Constrained optimization: equality constraints

Lluís Garrido – lluis.garrido@ub.edu

November 2018

Abstract

This laboratory is focused on constrained optimization and, in particular, on equality constraints. The constrained optimization problem will be easily transformed into an unconstrained optimization problem using Lagrange multipliers.

1 Equality constraints: KKT conditions

Let us begin with a summary of equality constrained optimization[1].

Consider the problem of minimizing $f(\mathbf{x})$ subject to the constraints $h_i(\mathbf{x}) = 0$ for $i = 1 \dots m$, where $\mathbf{x} \in R^n$. Without any constraint, $m = 0$, the necessary condition for optimality is $\nabla f(\mathbf{x}) = 0$.

Let us now examine the case where $m = 1$, that is, a single constraint. With the constraint $h(\mathbf{x}) = 0$, we require that x lies on the graph of the (nonlinear) equation $h(\mathbf{x})$, see figure 1. Assume that $\mathbf{x}^*$ is the optimal point we are looking for. The steepest descent direction at $\mathbf{x}^*$, $-\nabla f(\mathbf{x}^*)$, is orthogonal to the tangent of the contours of f through the point $\mathbf{x}^*$. A similar reasoning tells us that $\nabla h(\mathbf{x}^*)$ also is orthogonal to the curve $h(\mathbf{x}) = 0$ at $\mathbf{x}^*$. Otherwise, $\mathbf{x}^*$ wouldn't be the optimum, see figure 1.

In addition, $\nabla f(\mathbf{x}^*)$ must be orthogonal to the tangent of the curve $h(\mathbf{x}) = 0$ at $\mathbf{x}^*$. This can be easily demonstrated and here we just give an insight. Assume that $c(t)$ is a curve $\{c = c(t) : t_0 \leq t \leq t_1\}$ such that $h(c(t)) = 0$. In other words, $c(t)$ is the feasible curve with respect to the constraint $h(\mathbf{x}) = 0$. For t^*, we have that $c(t^*) = \mathbf{x}^*$, the optimal value. Observe that $f(c(t))$ has a minimum at $t = t^*$, that is, the derivative of $f(c(t))$ with respect to t must vanish (i.e. be zero) at $t = t^*$. The derivative of $f(c(t))$ with respect to t at $t = t^*$ is

$$\frac{d}{dt} f(c(t))\Big|_{t=t^*} = c'(t)^T \nabla f(c(t))\Big|_{t=t^*} = c'(t^*)^T \nabla f(\mathbf{x}^*)$$

The previous expression indicates us that $\nabla f(\mathbf{x}^*)$ must be orthogonal to the tangent of the curve $h(\mathbf{x}) = 0$ at $\mathbf{x} = \mathbf{x}^*$.

Therefore, $-\nabla f(\mathbf{x}^*)$ and $\nabla h(\mathbf{x}^*)$ must both lie along the same line. That is, for some $\lambda \in R^+$ we must have

$$-\nabla f(\mathbf{x}^*) = \lambda \nabla h(\mathbf{x}^*)$$

Thus, if $\mathbf{x}^*$ is a minimizer, the necessary condition reduces to

$$\nabla_{\mathbf{x},\lambda} f(\mathbf{x}^*) + \lambda \nabla_{\mathbf{x},\lambda} h(\mathbf{x}^*) = 0$$

[1]The text written in this section has been obtained from Griva, I.; Nash, S.; Sofer, A., "Linear and nonlinear optimization", SIAM and http://www.pitt.edu/~jrclass/opt/notes3.pdf.

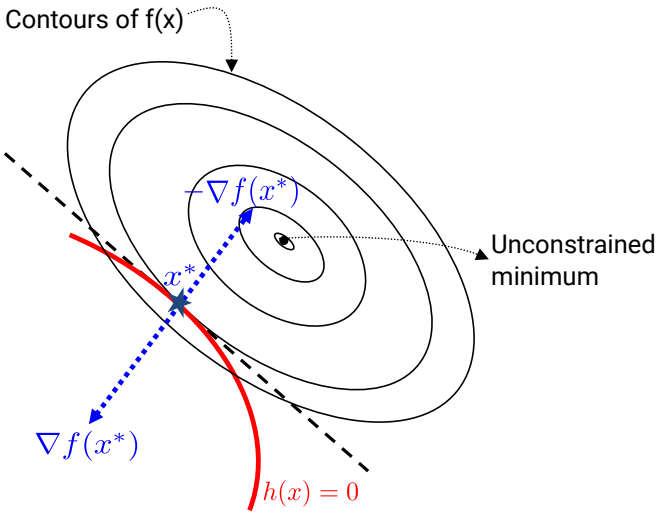

Figure 1: For a single constraint, $m = 1$, we seek a point $\mathbf{x}^*$such that $-\nabla f(\mathbf{x}^*) = \lambda \nabla h(\mathbf{x}^*)$.

