4. Constrained Optimization

Quite often the solution we are seeking comes with strings attached. We do not just want to minimize some function, but we need to do so subject to certain constraints. Let's see a realistic example.

Assume we are tracking the projections of N points in a sequence of images and we want to compute the velocity of the points. We consult the visual motion literature and we decide to use a certain function $Q(u, v)$ that is minimized by the most probable u and v. But we also know that the scene is a rigid scene, which gives us a powerful constraint, since a rigid motion is a special kind of motion that gives rise to specific patterns of velocities for the projections of the points. So what we have is a minimization tempered by a constraint. These kinds of minimizations appear in many disciplines of Science and Engineering as well Political Science, Economics etc.

The simplest way to do the minimization is to use the constraint to solve for one or more of the unknowns, and eliminate it from the minimization. This is the method of choice when such elimination is possible. Unfortunately, it is not always.

4.1. Lagrange Multipliers

One of the most popular ways to do constraint minimization is Lagrange Multipliers. It works like magic, ones feels difficulty believing it when one sees it but it has been used on an extreme range of things, from flying in the air to sorting your socks. It is the Mary Poppins of methods.

Assume you want to minimize $Q(p)$ where p is a vector of dimension K subject to a constraint $c(q) = 0$ where $c(q)$ is a vector valued function of dimension M, $M < K$. It can be shown that this constrained minimization is equivalent to performing unconstrained minimization to the following expression

$$L(p, \lambda) = Q(p) + \lambda^T c(p) \tag{4.1}$$

where λ is a vector of dimension M. We now have $K + M$ unknowns, the vectors p and λ and an equal number of equations. We solve for the unknowns, throw away the vector λ and keep p. That's all? Yes, that's all.

Yet, as opposed to Mary Poppins, the method is not just perfect in every way. If we eliminated M unkowns by using the constraint $c(q) = 0$ we would have $K - M$ left. Now we have $K + M$. Moreover, as we shall see later, some of the nicer properties that we have been addicted to, are lost with Lagrange multipliers.

Well, magic is not really magic, it is mathematics or science. At least that's what some scientists say. Then how do the Lagrange Multipliers work? The exact proof is way beyond the scope of the text and the patience of the sane among its readers, but a little intuition can be helpful. We do this with a simple two dimensional example where we minimize a function of two variables x and y that is subject to an 1-D constraint on x and y sketched in Fig. 4.1. The constraint $c(x, y) = 0$ is represented by the almost straight line running from top left to about bottom right. The function S we want to minimize is depicted by a few of its level crossings at $S(x, y) = 8, 6, 4, 2$. These level crossings are

usually closed curves. If the constraint was absent S would achieve its minimum somewhere in the inner oval, but since we are obliged to choose a solution that satisfies the constraint, we have to move up and down the constraint curve $c(x, y,) = 0$ until we find a minimum. The minimum on the constraint curve is where this curve touches a level crossing. In this example this level crossing is depicted by a dotted line. So the minimum is achieved right at the point of contact.

Now that we know that the minimum is achieved at such a point of contact all we have to do is ask a mathematician to translate this to equations and any competent mathematician will tell us that at such a point the two curves have gradients that are a scalar multiples of each other. If we name this scalar factor $-\lambda$, we will have little difficulty arriving at Eq. (4.1).

4.2. Application to Rayleigh Quotient

We can try the Lagrangian multipliers on something with known answer before we jump head first to something with unknown answer. What better than the Rayleigh Quotient we met a section ago.

