Convex Optimization Examples

Jaime Tenorio

April 2023

Convex Optimization

A *convex optimization problem* can be stated as

$$\begin{aligned} \text{minimize} \quad & f_0(x) \\ \text{subject to} \quad & f_i(x) \leq 0, \qquad i = 1, \ldots, m \\ & a_i^T x = b_i, \qquad i = 1, \ldots, p, \end{aligned} \tag{1}$$

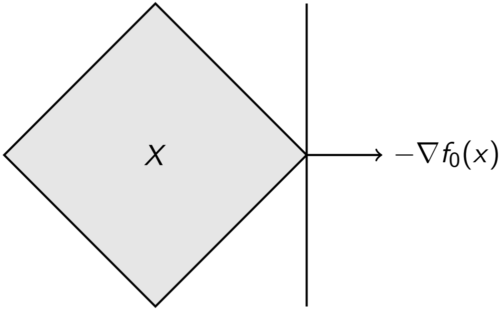
where the function $f_0(x)$ and the inequality constraints $f_i(x)$ are convex, and the equality constraint must be affine.
The feasible set is convex because it is the intersection of of the domain of the constraints, which are convex.

$$D = \bigcap_{i}^{m} \mathbf{dom} f_i \tag{2}$$

Suppose we have a convex optimization problem where the objective function f_0 is differentiable. Let X be the feasible set of the problem. Then $x \in X$ is optimal if and only if:

$$\nabla f_0(x)^T (y - x) \geq 0 \quad \forall y \in X \tag{3}$$

Geometrically, this means that the gradient of the objective function at x points in the direction of the feasible set, in other words, it is a supporting hyperplane to X.



Unconstrained Problems

Suppose we have no constraint inequalities f_i and the objective function f_0 is differentiable. Then $x \in X$ is optimal if and only if:

$$\nabla f_0(x) = 0 \tag{4}$$

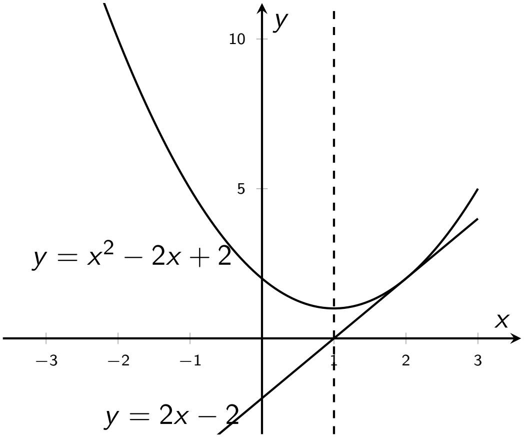
Take as an example the *unconstrained quadratic optimization*:

$$\text{minimize} \quad f_0(x) = \frac{1}{2}x^T P x + q^T x + r \tag{5}$$

where $P \in \mathbf{S}_+^n$. The optimality condition gives:

$$\nabla f_0(x) = Px + q = 0 \tag{6}$$

We can visualize this in two dimensions in the following figure:



Equality Constraints

If we have only equality constraints

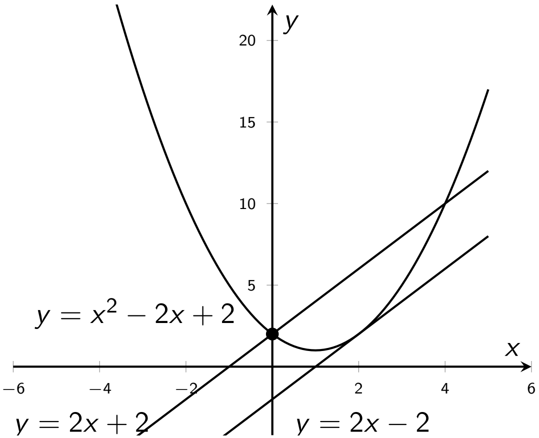
$$\begin{aligned} \text{minimize} \quad & f_0(x) \\ \text{subject to} \quad & Ax = b \end{aligned} \tag{7}$$

the feasible set is affine. The optimality condition for $x \in X$ is

$$\exists \nu \in \mathbb{R}^p \quad s.t. \quad \nabla f_0(x) + A^T \nu = 0 \quad \text{and} \quad Ax = b \tag{8}$$

which is the Lagrange multiplier optimality condition, also known as the Karush-Kuhn-Tucker (KKT) condition.

