Nonlinear Optimization

(COMS 4770/5770 Notes)

Yan-Bin Jia

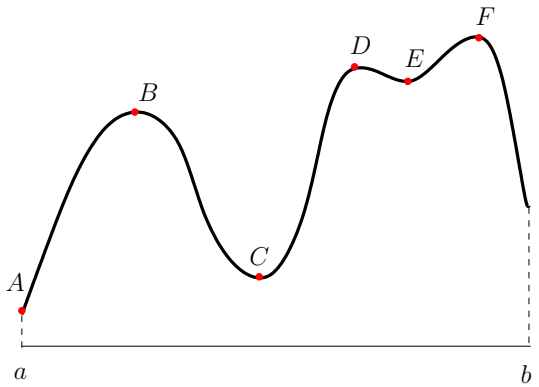
Oct 17, 2024

1 Introduction

Given a single function f that depends on one or more independent variables, we want to find the values of those variables where f is maximized or minimized. Often the computational cost is dominated by the cost of evaluating f (and also perhaps its partial derivatives with respect to all variables).

Finding a global extremum is, in general, a very difficult problem. Two standard heuristics are widely used: i) find local extrema starting from widely varying values of the independent variables, and then pick the most extreme of these; ii) perturb a local extremum by taking a finite amplitude step away from it, and then see if your routine can get to a better point, or "always" to the same one. Recently, "simulated annealing" methods have demonstrated important successes on a variety of global optimization problems.

The diagram below describes a function in an interval $[a, b]$. The derivative vanishes at the points B, C, D, E, F. The points B and D are local but not global maxima. The points C and E are local but not global minima. The global maximum occurs at F where the derivative vanishes. The global minimum occurs at the left endpoint A of the interval so that the derivative need not vanish.



Nonlinear optimization, also referred to as *nonlinear programming* when nonlinear constraints are involved, have many applications. Just to name a few: minimization of energy/power consumption, optimal control, robot path planning in potential fields, thermal and fluid model calibration.

Recall how the optimization process works in one dimension when f is a function from $\mathbb{R}$ to $\mathbb{R}$. First we might compute the critical set of f,

$$c_f = \{\, x \mid f'(x) = 0 \,\}$$

By examining this set we can determine those x that are global minima or maxima. Notice that the computation of c_f seems to entail finding the zeros of the derivative f'. In other words, we have reduced the optimization problem to the root-finding problem. So why do we need to study nonlinear programming? Well, in higher dimensions, it is often easier to find (local) minimum than one would think. Intuitively, this is because f' is not an arbitrary function but rather a derivative whose integral (f) is given. We will thus have a lot more to say about optimization in higher dimension than we did about root finding.

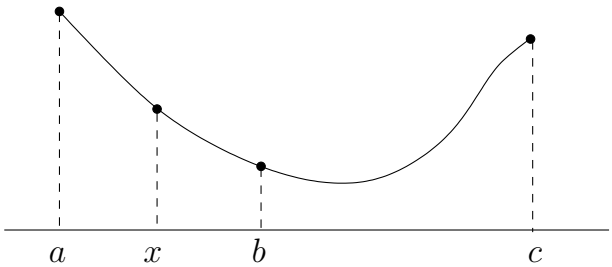
Since a maximization problem can be turned into a minimization problem simply by negating the objective function, we will deal with minimization only from now on.

2 Golden Section Search

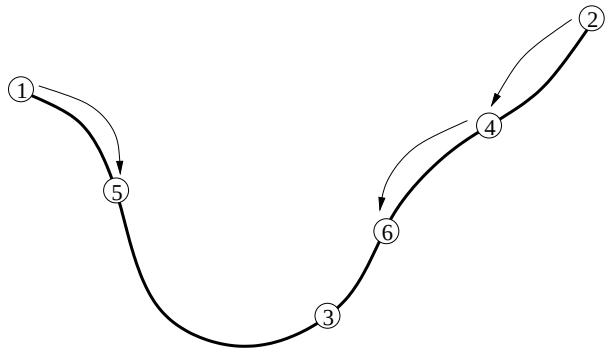
This method bears some analogy to bisection used in root finding. The basic idea is to bracket a minimum and then shrink the bracket. One problem is that we do not know the value of the function at the minimum, so we cannot know whether we have bracketed a minimum. Fortunately, if we are willing to settle for a local minimum, then we really just want to bracket a zero of f'. That we can do, by a numerical approximation to the derivative of f. The inner part of the loop works as follows. We start with three points a, b, c with $a < b < c$ such that

$$f(b) < \min\{f(a), f(c)\}.$$

Now choose a point x, say, half way between a and b. If $f(x) > f(b)$, as shown in the figure below, then the new bracketing triple becomes $[x, b, c]$. If $f(x) < f(b)$, then the new bracketing triple becomes $[a, x, b]$. To ensure that the interval $[a, c]$ will shrink toward a point, *one should alternate the bracketing tuples*, at least after a few rounds. For instance, halve $[a, b]$ in this round while $[b, c]$ in the next round.



The figure on the next page shows a more complete example about how golden section works. The minimum is originally bracketed by points 1, 3, 2. The function is evaluated at 4, which replaces 2; then at 5, which replaces 1; and then at 6, which replaces 4. Note that the center point is always lower than the two outside points. The minimum is bracketed by points 5, 3, 6 after the three steps.



Golden section search is applicable to extremizing functions of one variable only, just like bisection is to finding the roots of functions of this type only.