$$R = \frac{q^T C q}{q^2}$$

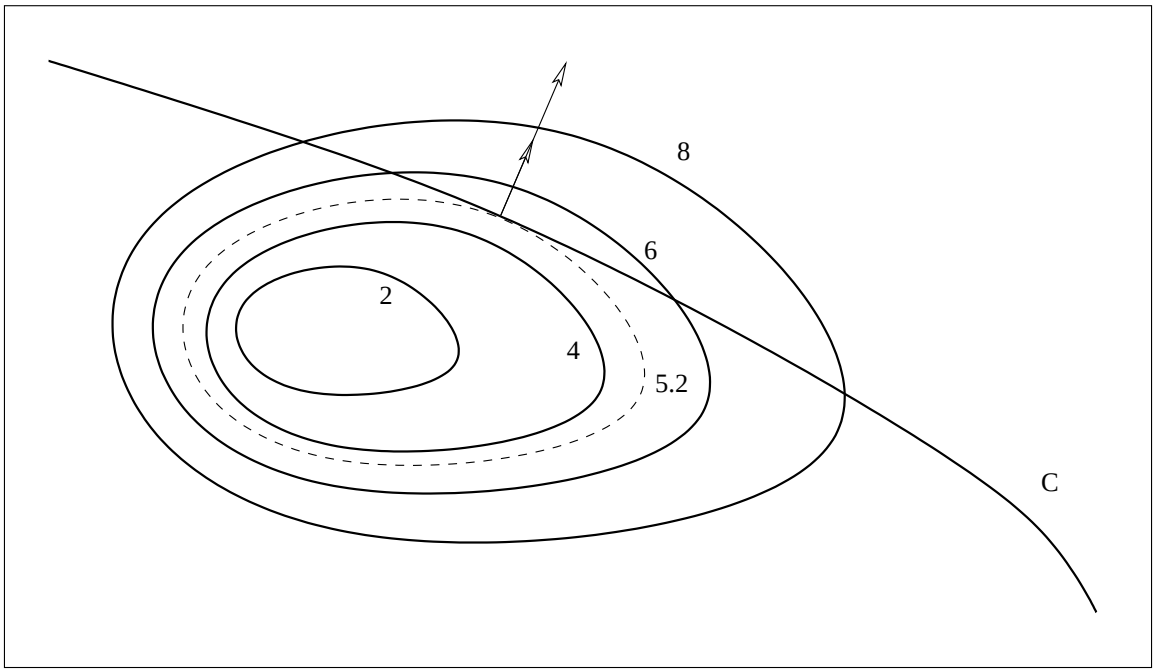

Figure 4.1: The minimum without the constraint should be somewhere in the middle of the innermost contour, but with the constraint the minimum is where the constraint line and the dotted line touch.

can be simplified a bit if we set $s = \frac{q}{|q|}$

$$R = s^T C s \tag{4.2}$$

and we can now find s without nasty denominators and the funny differentiation rules for division. But from the above definition we have the constraint that s is a unit vector, or

$$1 - s^2 = 0. \tag{4.3}$$

Since we have a single constraint, λ is a scalar. The quantity to minimize is

$$L(s, \lambda) = s^T C s + \lambda(1 - s^2). \tag{4.4}$$

Differentiation with respect to λ will give us the original constraint from Eq. (4.3) and differentiation with respect to the original unknowns q will give

$$\frac{\partial}{\partial s} L(s, \lambda) = 2Cs - 2\lambda s$$

which, if we equate to zero we get

$$Cs = \lambda s$$

or that s is an eigenvector of matrix C. Almost done. We know that the solution is an eigenvector but there are as many of them as dimensions in matrix C. So we replace s with eigenvector e_i in Eq. (4.2)

$$R = e_i^T C e_i = \lambda_i e_i^T e_i = \lambda_i$$

and we see that it is equal to the corresponding eigenvalue. So if we want to minimize R then s is the eigenvector corresponding to the smallest eigenvalue. If we want to maximize R, to the largest eigenvalue.

5. Overdetermined Linear Systems

Quite often we have to solve a system of linear equations where we have many more equations than we need but each one is of low quality. If we discard the extra equations and solve the linear system, then we might get low quality results. The solution is nothing less than least squares. How could it be. We are in the chapter about least squares.

As always we take all these equations, put everything in the left hand side, if it is not there already, square them and add them together. Minimizing this sum is a simple issue of differentiating with respect to the unknowns.

Let, then, A be a $M \times N$ matrix where $M > N$, b be a M-dimensional known vector and x an N-dimensional vector of unknowns. The quest is to find the best possible solution to

$$Ax = b$$

which we do by differentiating the squared norm of $Ax - b$, which is nothing more than the sum of the squares of the elements of $Ax - b$

$$\frac{\partial}{\partial x}(Ax-b)^2 = \frac{\partial}{\partial x}\left((Ax-b)^T(Ax-b)\right) =$$

$$2\left(\frac{\partial}{\partial x}(Ax-b)\right)^T(Ax-b) = 2A^T(Ax-b) = 2A^TA - A^Tb$$

which we equate to zero and get

$$A^TAx = A^Tb$$

what statisticians call “normal equations”[‡].

