using these methods.

ACKNOWLEDGEMENTS

First, the author wishes to acknowledge his family, especially his parents for their unconditional love and faith in him since his birth, without whose support and encouragement, this would not have been a reality.

Also, his warmest and honorable thanks also go to his best friend, Abreha Hailezgi who motivated and told him about his potential, and contributed a lot for the success of this research paper.

Finally, he thanks all his friends who have helped him directly or indirectly in this endeavor, especially those who love him more.

Some notations

The following notations are frequently appearing in this research:

μ = Penalty parameter.
$x = (x_1, x_2, x_3, \ldots, x_n)$ is n-dimensional vector.
$\theta(x, \mu)$ = Unconstrained representation of the primal problem (P).
$\theta(\mu)$ = is the infimum of $\theta(x, \mu)$ with respect to x.
x_μ = A minimum point of $\theta(\mu)$.
X = A nonempty set in R^n.
M(f, S) = Set of minimum points of the constrained optimization problem (P).
M(f, X) = Set of minimum points of the unconstrained optimization problem $\theta(x, \mu)$.
F_{ALAG} = Augmented Lagrangian Penalty Function.
p(x) = Penalty function.
$L_M(\bar{x})$ = L restricted to the subspace M that is tangent to the constraint surface.

REFERENCES

Bazaraa MS, Sherali HD, Shetty CM (2006). Nonlinear Programming: Theory and Algorithms, Second Edition, John Wiley & Sons, New York. pp. 469-500.

Belegundu AD, Chandrupatla TR (1999). Optimization concepts and Applications in Engineering 2nd edition, Pensylvania State University, pp. 278-290.

Bhatti MA (2000). Practical Optimization Methods with Mathematica Applications, Department of Civil and Environmental Engineering University of Iowa, Springer-Verlag New York, Inc. pp. 512-680.

Charalambous CA (1978). Lower Bound for the controling parametres of exact panalty functions, Mathematical Programming, 15:278-290.

Coleman TF, Conn AR (1982). Nonlinear Programming Via an exact penalty function: Asymptotic analysis, Mathematical programming, pp.123-136

Deumlich R (1996). A course in Mathematica, Addis Ababa University, Faculty of science, Department of Mathematics. pp.1-140

Fiacco AV, McCormick GP (1968). Extensions of SUMT for nonlinear programming: Equality constraints and extrapolation. Manage. Sci. 12(11):816-828.

Fletcher R (1987). Practical Methods of Optimization, Second Edition, John Wiley & Sons, New York. pp. 277-318.

Gland ST, Polak E (1979). A multiplier method with Authomatic Limitation of penalty growth. Math. Programming,17:140-155

Hestenes MR (1969). Multiplier and gradient methods. J. Optim. Theory Appl. 4(5):123-136.

Himmelblau DH (1972). Applied Nonlinear Programming, New York, McGraw-Hill, pp. 342-355.

Kiusalaas J (2005). Numerical Methods in Engineering with MATLAB, the Pennsylvania State University, and Cambridge University Press Cambridge, New York, Melbourne, Madrid, Cape Town, Singapore, São Paulo. pp. 391-404.

Luenberger DG (1974). A combined penalty function and Gradient projection method for nonlinear programming. J. Opt. Appl. 14:5.
Luenberger DG (1984). Linear and Nonlinear Programming, 2nd ed., Addison-Wesley Publishing Company, Reading, MA. pp. 401-430.
Nash SG, Sofer A (1993). Linear and Nonlinear Programming, McGraw Hill, New York. pp. 469-765.
Pietrgykowski T (1969). An exact potential method for constrained maxima, SIAM J. Num. Anal. 6:217-238.
Powell MJD (1997). A fast algorithm for nonlinearity constrained optimization calculations, in Lecture Notes in Mathematics, Watson GA et al., Eds., Springer-Verlag, Berlin. pp. 343-357.
Rao SS (2009). Engineering Optimization: Theory and Practice, Fourth Edition, John Wiley & Sons, Inc. pp. 248-318.
Zangwill WI (1967). Nonlinear programming via Penalty Functions. Manage. Sci. 13(5):344-358.

Appendix

General description of the penalty function algorithm

The SUMT iteration involves updating the penalty parameters and initial design vector and calling the unconstrained problem again. In the algorithm Powell's method (which is the zero order method) together with golden-bracket and golden-section method for line minimization is used. The program expects the following files to be available in the path

i. Objective function,
ii. Equality and inequality constraints together,
iii. Unconstrained function,
iv. The flines function (for a line search).

For each an iteration of the penalty method there is an inner iteration of the Powell's method.