3 Local and Global Minima

Now we consider functions of more than one variables. Let the function $f : \mathbb{R}^n \to \mathbb{R}$ be twice continuous differentiable, that is, $f \in C^2$. A point $\boldsymbol{x}^* \in \mathbb{R}^n$ is said to be a *local minimum point* or a *relative minimum point* if there is an $\epsilon > 0$ such that $f(\boldsymbol{x}) \geq f(\boldsymbol{x}^*)$ for all $\boldsymbol{x} \in \mathbb{R}^n$ with $\|\boldsymbol{x} - \boldsymbol{x}^*\| < \epsilon$. If $f(\boldsymbol{x}) > f(\boldsymbol{x}^*)$ for all $\boldsymbol{x} \neq \boldsymbol{x}^*$ with $\|\boldsymbol{x} - \boldsymbol{x}^*\| < \epsilon$, then $\boldsymbol{x}^*$ is said to be a *strict local minimum point* of f.

A point $\boldsymbol{x}^*$ is said to be a *global minimum point* of f if $f(\boldsymbol{x}) \geq f(\boldsymbol{x}^*)$ for all $\boldsymbol{x}$. It is said to be a *strict global minimum point* if $f(\boldsymbol{x}) > f(\boldsymbol{x}^*)$ for all $\boldsymbol{x} \neq \boldsymbol{x}^*$.

Let $\boldsymbol{x} = (x_1, x_2, \ldots, x_n)^\top$. The *gradient* of f is represented as a column vector:

$$\nabla f(\boldsymbol{x}) = \begin{pmatrix} \frac{\partial f}{\partial x_1} \\ \frac{\partial f}{\partial x_2} \\ \vdots \\ \frac{\partial f}{\partial x_n} \end{pmatrix} \tag{1}$$

It gives the direction in which the value of f increases the fastest. The *Hessian* of f is defined as an $n \times n$ matrix:

$$\nabla^2 f(\boldsymbol{x}) = \begin{pmatrix} \frac{\partial^2 f}{\partial x_1^2} & \frac{\partial^2 f}{\partial x_1 \partial x_2} & \cdots & \frac{\partial^2 f}{\partial x_1 \partial x_n} \\ \frac{\partial^2 f}{\partial x_2 \partial x_1} & \frac{\partial^2 f}{\partial x_2^2} & \cdots & \frac{\partial^2 f}{\partial x_2 \partial x_n} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial^2 f}{\partial x_n \partial x_1} & \frac{\partial^2 f}{\partial x_n \partial x_2} & \cdots & \frac{\partial^2 f}{\partial x_n^2} \end{pmatrix} \tag{2}$$

With the gradient of a vector (5) defined in Appendix A.1, we have

$$\nabla^2 f = \nabla(\nabla f).$$

The Taylor series at $\boldsymbol{x}^* = (x_1^*, x_2^*, \ldots, x_n^*)^\top$ has the form

$$f(\boldsymbol{x}) \quad = \quad f(\boldsymbol{x}^*) + \left(\frac{\partial f}{\partial x_1}(\boldsymbol{x}^*)(x_1 - x_1^*) + \cdots + \frac{\partial f}{\partial x_n}(\boldsymbol{x}^*)(x_n - x_n^*) \right)$$

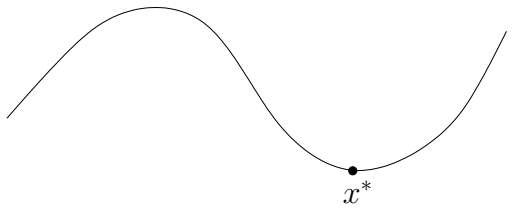
$$+\frac{1}{2}\left(\frac{\partial^2 f}{\partial x_1^2}(\boldsymbol{x}^*)(x_1-x_1^*)^2+\frac{\partial^2 f}{\partial x_1\partial x_2}(\boldsymbol{x}^*)(x_1-x_1^*)(x_2-x_2^*)+\cdots+\frac{\partial^2 f}{\partial x_n^2}(\boldsymbol{x}^*)(x_n-x_n^*)^2\right)$$
$$+\cdots$$
$$= f(\boldsymbol{x}^*)+\nabla f(\boldsymbol{x}^*)^\top(\boldsymbol{x}-\boldsymbol{x}^*)+\frac{1}{2}(\boldsymbol{x}-\boldsymbol{x}^*)^\top\nabla^2 f(\boldsymbol{x}^*)(\boldsymbol{x}-\boldsymbol{x}^*)+O\Big(\|\boldsymbol{x}-\boldsymbol{x}^*\|^3\Big). \tag{3}$$

If $\boldsymbol{x}^*$ is a local minimum, then the following conditions hold:

i) $\nabla f(\boldsymbol{x}^*)=\mathbf{0}$,
ii) $\boldsymbol{d}^\top\nabla^2 f(\boldsymbol{x}^*)\boldsymbol{d}\geq 0$ *for every* $\boldsymbol{d}\in\mathbb{R}^n$.

In one dimension, the above necessary conditions are familiar to us:

$$f'(x^*)=0 \qquad \text{and} \qquad f''(x^*)\geq 0.$$



The Hessian $\nabla^2 f(\boldsymbol{x}^*)$ at a local minimum $\boldsymbol{x}^*$ is symmetric *positive semi-definite*, that is, $\boldsymbol{x}^\top\nabla^2 f(\boldsymbol{x}^*)\boldsymbol{x}\geq 0$ for any $\boldsymbol{x}$. If $\boldsymbol{x}^*$ is a strict local minimum, then $\nabla^2 f(\boldsymbol{x}^*)$ is *positive definite*, that is, $\boldsymbol{x}^\top\nabla^2 f(\boldsymbol{x}^*)\boldsymbol{x}>0$ for any $\boldsymbol{x}\neq\mathbf{0}$. We can now derive sufficient conditions for a local minimum.