Matrix A^TA has a host of nice properties. It is symmetric, it is non-negative definite (and if invertible positive definite) and requires less storage than the original matrix A and more often than not we do not even need to compute matrix A as an intermediate result at all. If, at the set up stage of the problem the rows A_i of A and the corresponding elements b_i of b (e.g. the individual equations and the corresponding knowns) are produced successively then A^TA and A^Tb can be computed by

$$A^TA = \sum_i A_i A_i^T$$

and

$$A^Tb = \sum_i A_i b_i$$

both of which can easily be done incrementally.

The definiteness of A^TA makes it easy to invert and any matrix inversion method performs better on this than other non-definite equations. And as if this was not enough, there are methods best suited for such normal equations, most notably *Conjugate Gradient*, *Cholesky Factorization*, *Singular Value Decomposition*, *Successive Overrelaxation* etc. The variety is stunning if not truly disheartening to anyone that has never heard any of these methods. But hold this pill. Rather few of these are needed to survive in Computer Vision.

5.1. Overdetermined System with Additional Constraints

Let’s look at a problem that combines the two previous techniques. We have an overdetermined system of linear equations and we want to solve it subject to a single (scalar) linear constraint. It should not be hard, and in a sense it is not.

Let, then as before, A be a $M \times N$ matrix where $M > N$, b be a M-dimensional known vector, x an N-dimensional vector of unknowns and c an N-dimensional known vector. We want to minimize

$$(Ax-b)^2$$

[‡]since there is no mention in the literature of any abnormal equations, one can speculate that normal here means orthogonal.

subject to the constraint

$$c^T x = 0$$

which we know is minimized for the same value of x as

$$L(x, \lambda) = (Ax - b)^2 + \lambda c^T x$$

where λ is an scalar unknown. If we differentiate $L(x, \lambda)$ we get

$$\frac{\partial}{\partial x} L(x, \lambda) = 2A^T(Ax - b) + \lambda c = 0$$

and

$$\frac{\partial}{\partial \lambda} L(x, \lambda) = c^T x = 0$$

which are linear equations and should be easy to solve. As we know any finite system of linear equations can be written in matrix form so we combine the two equations together to get

$$\begin{bmatrix} & . & \\ 2A^T A & . & c \\ & . & \\ \cdots\cdots & . & \cdots \\ c^T & . & 0 \end{bmatrix} \begin{bmatrix} x \\ \\ \cdots \\ \lambda \end{bmatrix} = \begin{bmatrix} 2A^T b \\ \\ \cdots\cdots \\ 0 \end{bmatrix}$$

but unfortunately the matrix in the left hand side is not positive definite. This does not mean that the system is unsolvable, just means that it is much harder. The moral of the story: use Lagrange multipliers for analytic rather than numerical work, or do something about your addiction to positive definiteness.

Exercises

1. Find the extrema of the Rayleigh Quotient without using Lagrangian multipliers.

2. Let A be a symmetric matrix and $\dot{A}$ its time derivative. Find the time derivative of one of its eigenvalue and eigenvector pairs, say λ_0, e_0. No, no, no. That' s too hard. Show that the derivatives are:

$$\dot{\lambda}_0 = e_0^T \dot{A} e_0$$

$$\dot{e}_0 = \sum_{i=1}^{N-1} \frac{e_i e_i^T}{\lambda_0 - \lambda_i} \dot{A} e_0$$

3. Let u_i be 2-D flow vectors measured at locations x_i. In a four dimensional space form the vector

$$v_i = \begin{bmatrix} u_i \\ \cdots \\ x_i \end{bmatrix}$$

and fit a plane through these points. Using this plane express flow as an affine function of x.

4. Let $u[i]$, $i = 1..N$, be a vector of unknowns and

$$u_g[i] = u(*)g[i] = \sum_{j=j_{\min}}^{j_{\max}} u[i-j]g[j]$$

be the convolution of u with convolution kernel or template g. Find the u that minimizes

$$\sum_i \sum_k \left(\alpha[i,k]u[i] + \beta[i,k]u_g[i] + c[i,k] \right)^2$$