The program uses global statements to communicate penalty parameters, initial point, search direction (V), whereas the initial penalty parameters, initial design variable, the number of iterations for penalty method, tolerances for the penalty and Powell's method are given by user automatically.

Several parameters are coded into the program, especially those needed for golden bracket and golden section methods.

Step 0: (Initialization) Choose x_0, number of SUMT iterations (N), penalty parameter (μ), and penalty multiplier (β), tolerance for the penalty method (tol1) and for the Powell's method (tol).
k = 1 (SUMT iteration counter)
Step 1: Start the Powell's method to minimize f(x, μ)
Output x_k.
Step 3: Convergence of exterior penalty method.
Stopping criteria:
$\nabla f = \text{funcnstrained}_k - \text{fobjective}_k$, $\Delta x = x_k - x_{k-1}$.
If $(\nabla f)^2 \le \varepsilon_1$: stop (they have approximately the same solution)
else if $\Delta x^t \Delta x \le \varepsilon_1$: stop (design not changing)
else if k = N_ε : stop (max SUMT iteration reached)
continue
k ← k + 1
$x_k \leftarrow x_k^*$
$\mu_{k+1} = \beta\mu_k$
go to step 2

Input for the welded beam example given in example 8 of penalty function method.

```
function f = obweldedbeam(x) % objective function
f=1.10471*x(1).^2*x(2)+0.04811*x(3)*x(4)*(14+x(2));

function [c,ceq] = conweledbeam(x) % constraints

l=(504000./((x(3)).^2*x(4)));
k=64746.022*(1-0.0282346*x(3))*x(3)*x(4).^3;
m =(2.1952./(x(3).^3*x(4)));
t1=(6000./(sqrt(2)*x(1)*x(2)));
t2=6000*(14+0.5*x(2))*sqrt(0.25*(x(2).^2+(x(1)+x(3)).^2));
t3=2*(0.707*x(1)*x(2)*((x(2).^2/12)+0.25*(x(1)+x(3)).^2));
t=t2./t3;
T=sqrt((t1).^2+(t).^2+((x(2)*t1*t)./sqrt(0.25*(x(2).^2+(x(1)+
x(3)).^2))));
c=[T-13600;l-30000;x(1)-x(4);6000-k;m-0.25;-
x(1)+0.125;x(1)-10;-x(2)+0.1;x(2)-10;...
-x(3)+0.1;x(3)-10;-x(4)+0.1;x(4)-10];
ceq =[]; % no equality constraints.

function z = unconweldedbeam(x,miw) %
The corresponding unconstrained problem
l=(504000./((x(3)).^2*x(4)));
k=64746.022*(1-0.0282346*x(3))*x(3)*x(4).^3;
m =(2.1952./(x(3).^3*x(4)));
t1=(6000./(sqrt(2)*x(1)*x(2)));
t2=6000*(14+0.5*x(2))*sqrt(0.25*(x(2).^2+(x(1)+x(3)).^2));
t3=2*(0.707*x(1)*x(2)*((x(2).^2/12)+0.25*(x(1)+x(3)).^2));
t=t2./t3;
T=sqrt((t1).^2+(t).^2+((x(2)*t1*t)./sqrt(0.25*(x(2).^2+(x(1)+
x(3)).^2))));
z=obweldedbeam(x)+miw*(max(0,T-13600)).^2
+miw*(max(0,l-30000)).^2+...
miw*(max(0,x(1)-x(4))).^2 +miw*(max(0,6000-
k)).^2+miw*(max(0,m-0.25)).^2 +miw*(max(0,-
x(1)+0.125)).^2+...
miw*(max(0,x(1)-10)).^2 +miw*(max(0,-
x(2)+0.1)).^2+miw*(max(0,x(2)-10)).^2 +miw*(max(0,-
x(3)+0.1)).^2+...
miw*(max(0,x(3)-10)).^2 +miw*(max(0,-
x(4)+0.1)).^2+miw*(max(0,x(4)-10)).^2;
```