If $\nabla f(\boldsymbol{x}^*)=\mathbf{0}$ and $\nabla^2 f(\boldsymbol{x}^*)$ is positive definite, then $\boldsymbol{x}^*$ is a strict local minimum.

EXAMPLE 1 Suppose c is a real number, $\boldsymbol{b}$ is a $n\times 1$ vector, and A is an $n\times n$ symmetric positive definite matrix. Consider the function $f:\mathbb{R}^n\to\mathbb{R}$ given by

$$f(\boldsymbol{x})=c+\boldsymbol{b}^\top\boldsymbol{x}+\frac{1}{2}\boldsymbol{x}^\top A\boldsymbol{x}. \tag{4}$$

It follows that, given the symmetriy of A,

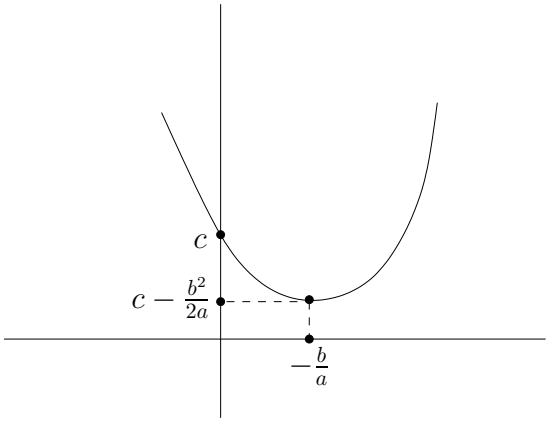
$$\begin{aligned}\nabla f(\boldsymbol{x}) &= \boldsymbol{b}+A\boldsymbol{x},\\ \nabla^2 f(\boldsymbol{x}) &= A.\end{aligned}$$

So there is a single extremum, located at $\boldsymbol{x}^*$, which is the solution to the system $A\boldsymbol{x}=-\boldsymbol{b}$. Since A is positive definite, this extremum is a strict local minimum. Since it is the only one, it is in fact a global minimum.

If we neglect the higher order terms inside the Big-O in the Taylor series (3), every function in one dimension behaves like this (locally near a minimum):

$$\begin{aligned}y &= c+bx+\frac{1}{2}ax^2,\\ y' &= b+ax,\\ y'' &= a > 0.\end{aligned}$$

So $y' = 0$ when $x = -\frac{b}{a}$. This is illustrated in the next figure.



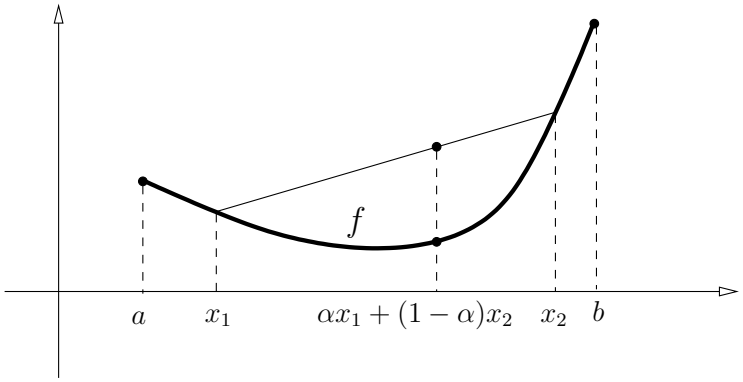
4 Convex Function

We just saw that H is positive definite at and near a strict local minimum. By Taylor's theorem every function looks like a quadratic near a strict local minimum. Furthermore, if f happens to be quadratic globally, formed from a symmetric positive definite matrix A as in (4), then it has a unique local minimum. This local minimum is therefore a global minimum.

Can we say more about functions whose local minima are global minima? A broad class of such functions are the *convex functions*.

A function $f : \Omega \to \mathbb{R}$ defined on a convex domain Ω is said to be *convex* if for every pair of points $\boldsymbol{x}_1, \boldsymbol{x}_2 \in \Omega$ and any α with $0 \leq \alpha \leq 1$, the following holds:

$$f\Big(\alpha \boldsymbol{x}_1 + (1-\alpha)\boldsymbol{x}_2\Big) \leq \alpha f(\boldsymbol{x}_1) + (1-\alpha) f(\boldsymbol{x}_2).$$



Convex functions describe functions near local minima as studied in the following proposition.

Proposition 1 *Let $f \in C^2$. Then f is convex over a convex set Ω containing an interior point if and only if the Hessian matrix H is positive semi-definite in Ω.*

The minima of convex functions are global minima, as shown by the following theorem.

