Lecture 6

Convex Optimization (II)

6.1 Lecture Objectives

- Understand what are convex functions.
- Understand what are unconstrained convex optimization problems.
- Understand what are convex sets and constrained convex optimization problems.
- Recognize the implications of convex optimization problems in terms of global optimization.
- Recognize convex formulations when being faced with them.

6.2 Unconstrained Convex Optimization Problems

An unconstrained optimization problem:

$$\min_{\mathbf{x}} f(\mathbf{x}) \tag{6.1}$$

is convex if and only if $f(\mathbf{x})$ is a convex function. Importantly, convex problems have the guarantee (stated here informally) that if an iterative algorithm converges to a local optimizer, then this is guaranteed to also be a global optimizer. Further, if the function f is strictly convex, then this local optimizer is guaranteed to be the *unique* global minimizer.

What do we mean by a local optimizer? A value $\hat{\mathbf{x}}$ such that $f(\hat{\mathbf{x}}) \leq f(\mathbf{x})$ for all $\mathbf{x}$ in the vicinity of $\hat{\mathbf{x}}$. We will delve into precise mathematical conditions in the following lectures.

What do we mean by a global optimizer? A value $\hat{\mathbf{x}}$ such that $f(\hat{\mathbf{x}}) \leq f(\mathbf{x})$ for all $\mathbf{x} \in \mathbb{R}^N$.

6.3 Constrained Convex Optimization Problems

A constrained optimization problem

$$\begin{aligned} &\text{minimize } f(\mathbf{x}) \\ &\text{such that } g_k(\mathbf{x}) \leq b_k, \text{ for } k = 1, \ldots, K \\ &\text{such that } h_l(\mathbf{x}) = c_l, \text{ for } l = 1, \ldots, L \end{aligned} \tag{6.2}$$

is convex if $f(\mathbf{x})$ is a convex function, and the constraints (inequality and equality) describe a convex set. For instance, our optimization problem will be convex if $f(\mathbf{x})$ and $g_k(\mathbf{x})$ are all convex functions, and $h_l(\mathbf{x})$ are linear functions. To reiterate this important point, in a constrained convex optimization problem, we minimize a convex function over a constraint set (the set of values $\mathbf{x}$ allowed by the constraints) that is itself a convex set.

What is a convex set? A set such that, for any two points within the set, the whole segment that joins the two points is entirely within the set as well. Please see figure 6.1 for an illustration of this concept.

Question: What could be the problem with trying to (use a descent algorithm to) optimize a function (even a convex function) over a non-convex constraint set?

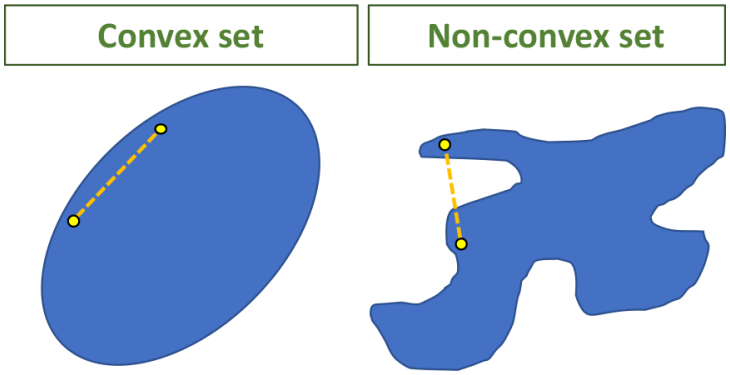

Figure 6.1: Pictorial depiction of a convex vs a non-convex set. In a convex set, the straight line that connects any two points within the set is completely within the set. In a non-convex set, the straight line that connects two points within the set may be partly outside of the set.

6.4 Non-convex Optimization Problems

With convex optimization problems, we have a solid foundation and important guarantees of global optimality if we can find a local optimizer. This does not mean that all convex problems are equally easy, but it is typically critical to understand whether a problem we are facing is convex.

When faced with a non-convex optimization problem, we need to examine it carefully. Some non-convex problems can actually be solved globally by utilizing clever tricks. Also,

in some applications we may not need a globally optimal solution, and a local minimizer may be an acceptable compromise. Further, in yet other applications, when faced with a non-convex problem, we may choose to reformulate our optimization problem as a convex problem that we can actually solve globally. As is the case with many aspects of our research and jobs, there is no silver bullet, and no substitute for thinking carefully about our specific problem.

6.5 Examples from Image Reconstruction and Processing

In this section we will review a few image reconstruction/processing examples we have considered in earlier lectures, and examine their convexity or lack thereof.

6.5.1 Image Reconstruction

In a relatively simple image reconstruction formulation, the cost function calculates the energy (squared ℓ_2 norm) of the data consistency error plus a measure of the "roughness" of the resulting image:

$$\text{minimize } \|\mathbf{Ax} - \mathbf{b}\|_2^2 + \lambda \|\mathbf{Dx}\|_2^2 \tag{6.3}$$

where the matrix $\mathbf{A}$ captures the imaging physics (eg: calculates different Fourier Transform coefficients in the case of MRI, or perform projections along different angles in the case of CT), the vector $\mathbf{b}$ contains the acquired data, the matrix $\mathbf{D}$ calculates finite spatial differences (ie: in order to penalize rougher images over smoother images). The regularization parameter λ would balance the importance of data consistency vs image smoothness.