The MATLAB Code for Penalty Function Method:

```
Function penaltyfunction
% Penalty function method for minimizing f(x1,x2, ..., xn).
% Example for Logarithmic function on Example 8.

% input:
% tol and tol1 are error tolerances for Powell's method
and penalty method respectively.
% x = starting point (vector).
% μ = the penalty parameter.
% beta = the penalty multiplier.
% N = number of iterations for the penalty method, we
choose it depending on the problem.
% h = initial step size used in search for golden bracket.
% output:
% xmin = minimum point.
% objmin = miminum value of objective function.
% augmin = minimum of the corresponding unconstrained
problem

% globals (must be declared global in calling program).
```

```
% V = search direction, the same as the unit vectors in the coordinate directions.

% Starting of the program.
clc; % clears the screen.
clear all; % clears all values of variables for memory advantage.
global x μ V
x = [0.4; 6; 0.01; 0.05];
μ = 0.1; beta = 2;
tol = 1.0e-2; tol1 = 1.0e-6; h = 0.1;N = 30;
if size(x,2) > 1; x = x'; end % x must be column vector
n = length(x); % Number of design variables
df = zeros(n,1); % Decreases of f stored here
u = eye (n); % Columns of u store search directions V
disp(sprintf(' μ xmin objmin augmin '))
disp(sprintf(' ------ ------------------ ------------ --------------- '))
for k=1:N % loop for the penalty function method
[c,ceq]= conweledbeam(x);
obj= obweldedbeam(x);
f= unconweldedbeam(x,μ);
disp(sprintf('%1.5f (%3.12f,%3.12f) %2.10f %2.10f ',μ,x, obj,f))
for j = 1:30 % Allow up to 30 cycles for Powell's method
xold = x;
fold = feval(@unconweldedbeam,xold,μ);
% First n line searches record the decrease of f
for i = 1:n
V = u(1:n,i);
[a,b] = goldbracket(@fline,0.0,h);
[s,fmin] = goldsearch(@fline,a,b);
df(i) = fold - fmin;
fold = fmin;
x = x + s*V;
end
% Last line search in the cycle
V = x - xold;
[a,b] = goldbracket(@fline,0.0,h);
[s,fmin] = goldsearch(@fline,a,b);
x = x + s*V;
if sqrt(dot(x-xold,x-xold)/n) < tol
y = x; % assign the solution to y
end
% Identify biggest decrease of f & update search directions
imax = 1; dfmax = df(1);
for i = 2:n
if df(i) > dfmax
imax= i; dfmax = df(i);
end
end
for i = imax:n-1
u(1:n,i) = u(1:n,i+1);
end
u(1:n,n) = V;
end % end of Powell's method
x=y; % y is the minimum point found using Powell's method in the K^th iteration
μ=beta*μ;
sqrt(dot(f - obj,(f- obj));
if sqrt(dot(f - obj, f - obj)) < tol1
return
end
end % end of SUMT iteration.

% f in the direction of coordinate axes.

function z = flines(s) % f in the search direction V
global x μ V
z = feval(@unconweldedbeam,x+s*V,μ);

% Start of golden bracketing for the minimum.

function [a,b] = goldbracket(func,x1,h)
% Brackets the minimum point of f(x).
% Usage: [a,b] = goldbracket(func,xstart,h)

% input:
% func = handle of function that returns f(x).
% x1 = starting value of x.
% h = initial step size used in search.
% c = a constant factor used to increase the step size h

% output:
% a, b = limits on x at the minimum point.
c = 1.618033989;
f1 = feval(func,x1);
x2 = x1 + h; f2 = feval(func,x2);
% Determine downhill direction and change sign of h if needed.
if f2 > f1
h = -h;
x2 = x1 + h; f2 = feval(func,x2);
% Check if minimum is between x1 - h and x1 + h
if f2 > f1
a = x2; b = x1 - h; return
end
end
% Search loop for the minimum
for i = 1:100
h = c*h;
x3 = x2 + h; f3 = feval(func,x3);
if f3 > f2
a = x1; b = x3; return
end
x1 = x2; x2 = x3; f2 = f3;
end
error('goldbracket did not find a minimum please try another starting point')

% Start of golden search for the minimum.
function [xmin,fmin] = goldsearch(func,a,b,tol2)
% Golden section search for the minimum of f(x).
% The minimum point must be bracketed in a <= x <= b.
% usage: [fmin,xmin] = goldsearch(func,xstart,h).
```

```
% input:
% func = handle of function that returns f(x).
% a, b = limits of the interval containing the minimum.
% tol2 = error tolerance used in golden section.

% output:
% fmin = minimum value of f(x).
% xmin = value of x at the minimum point.
if nargin < 4; tol2 = 1.0e-6; end
nIter = ceil(-2.078087*log(tol2/abs(b-a)));
R = 0.618033989; % R is called golden ratio.
C = 1.0 - R;
% First telescoping
x1 = R*a + C*b;
x2 = C*a + R*b;
f1 = feval(func,x1);
f2 = feval(func,x2);
% Main loop
for i =1:nIter
if f1 > f2
a = x1; x1 = x2; f1 = f2;
x2 = C*a + R*b;
f2 = feval(func,x2);
else
b = x2; x2 = x1; f2 = f1;
x1 = R*a + C*b;
f1 = feval(func,x1);
end
end
if f1 < f2; fmin = f1; xmin = x1;
else
fmin = f2; xmin = x2;
end
```