Theorem 2 *Let f be a convex function defined on a convex set Ω. Then the set Γ where f achieves its minimum value is convex. Furthermore, any local minimum is a global minimum.*

Proof If f has no relative minima then the theorem is valid by default. Assume therefore that c_0 is the minimum value of f on Ω. Define the set

$$\Gamma = \{\, \boldsymbol{x} \in \Omega \mid f(\boldsymbol{x}) = c_0 \,\}.$$

Suppose $\boldsymbol{x}_1, \boldsymbol{x}_2 \in \Gamma$ are two values that minimize f. We have that, for $0 \leq \alpha \leq 1$,

$$\begin{aligned} f\Big(\alpha\boldsymbol{x}_1 + (1-\alpha)\boldsymbol{x}_2\Big) &\leq \alpha f(\boldsymbol{x}_1) + (1-\alpha) f(\boldsymbol{x}_2) \\ &= \alpha c_0 + (1-\alpha) c_0 \\ &= c_0. \end{aligned}$$

But c_0 is the minimum, hence the above inequality must be an equality:

$$f\Big(\alpha\boldsymbol{x}_1 + (1-\alpha)\boldsymbol{x}_2\Big) = c_0.$$

In other words, f is also minimized at the point $\alpha\boldsymbol{x}_1 + (1-\alpha)\boldsymbol{x}_2$. Thus all the points on the line segment connecting $\boldsymbol{x}_1$ and $\boldsymbol{x}_2$ are in Γ. Since $\boldsymbol{x}_1$ and $\boldsymbol{x}_2$ are arbitrarily chosen from Γ, the set must be convex.

Suppose now that $\boldsymbol{x}^* \in \Omega$ is a local minimum point of f but not a global minimum. Then there exists some $\boldsymbol{y} \in \Omega$ such that $f(\boldsymbol{y}) < f(\boldsymbol{x}^*)$. On the line $\{\, \alpha\boldsymbol{y} + (1-\alpha)\boldsymbol{x}^* \mid 0 \leq \alpha \leq 1 \,\}$, we have, for $0 < \alpha \leq 1$

$$\begin{aligned} f\Big(\alpha\boldsymbol{y} + (1-\alpha)\boldsymbol{x}^*\Big) &\leq \alpha f(\boldsymbol{y}) + (1-\alpha) f(\boldsymbol{x}^*) \\ &< \alpha f(\boldsymbol{x}^*) + (1-\alpha) f(\boldsymbol{x}^*) \\ &= f(\boldsymbol{x}^*), \end{aligned}$$

contradicting the fact that $\boldsymbol{x}^*$ is a local minimum. □

5 Steepest Descent

Now let us return to the general problem of minimizing a function $f : \mathbb{R}^n \to \mathbb{R}$. We want to find the critical values where the gradient $\nabla f = \mathbf{0}$. This is a system of n equations:

$$\begin{aligned} \frac{\partial f}{\partial x_1} &= 0, \\ &\vdots \\ \frac{\partial f}{\partial x_n} &= 0. \end{aligned}$$

We might expect to encounter the usual difficulties associated with higher-dimensional root finding. Fortunately, the derivative nature of the equations imposes some helpful structure to the problem.

In one dimension, to find a local minimum, we might employ the following rule:

$$\begin{array}{ll} \text{if } f'(x) < 0 & \text{then move to the right;} \\ \text{if } f'(x) > 0 & \text{then move to the left;} \\ \text{if } f'(x) = 0 & \text{then stop.} \end{array}$$

In higher dimensions we use the negative gradient $-\nabla f$ to point us toward a minimum. This is called *steepest descent*. In particular, the algorithm repeatedly performs one-dimensional minimizations along the direction of steepest descent.

In the algorithm, we start with $\boldsymbol{x}^{(0)}$ as an approximation to a local minimum of $f : \mathbb{R}^n \to \mathbb{R}$.

for $m = 0, 1, 2, \ldots$ **until** satisfied **do**
  $\boldsymbol{u} = \nabla f\left(\boldsymbol{x}^{(m)}\right)$
  if $\boldsymbol{u} = \boldsymbol{0}$ **then** stop
    else minimize the function $g(t) = f\left(\boldsymbol{x}^{(m)} - t\boldsymbol{u}\right)$
      let $t^* > 0$ be the closest such minimum to zero
      $\boldsymbol{x}^{(m+1)} \leftarrow \boldsymbol{x}^{(m)} - t^*\boldsymbol{u}$

The method is also referred to as the *line search* strategy since during each iteration it moves on the line $\boldsymbol{x}^{(m)} - t\boldsymbol{u}$ away from $\boldsymbol{x}^{(m)}$ until encountering a local minimum of $g(t)$. How to carry out the line minimization of $g(t)$? Anyway you want. For instance, solve $g'(t) = 0$ directly. Or, step along the line until you produce a bracket, and then refine it.

Example 2. The function

$$f(x_1, x_2) = x_1^3 + x_2^3 - 2x_1^2 + 3x_2^2 - 8$$

has the gradient

$$\nabla f = (3x_1^2 - 4x_1, 3x_2^2 + 6x_2)^\top.$$

Given the guess $\boldsymbol{x}^{(0)} = (1, -1)^\top$ for a local minimum of f, we find

$$\nabla f(\boldsymbol{x}^{(0)}) = (-1, -3)^\top.$$

Thus, in the first step of steepest descent, we look for a minimum of the function

$$\begin{aligned} g(t) &= f\left(\boldsymbol{x}^{(0)} - t\nabla f(\boldsymbol{x}^{(0)})\right) \\ &= f(1 + t, -1 + 3t) \\ &= (1 + t)^3 + (-1 + 3t)^3 - 2(1 + t)^2 + 3(-1 + 3t)^2 - 8. \end{aligned}$$