where $\nabla_{\mathbf{x},\lambda}$ referers to the gradient computation with respect $\mathbf{x}$ and λ. If not specified, the gradient is computed only with respect to $\mathbf{x}$.

For the general case, in which we have the equality constraints $h_i(\mathbf{x}) = 0$ for $j = 1 \dots m$, the above necessary condition should hold for each constraint. Assuming that $\nabla h_i(\mathbf{x}^*)$ are linearly independent (or equivalenly, the Jacobian matrix has full row rank), the point $\mathbf{x}^* \in R^n$ must satisfy the following necessary condition for some $\lambda^* \in R^m$, that is

$$\nabla_{\mathbf{x},\lambda} f(\mathbf{x}^*) + \sum_{i=1}^{m} \lambda_i \nabla h_i(\mathbf{x}^*) = 0 \qquad h_i(\mathbf{x}^*) = 0 \quad \forall i$$

The latter conditions are known as the **Karush-Kuhn-Tucker** (KKT) conditions. The previous necessary condition can be easily extended to inequalities.

The previous necessary condition may give us the hope that with a simple gradient descent over the Lagrangian the optimum may be obtained. The Lagrangian is

$$\mathcal{L}(\mathbf{x}, \lambda) = f(\mathbf{x}) + \sum_{i=1}^{m} \lambda_i h_i(\mathbf{x})$$

For example, we may insist that

$$\mathcal{L}(x^{k+1}, \lambda^{k+1}) < \mathcal{L}(x^k, \lambda^k)$$

Unfortunately, the optimum value $\mathbf{x}^*$ is in general a saddle point of the Lagrangian – not a minimizer – as the following example shows.

Example

Consider the one dimensional problem of minimizing $f(x) = x^2$ subject to the constraint $x - 1 = 0$. The Lagrangian function is

$$\mathcal{L}(x, \lambda) = x^2 - \lambda(x - 1)$$

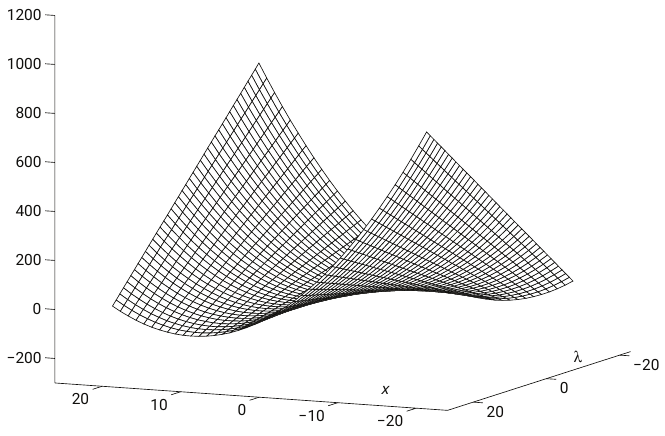

Figure 2: Lagrangian function, see example. This image has been obtained from I. Griva, S.G. Nash, A. Sofer, "Linear and Nonlinear Optimization".

The solution to this problem is $x^* = 1$, with Lagrange multiplier $\lambda^* = 2$. Since $\nabla\mathcal{L}(x, \lambda) = (2x - \lambda, -(x - 1))$, then $\nabla\mathcal{L}(1, 2) = (0, 0)^T$, and indeed (x^*, λ^*) is a stationary point of the Lagrangian. The Hessian matrix of the Lagrangian is

$$\nabla^2\mathcal{L}(x^*, \lambda^*) = \begin{pmatrix} 2 & -1 \\ -1 & 0 \end{pmatrix}$$

This is an indefinite matrix. Thus (x^*, λ^*) is a saddle point of the Lagrangian function, as can be seen in figure 2.