Question: Is this formulation convex? How about if we replace the second term with $\lambda\|\mathbf{Dx}\|_1$? How about if we replace the second term with $\lambda\|\mathbf{Dx}\|_0$? How about if we replace the unconstrained formulation by the constrained formulation:

$$\text{minimize } \|\mathbf{Ax} - \mathbf{b}\|_2^2 \text{ , subject to } \|\mathbf{Dx}\|_2^2 < c$$

for some $c > 0$?

6.5.2 Image Segmentation

A traditional formulation for segmentation of an input image $\mathbf{y}$ is the Potts model, which can be written as:

$$\text{minimize } \gamma\|\nabla\mathbf{x}\|_0 + \|\mathbf{x} - \mathbf{y}\|_2^2 \tag{6.4}$$

where ∇ performs a gradient operation, and the $\|\cdot\|_0$ notation is the ℓ_0 metric, that simply calculates the number of non-zero entries in a vector (in this case, the number of points with non-zero gradient).

Question: Is this formulation convex?

6.5.3 Image Registration

Registration of input images $\mathbf{y_1}$ and $\mathbf{y_2}$ can be described as a pixel-wise geometric operation on $\mathbf{y_2}$ that makes it closely aligned (co-registered) with $\mathbf{y_1}$.

$$\text{minimize } E(\mathbf{y_1}, r(\mathbf{y_2}, \mathbf{x})) + \lambda R(\mathbf{x}) \tag{6.5}$$

where the function r applies a geometric distortion to $\mathbf{y_2}$, as parameterized by $\mathbf{x}$, and the function E calculates a measure of dissimilarity between $\mathbf{y_1}$ and $r(\mathbf{y_2}, \mathbf{x})$. The 'regularization' term $\lambda R(\mathbf{x})$ penalizes distortions that are deemed unlikely (eg: huge distortions, or distortions where the tissue folds onto itself in non-physical ways). Typical choices for E are based on measures of mutual information using entropy, and seek to penalize images that contain features (eg: edges) at different locations.

Question: Is this formulation convex? (Note that there need not be a general answer given the non-specific nature of this formulation). How about if we replace the unconstrained formulation by the constrained formulation:

$$\text{minimize } E(\mathbf{y_1}, r(\mathbf{y_2}, \mathbf{x})) \text{ , subject to } R(\mathbf{x}) < c$$

for some $c > 0$? Again, note that this may depend on R - can you think of a typical examples of E and R, and answer for this example?

6.5.4 Neural Networks

Deep neural networks are often used for complex regression problems, such as estimating CT images from MRI images. These networks are generally highly complex and understanding the optimization of neural network objective functions is currently a research topic. However, these networks are often composed of very simple layers, which we can analyze. As a very simple example, consider a single layer neural network with a single output. In this case, we may wish to approximate a function $f(z)$ using a basis set of scaled and shifted functions. For example, in a single-hidden-layer, two-neuron network we might represent the network as a function:

$$g(\mathbf{w}, \mathbf{b}; z) = w_{2,1} ReLu(w_{1,1} z + b_1) + w_{2,2} ReLu(w_{1,2} z + b_2) \tag{6.6}$$

where $w_{i,j}$ represents a network weight in layer i, b_i represents a bias, and ReLu is a rectified linear unit, defined as:

$$ReLu(z) = \begin{cases} 0 & z < 0 \\ z, & z \geq 0 \end{cases} \tag{6.7}$$

For a regression problem, we might aim to represent some arbitrary function $h(z)$ using $g(\mathbf{w}, \mathbf{b}; z)$ by minimizing the squared differences between the neural network function and the desired function, over a set of different values of z, i.e. $[z_1, z_2, z_3, \ldots]$:

$$\underset{\mathbf{w},\mathbf{b}}{minimize} \sum_k |h(z_k) - g(\mathbf{w}, \mathbf{b}; z_k)|^2 \tag{6.8}$$

Note that this is done using a discrete number of example 'training' pairs of z_k and $h(z_k)$. In other words, we are trying to find the optimal parameters $\mathbf{w}$ and $\mathbf{b}$ that allow $g(\mathbf{w}, \mathbf{b}; z)$ to approximate some function $h(z)$ as well as possible, as defined by a least-squares formulation over a set of training data $[z_1, z_2, z_3, ...]$ and $[h(z_1), h(z_2), h(z_3), \ldots]$.

Question: Is our cost function in Equation 6.8 ($f(\mathbf{w}, \mathbf{b}) = \sum_k |h(z_k) - g(\mathbf{w}, \mathbf{b}; z_k)|^2$) a convex function? Is this convex with respect to the weights, w? Is this convex with respect to the biases, b? Do any of these conditions depend on the convexity of $h(z)$ (with respect to z)?