Setting $g'(t) = 0$ yields the equation

$$\begin{aligned} 0 &= 3(1 + t)^2 + 3(3t - 1)^2 3 - 4(1 + t) + 3 \cdot 2(3t - 1)3 \\ &= 84t^2 + 2t - 10, \end{aligned}$$

which has two solutions

$$t_1 = \frac{1}{3} \qquad \text{and} \qquad t_2 = -\frac{5}{14}.$$

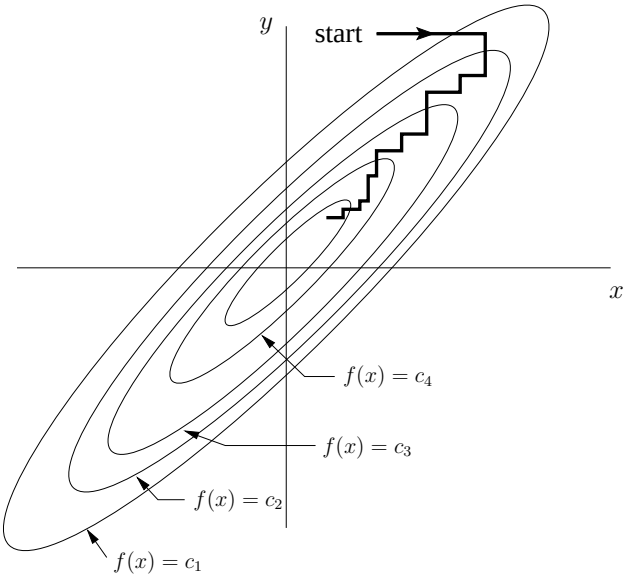
We choose the positive root, $t_1 = \frac{1}{3}$, since we intend to walk from $\boldsymbol{x}^{(0)}$ in the direction of $-\nabla f\left(\boldsymbol{x}^{(0)}\right)$. This gives $\boldsymbol{x}^{(1)} = (\frac{4}{3}, 0)^\top$. The gradient ∇f vanishes at $\boldsymbol{x}^{(1)}$. Therefore f achieves at least a local minimum at $(\frac{4}{3}, 0)^\top$.

It turns out that steepest descent converges globally to a local minimum. And the convergence rate is linear. Let A and a be the largest and smallest eigenvalues, respectively, of the Hessian H at the local minimum. Then the following holds regarding the ratio between the errors at two adjacent steps

$$\frac{|e_{m+1}|}{|e_m|} \sim \left(\frac{A-a}{A+a}\right)^2.$$

The steepest descent method can take many steps. The problem is that the method repeatedly moves in a steepest direction to a minimum along that direction. Consequently, consecutive steps are perpendicular to each other (this behavior is illustrated in the figure below). To see why, consider moving at $\boldsymbol{x}^{(k)}$ along $\boldsymbol{u} = -\nabla f(\boldsymbol{x}^{(k)})$ to reach $\boldsymbol{x}^{(k+1)} = \boldsymbol{x}^{(k)} + t^*\boldsymbol{u}$ where $f(\boldsymbol{x})$ no longer decreases. The rate at which the value of f changes, after a movement of $t\boldsymbol{u}$ from $\boldsymbol{x}^{(k)}$, is measured by the directional derivative, $\nabla f(\boldsymbol{x}^{(k)} + t\boldsymbol{u})^\top \boldsymbol{u}$. This derivative is negative at $\boldsymbol{x}^{(k)}$ (when $t = 0$), and has not changed its sign before $\boldsymbol{x}^{(k+1)}$ (when $t = t^*$). Suppose $\nabla f(\boldsymbol{x}^{(k+1)}) \not\perp \nabla f(\boldsymbol{x}^{(k)})$. Then, $\nabla f(\boldsymbol{x}^{(k+1)})^\top \boldsymbol{u} < 0$ must hold. This means that the value of f would further decrease if continuing the movement in the direction $\boldsymbol{u}$ from $\boldsymbol{x}^{(k+1)}$. Hence a contradiction.

So there are a number of back and forth steps that only slowly converge to a minimum. This situation can get very bad in a narrow valley, where successive steps undo some of their previous progress. Ideally, in $\mathbb{R}^n$ we would like to take n perpendicular steps, each of which attains a minimum. This idea will lead to the conjugate gradient method.



A Matrix Calculus

This appendix presents some basic rules of differentiations of scalars and vectors with respect to vectors and matrices. These rules will be used later on in the course. Throughout the appendix, $\boldsymbol{x} = (x_1, x_2, \ldots, x_n)^\top$ is an n-vector.

A.1 Differentiation With Respect to a Vector

The derivative of a vector function $\boldsymbol{f}(\boldsymbol{x}) = (f_1(\boldsymbol{x}), f_2(\boldsymbol{x}), \ldots, f_m(\boldsymbol{x}))^\top$ with respect to $\boldsymbol{x}$ is an $m \times n$ matrix:

$$\nabla \boldsymbol{f} = \frac{\partial \boldsymbol{f}}{\partial \boldsymbol{x}} = \begin{pmatrix} \frac{\partial f_1}{\partial x_1} & \frac{\partial f_1}{\partial x_2} & \cdots & \frac{\partial f_1}{\partial x_n} \\ \frac{\partial f_2}{\partial x_1} & \frac{\partial f_2}{\partial x_2} & \cdots & \frac{\partial f_2}{\partial x_n} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial f_m}{\partial x_1} & \frac{\partial f_m}{\partial x_2} & \cdots & \frac{\partial f_m}{\partial x_n} \end{pmatrix}. \tag{5}$$

The above matrix is referred to as the *Jacobian* of $\boldsymbol{f}$. It trivially follows from the above that

$$\nabla \boldsymbol{x} = \frac{\partial \boldsymbol{x}}{\partial \boldsymbol{x}} = I_n,$$

where I_n is the $n \times n$ identity matrix.