There are two approaches that can be used to ensure convergence. The first approach attempts to ensure that $\mathbf{x}^k$ is feasible at each iteration (i.e. the constraints are satisfied at each iteration). This is the approach that is going to be taken in this lab. The second approach constructs a new function, related to the Lagrangian, that (ideally) has a minimum at $(\mathbf{x}^*, \lambda^*)$. Whereas the former are called feasible-point methods, the second are known as penalty and barrier methods.

2 Sequential Quadratic Programming

Sequential quadratic programming is a popular and successful technique for solving nonlinearly constrained problems. The main idea is to obtain a search direction by solving a quadratic program, that is, a problem with a quadratic objective function and linear constraints.

Our focus is to solve (we focus only on one constraint)

$$\begin{array}{ll} \text{minimize} & f(\mathbf{x}) \\ \text{subject to} & h(\mathbf{x}) = 0 \end{array}$$

The Lagrangian is

$$\mathcal{L}(\mathbf{x}, \lambda) = f(\mathbf{x}) + \lambda h(\mathbf{x})$$

The first order optimality condition is

$$\nabla\mathcal{L}(\mathbf{x}, \lambda) = 0$$

Assume that we use the Newton method to minimize the Lagrange function

$$\begin{pmatrix} \mathbf{x}^{k+1} \\ \lambda^{k+1} \end{pmatrix} = \begin{pmatrix} \mathbf{x}^{k} \\ \lambda^{k} \end{pmatrix} + \alpha^k \begin{pmatrix} \mathbf{d}^{k} \\ \nu^{k} \end{pmatrix} \tag{1}$$

where $\mathbf{d}^k$and ν^k are obtained as the solution to the Newton linear system

$$\nabla^2 \mathcal{L}(\mathbf{x}^k, \lambda^k) \begin{pmatrix} \mathbf{d}^{k} \\ \nu^{k} \end{pmatrix} = -\nabla \mathcal{L}(\mathbf{x}^k, \lambda^k)$$

This linear system has the form

$$\begin{pmatrix} \nabla^2_{xx} \mathcal{L}(\mathbf{x}^k, \lambda^k) & -\nabla h(\mathbf{x}^k) \\ -\nabla h(\mathbf{x}^k)^T & 0 \end{pmatrix} \begin{pmatrix} \mathbf{d}^{k} \\ \nu^{k} \end{pmatrix} = \begin{pmatrix} -\nabla_x \mathcal{L}(\mathbf{x}^k, \lambda^k) \\ h(\mathbf{x}^k) \end{pmatrix} \tag{2}$$

This system of equations represents the first-order optimality conditions for the optimization problem

$$\begin{array}{ll} \text{minimize} & q(\mathbf{d}) = \frac{1}{2}\mathbf{d}^T \left[\nabla^2_{xx} \mathcal{L}(\mathbf{x}^k, \lambda^k)\right] \mathbf{d} + \mathbf{d}^T \left[\nabla_x \mathcal{L}(\mathbf{x}^k, \lambda^k)\right] \\ \text{subject to} & \left[\nabla h(\mathbf{x}^k)^T\right] \mathbf{d} + h(\mathbf{x}^k) = 0 \end{array} \tag{3}$$

This optimization is a quadratic program; that is, it is the minimization of a quadratic function subject to linear constraints. The quadratic function is a Taylor series approximation to the Lagrangian at $(\mathbf{x}^k, \lambda^k)$ and the constraints are a linear approximation to $h(\mathbf{x}^k + \mathbf{d}) = 0$.

In a Sequential Quadratic Optimization method, at each iteration the linear system of equation (2) is solved to obtain $(\mathbf{d}^k, \nu^k)$. These are used to update $(\mathbf{x}^k, \lambda^k)$, and the process repeats at the new point.

Exercise

We apply the Sequential Quadratic Optimization method to the problem

$$\begin{array}{ll} \text{minimize} & f(x_1, x_2) = e^{3x_1} + e^{-4x_2} \\ \text{subject to} & h(x_1, x_2) = x_1^2 + x_2^2 - 1 = 0 \end{array}$$

The solution to this problem is $(x_1^*, x_2^*) \approx (-0.74834, 0.66332)^T$with $\lambda^* \approx -0.21233$, which in fact is a saddle point! The Hessian matrix of the Lagrangian has two positive and one negative eigenvalues.