Let $\boldsymbol{a}$ be a constant n-vector and $A = (\boldsymbol{a}_1, \boldsymbol{a}_2, \ldots, \boldsymbol{a}_m)^\top$ a constant $m \times n$ matrix. Then $\boldsymbol{a}^\top \boldsymbol{x}$ and $A\boldsymbol{x}$ are scalar and vector functions of $\boldsymbol{x}$, respectively. Applying (1) and (5), respectively, we can easily verify the following:

$$\nabla(\boldsymbol{a}^\top \boldsymbol{x}) = \frac{\partial(\boldsymbol{a}^\top \boldsymbol{x})}{\partial \boldsymbol{x}} = \frac{\partial(\boldsymbol{x}^\top \boldsymbol{a})}{\partial \boldsymbol{x}} = \boldsymbol{a}, \tag{6}$$

$$\begin{aligned} \nabla(A\boldsymbol{x}) &= \frac{\partial(A\boldsymbol{x})}{\partial \boldsymbol{x}} = \frac{\partial}{\partial \boldsymbol{x}} \left(\begin{pmatrix} \boldsymbol{a}_1^\top \boldsymbol{x} \\ \vdots \\ \boldsymbol{a}_m^\top \boldsymbol{x} \end{pmatrix} \right) \\ &= \begin{pmatrix} \boldsymbol{a}_1^\top \\ \vdots \\ \boldsymbol{a}_m^\top \end{pmatrix} \qquad (7) \\ &= A. \qquad (8) \end{aligned}$$

Meanwhile, let $\boldsymbol{b} = (b_1, b_2, \ldots, b_m)^\top$ be a constant m-vector. We have

$$\begin{aligned} \nabla(\boldsymbol{b}^\top \boldsymbol{f}(\boldsymbol{x})) &= \frac{\partial(\boldsymbol{b}^\top \boldsymbol{f})}{\partial \boldsymbol{x}} \\ &= \frac{\partial}{\partial \boldsymbol{x}}(b_1 f_1 + b_2 f_2 + \cdots + b_m f_m) \\ &= \nabla \boldsymbol{f}^\top \boldsymbol{b}, \qquad (9) \\ \frac{\partial(\boldsymbol{f}(\boldsymbol{x})^\top \boldsymbol{b})}{\partial \boldsymbol{x}} &= \frac{\partial(\boldsymbol{b}^\top \boldsymbol{f})}{\partial \boldsymbol{x}} \\ &= \nabla \boldsymbol{f}^\top \boldsymbol{b}. \qquad (10) \end{aligned}$$

Consider another vector function $\boldsymbol{g}(\boldsymbol{x}) = (g_1(\boldsymbol{x}), g_2(\boldsymbol{x}), \ldots, g_m(\boldsymbol{x}))^\top$. Applying (9) and (10), we easily obtain the gradient of its inner product with $\boldsymbol{f}$:

$$\nabla(\boldsymbol{f}^\top \boldsymbol{g}) = \nabla \boldsymbol{f}^\top \boldsymbol{g} + \nabla \boldsymbol{g}^\top \boldsymbol{f}. \tag{11}$$

For a scalar function $h(\boldsymbol{x})$ and a vector fuction $\boldsymbol{l}(\boldsymbol{x}) : \mathbb{R}^n \to \mathbb{R}^n$, we differentiate the inner product $\nabla h^\top \boldsymbol{l}$ with respect $\boldsymbol{x}$ as follows, by applying (9) and (10):

$$\begin{aligned} \nabla \left(\nabla h^\top \boldsymbol{l} \right) &= \Big(\nabla(\nabla h) \Big)^\top \boldsymbol{l} + \nabla \boldsymbol{l}^\top \nabla h \\ &= \left(\nabla^2 h \right)^\top \boldsymbol{l} + \nabla \boldsymbol{l}^\top \nabla h. \end{aligned} \tag{12}$$

For a constant $k \times m$ matrix $B = (\boldsymbol{b}_1, \ldots, \boldsymbol{b}_k)^\top$, it follows that

$$\begin{aligned} \nabla(B\boldsymbol{f}) &= \frac{\partial (B\boldsymbol{f})}{\partial \boldsymbol{x}} \\ &= \frac{\partial}{\partial \boldsymbol{x}} \left(\begin{pmatrix} \boldsymbol{b}_1^\top \boldsymbol{f} \\ \vdots \\ \boldsymbol{b}_k^\top \boldsymbol{f} \end{pmatrix} \right) \\ &= \begin{pmatrix} \left(\nabla \boldsymbol{f}^\top \boldsymbol{b}_1 \right)^\top \\ \vdots \\ \left(\nabla \boldsymbol{f}^\top \boldsymbol{b}_k \right)^\top \end{pmatrix} \\ &= \begin{pmatrix} \boldsymbol{b}_1^\top \nabla \boldsymbol{f} \\ \vdots \\ \boldsymbol{b}_k^\top \nabla \boldsymbol{f} \end{pmatrix} \\ &= B \nabla \boldsymbol{f}. \end{aligned} \tag{13}$$

Equations (6) and (13) reduce to equations (10) and (8), respectively, when $\boldsymbol{f}(\boldsymbol{x}) = \boldsymbol{x}$ (and subsequently $\boldsymbol{b}$ and B are identified with $\boldsymbol{a}$ and A, respectively).

Last, we consider a matrix function $C(\boldsymbol{x}) = (\boldsymbol{c}_1, \boldsymbol{c}_2, \ldots, \boldsymbol{c}_k)^\top$ of dimension $k \times m$. The gradient of $C(\boldsymbol{x})\boldsymbol{b}$ can be written out by applying (9) as below:

$$\begin{aligned} \nabla \Big(C(\boldsymbol{x}) \boldsymbol{b} \Big) &= \begin{pmatrix} \left(\nabla (\boldsymbol{c}_1^\top \boldsymbol{b}) \right)^\top \\ \vdots \\ \left(\nabla (\boldsymbol{c}_k^\top \boldsymbol{b}) \right)^\top \end{pmatrix} \\ &= \begin{pmatrix} \left(\nabla \boldsymbol{c}_1^\top \boldsymbol{b} \right)^\top \\ \vdots \\ \left(\nabla \boldsymbol{c}_k^\top \boldsymbol{b} \right)^\top \end{pmatrix} \\ &= \begin{pmatrix} \boldsymbol{b}^\top \nabla \boldsymbol{c}_1 \\ \vdots \\ \boldsymbol{b}^\top \nabla \boldsymbol{c}_k \end{pmatrix}. \end{aligned} \tag{14}$$

A.2 Differentiation of Inner and Cross Products

Suppose that $\boldsymbol{u}$ and $\boldsymbol{v}$ are both three-dimensional vectors. Then, equation (6) is applied in differentiation of their dot product:

$$\frac{\partial (\boldsymbol{u} \cdot \boldsymbol{v})}{\partial \boldsymbol{v}} = \frac{\partial (\boldsymbol{u}^\top \boldsymbol{v})}{\partial \boldsymbol{v}} = \boldsymbol{u}^\top,$$

$$\frac{\partial(\boldsymbol{u}\cdot\boldsymbol{v})}{\partial\boldsymbol{u}} \;=\; \frac{\partial(\boldsymbol{v}^\top\boldsymbol{u})}{\partial\boldsymbol{u}} \;=\; \boldsymbol{v}^\top.$$

Before differentiating the cross product $\boldsymbol{u}\times\boldsymbol{v}$, for any vector $\boldsymbol{w}=(w_1,w_2,w_3)^\top$ we denote by $\boldsymbol{w}\times$ the following 3×3 anti-symmetric matrix:

$$[\boldsymbol{w}]_\times = \begin{pmatrix} 0 & -w_3 & w_2 \\ w_3 & 0 & -w_1 \\ -w_2 & w_1 & 0 \end{pmatrix}.$$

Apparently, the product of the matrix $[\boldsymbol{u}]_\times$ with $\boldsymbol{v}$ is the cross product $\boldsymbol{u}\times\boldsymbol{v}$. It then follows that

$$\begin{aligned} \frac{\partial(\boldsymbol{u}\times\boldsymbol{v})}{\partial\boldsymbol{v}} &= [\boldsymbol{u}]_\times, \\ \frac{\partial(\boldsymbol{u}\times\boldsymbol{v})}{\partial\boldsymbol{u}} &= -\frac{\partial(\boldsymbol{v}\times\boldsymbol{u})}{\partial\boldsymbol{u}} \\ &= -[\boldsymbol{v}]_\times. \end{aligned}$$

Now let us look at how to differentiate the scalar $\boldsymbol{x}^\top A\boldsymbol{x}$, where A is a constant $n\times n$ matrix, with respect to $\boldsymbol{x}$. We have

$$\begin{aligned} \nabla\left(\boldsymbol{x}^\top A\boldsymbol{x}\right) &= \frac{\partial}{\partial\boldsymbol{x}}\left(\boldsymbol{x}^\top A\boldsymbol{x}\right) \\ &= \frac{\partial}{\partial\boldsymbol{x}}\left(\boldsymbol{x}^\top(A\boldsymbol{y})\right)\Big|_{\boldsymbol{y}=\boldsymbol{x}} + \frac{\partial}{\partial\boldsymbol{x}}\left((\boldsymbol{y}^\top A)\boldsymbol{x}\right)\Big|_{\boldsymbol{y}=\boldsymbol{x}} \\ &= A\boldsymbol{y}\Big|_{\boldsymbol{y}=\boldsymbol{x}} + \frac{\partial}{\partial\boldsymbol{x}}\left((A^\top\boldsymbol{y})^\top\boldsymbol{x}\right)\Big|_{\boldsymbol{y}=\boldsymbol{x}} \\ &= A\boldsymbol{x} + A^\top\boldsymbol{y}|_{\boldsymbol{y}=\boldsymbol{x}} \\ &= A\boldsymbol{x} + A^\top\boldsymbol{x}. \end{aligned}$$

The above reduces to $2A\boldsymbol{x}$ when A is symmetric.

A.3 Trace of a Product Matrix

The derivative of a scalar function $f(A)$, where $A=(a_{ij})_{m\times n}$ is a matrix, is defined as

$$\frac{\partial f}{\partial A} = \begin{pmatrix} \frac{\partial f}{\partial a_{11}} & \frac{\partial f}{\partial a_{12}} & \cdots & \frac{\partial f}{\partial a_{1n}} \\ \frac{\partial f}{\partial a_{21}} & \frac{\partial f}{\partial a_{22}} & \cdots & \frac{\partial f}{\partial a_{2n}} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial f}{\partial a_{m1}} & \frac{\partial f}{\partial a_{m2}} & \cdots & \frac{\partial f}{\partial a_{mn}} \end{pmatrix}. \tag{15}$$