Let us see how to proceed with the first iteration with the initial guess $(x_1^0, x_2^0) = (-1, 1)^T$ and $\lambda^0 = -1$. At this point the first order derivatives are

$$\begin{aligned} \nabla f &= \begin{pmatrix} 0.14936 \\ -0.07326 \end{pmatrix} \\ \nabla h &= \begin{pmatrix} -2 \\ 2 \end{pmatrix} \\ \nabla_x \mathcal{L} &= \nabla f - \lambda \nabla h = \begin{pmatrix} -1.85064 \\ 1.92674 \end{pmatrix} \end{aligned}$$

and the second order derivatives are

$$\begin{aligned}\nabla^2 f &= \begin{pmatrix} 0.44808 & 0 \\ 0 & 0.29305 \end{pmatrix} \\ \nabla^2 h &= \begin{pmatrix} 2 & 0 \\ 0 & 2 \end{pmatrix} \\ \nabla^2_{xx}\mathcal{L} &= \nabla^2 f - \lambda \nabla^2 h = \begin{pmatrix} 2.44808 & 0 \\ 0 & 2.29305 \end{pmatrix}\end{aligned}$$

The corresponding quadractic program is given by Eq. (3) and its solution can be found using the the optimality conditions given by Eq. (2)

$$\begin{pmatrix} 2.44808 & 0 & 2 \\ 0 & 2.29305 & -2 \\ 2 & -2 & 0 \end{pmatrix} \begin{pmatrix} d_1^0 \\ d_2^0 \\ \nu^0 \end{pmatrix} = \begin{pmatrix} 1.85064 \\ -1.92674 \\ 1 \end{pmatrix}$$

The solution of the quadratic program is

$$\begin{aligned}\begin{pmatrix} d_1^0 \\ d_2^0 \end{pmatrix} &= \begin{pmatrix} 0.22577 \\ -0.27423 \end{pmatrix} \\ \nu^0 &= (0.64896)\end{aligned}$$

We may then perform the update of the parameters $(\mathbf{x}^k, \lambda^k)$ using Eq. (1).

1. One simple way to proceed is to take $\alpha^k = 1$ and update the current point. This is a simple way to proceed that is proposed to perform first. The stopping condition should be performed over $\nabla_x \mathcal{L}$. Test this approach and check if it works.

2. As you may have guessed, this basic iteration also has drawbacks, leading to a number of vital questions. It is a Newton-like iteration, and thus may diverge from poor starting points. In this example we have started from a point that is near to the optimal solution. Try to perform some experiments with starting points that are farther away of the optimal solution.

3. The obvious way to solve the problems that arise at point 2 is to find a good value for α^k using a suitable merit function that allows to approach the optimal value. Usually, a merit function is the sum of terms that include the objective function and the amount of infeasibility of the constraints. One example of a merit function for this example is the quadratic penalty function (i.e. constraints are penalized quadratically)

   $$\mathcal{M}(x_1, x_2) = f(x_1, x_2) + \rho h(x_1, x_2)^2$$

   where ρ is some positive number. The greater the value of ρ, the greater the penalty for infeasibility. The difficulty arises in defining a proper merit function for a particular equality constrained problem. In this exercise we propose you to take $\rho = 10$ and perform the next steps to find the solution to the constrained problem

   (a) Compute the search direction using Eq. (2).

(b) Compute the value of α^k using the backtracking algorithm using the merit function.

(c) Perform the update using Eq. (1).

The proposed method has some drawbacks, among we may mention the fact that the minimizers of the merit function $\mathcal{M}(x_1, x_2)$ do not necessarily have to coincide with the minimizers of the constrained problem. One way to proceed is to use the merit function method to approach the solution. Once we have approached it "sufficiently" the Newton method ($\alpha = 1$) may be used to find the solution to the problem.

Therefore the next experiments are proposed

(a) Implement the minimization method using the Merit function without the additional Newton method. Does the method arrive to the same minimum as the previous Newton method?

(b) Implement the minimization method using the Merit function using the Newton method once we have approached "sufficienly" the minimum. How do you know that you have "sufficiently" approached the minimum? Do you know arrive to a better solution?

Report

You are asked to deliver an *individual* report (PDF, notebook, or whatever else you prefer) of the proposed exercise. Comment each of the steps you have followed as well as the results and plots you obtain. Do not expect the reader (i.e. me) to interpret the results for you. I would like to see if you are able to understand the results you have obtained.

If you want to include some parts of code, please include it within the report. Do not include it as separate files. You may just deliver the Python notebook if you want.