Let $\boldsymbol{u}=(u_1,u_2,\ldots,u_m)^\top$ and $\boldsymbol{v}=(v_1,v_2,\ldots,v_n)^\top$. In the product $\boldsymbol{u}^T A\boldsymbol{v}$ as a summation, a_{ij} appears in the summand $u_i a_{ij} v_j$ only. This leads to

$$\frac{\partial}{\partial A}(\boldsymbol{u}^\top A\boldsymbol{v}) = \boldsymbol{u}\boldsymbol{v}^\top.$$

The trace of an $n \times n$ square matrix $B = (b_{ij})$ is the sum of its diagonal entries, i.e., $\mathrm{Tr}(B) = \sum_{i=1}^{n} b_{ii}$. Trivially, by (15) we have

$$\frac{\partial}{\partial B}\mathrm{Tr}(B) = I_n.$$

Let $C = (c_{ij})_{m\times n}$ and $X = (x_{ij})_{m\times n}$ be two matrices. The trace of the product matrix $CX^\top$ is

$$\mathrm{Tr}(CX^\top) = \sum_{r=1}^{m}\left(\sum_{s=1}^{n} c_{rs}x_{rs}\right).$$

Since

$$\frac{\partial}{\partial x_{ij}}\mathrm{Tr}(CX^\top) = c_{ij},$$

we obtain

$$\frac{\partial}{\partial X}\mathrm{Tr}(CX^\top) = C.$$

Next, we differentiate the trace of the product matrix $XCX^\top$ as follows:

$$\begin{aligned}
\frac{\partial}{\partial X}\mathrm{Tr}(XCX^\top) &= \frac{\partial}{\partial X}\mathrm{Tr}(XCY^\top)\Big|_{Y=X} + \frac{\partial}{\partial X}\mathrm{Tr}\Big((YC)X^\top\Big)\Big|_{Y=X} \\
&= \frac{\partial}{\partial X}\mathrm{Tr}\Big((YC^\top)X^\top\Big)\Big|_{Y=X} + YC\Big|_{Y=X} \\
&= YC^\top\Big|_{Y=X} + XC \\
&= XC^\top + XC.
\end{aligned}$$

The above reduces to $2XC$ when C is symmetric.

A.4 Matrix in One Variable

Consider an $m\times n$ matrix A in which every element is a function of some variable t. Write the matrix as $A(t) = (a_{ij}(t))_{m\times n}$. Let the overdot denote differentiation with respect to t. Differentiation and integration of the matrix operate element-wise:

$$\begin{aligned}
\dot{A}(t) &= (\dot{a}_{ij}), \\
\int A(t)\,dt &= \left(\int a_{ij}(t)\,dt\right).
\end{aligned}$$

Suppose $A(t)$ is $n \times n$ and non-singular. Then we have $AA^{-1} = I_n$, the $n \times n$ identity matrix. Thus,

$$\begin{aligned}
0 &= \frac{d}{dt}(AA^{-1}) \\
&= \dot{A}A^{-1} + A\frac{d}{dt}(A^{-1}),
\end{aligned}$$

which yields the derivative of the inverse matrix:

$$\frac{d}{dt}(A^{-1}) = -A^{-1}\dot{A}A^{-1}. \tag{16}$$

An interesting case is with the rotation matrix R, which is also orthogonal, i.e., $RR^\top = R^\top R = I_n$. We obtain

$$\begin{aligned} RR^\top = I \quad &\Rightarrow \quad \dot{R}R^\top + R\dot{R}^\top = 0 \\ &\Rightarrow \quad \dot{R}R^\top + \left(\dot{R}R^\top\right)^\top = 0 \\ &\Rightarrow \quad \dot{R}R^\top = -\left(\dot{R}R^\top\right)^\top. \end{aligned}$$

The above implies that the matrix $\dot{R}R^\top$ is anti-symmetric. Therefore, it can be written as

$$\dot{R}R^\top = \begin{pmatrix} 0 & -\omega_z & \omega_y \\ \omega_z & 0 & -\omega_x \\ -\omega_y & \omega_x & 0 \end{pmatrix}.$$

The vector $\boldsymbol{\omega} = (\omega_x, \omega_y, \omega_z)^\top$ is the angular velocity, where the cross product $\boldsymbol{\omega} \times \boldsymbol{v} = \dot{R}R^\top \boldsymbol{v}$ describes the change rate of the vector $\boldsymbol{v}$ (i.e., the velocity of the destination point of the vector) due to the body rotation described by R.

Using the Taylor expansion, we define the exponential function:

$$e^{At} = \sum_{j=0}^{\infty} \frac{(At)^j}{j!},$$

where A is an $n \times n$ matrix. The function's importance comes from that it is the solution to the linear system $\dot{\boldsymbol{x}} = A\boldsymbol{x} + b\boldsymbol{u}$, where $\boldsymbol{u}$ is the control vector. It has the derivative

$$\frac{d}{dt}e^{At} = Ae^{At} = e^{At}A.$$

References

[1] S. Boyd and L. Vandenberghe. *Convex Optimization*. Cambridge University Press, 2004.

[2] M. Erdmann. Lecture notes for *16-811 Mathematical Fundamentals for Robotics*. The Robotics Institute, Carnegie Mellon University, 1998.

[3] D. G. Luenberger. *Linear and Nonlinear Programming*. Addison-Wesley, 2nd edition, 1984.

[4] J. Nocedal and S. J. Wright. *Numerical Optimization*. Springer, 2nd edition, 2006.

[5] W. H. Press, *et al. Numerical Recipes in C++: The Art of Scientific Computing*. Cambridge University Press, 2nd edition, 2002.