We can visualize this in two dimensions in the following figure:



Nonnegative Orthant

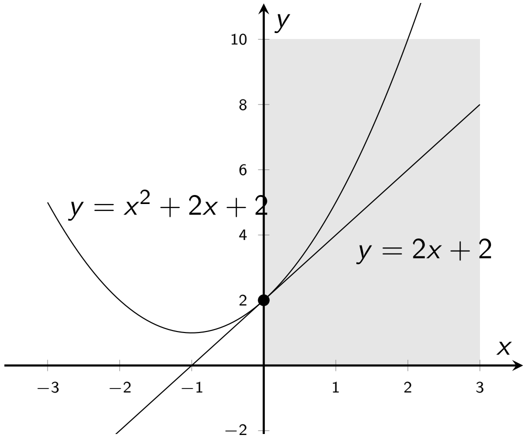
If we consider a problem like

$$\begin{aligned} \text{minimize} \quad & f_0(x) \\ \text{subject to} \quad & x \succeq 0 \end{aligned} \tag{9}$$

where the only constraint is the nonnegativity of x. The feasible set is the nonnegative orthant. The optimality condition for $x \succeq 0, x \in X$ is

$$\begin{cases} \nabla f_0(x)_i \geq 0 & x_i = 0 \\ \nabla f_0(x)_i = 0 & x_i > 0 \end{cases} \tag{10}$$

We can visualize this in two dimensions in the following figure:



Equivalent Problems

We can obtain **equivalent** convex problems using transformations that preserve convexity.

Original	Equivalent
$\begin{array}{ll} \text{minimize} & f_0(x) \\ \text{subject to} & f_i(x) \leq 0 \\ & Ax = b \end{array}$	$\begin{array}{ll} \text{minimize (over } z) & f_0(Fz + x_0) \\ \text{subject to} & f_i(Fz + x_0) \\ \multicolumn{2}{c}{Ax + b \iff x = Fz + x_0} \end{array}$
$\begin{array}{ll} \text{minimize} & f_0(A_0x + b_0) \\ \text{subject to} & f_i(A_ix + b_i) \leq 0 \end{array}$	$\begin{array}{ll} \text{minimize (over } x, y_i) & f_0(y_0) \\ \text{subject to} & f_i(y_i) \leq 0 \\ & y_i = A_ix + b_i \end{array}$
$\begin{array}{ll} \text{minimize} & f_0(x) \\ \text{subject to} & a_i^T x \leq b_i \end{array}$	$\begin{array}{ll} \text{minimize (over } x, s) & f_0(x) \\ \text{subject to} & a_i^T x + s_i = b_i \\ & s_i \geq 0 \end{array}$
$\begin{array}{ll} \text{minimize} & f_0(x) \\ \text{subject to} & f_i(x) \leq 0 \\ & a_i^T x = b_i \end{array}$	$\begin{array}{ll} \text{minimize (over } x, t) & t \\ \text{subject to} & f_0(x) - t \leq 0 \\ & f_i(x) \leq 0 \\ & Ax = b \end{array}$
$\begin{array}{ll} \text{minimize} & f_0(x_1, x_2) \\ \text{subject to} & f_i(x_1) \leq 0 \end{array}$	$\begin{array}{ll} \text{minimize} & \tilde{f}_0(x_1) \\ \text{subject to} & f_i(x_1) \leq 0 \\ \multicolumn{2}{c}{\tilde{f}_0(x_1) = \inf_{x_2} f_0(x_1, x_2)} \end{array}$

In the table we can see equivalent problems for the following transformations:

1. Eliminating equality constraints
2. Introducing equality constraints
3. Introducing slack variables for linear inequality constraints
4. Epigraph form of the standard convex problem
5. Minimizing over some variables

Neural Network Training

Training a neural network can be posed as a convex optimization problem.

$$\text{minimize} \quad L(y; \hat{y}) \tag{11}$$

where $L(y; \hat{y})$ is the loss function, y is the ground truth, and $\hat{y}$ is the prediction. The prediction is given by the neural network, which is a function of the weights W and the input x. If we consider a network of one layer, the prediction is given by

$$\hat{y} = \sigma(Wx) \tag{12}$$

where σ is the activation function. For this problem to be convex we need the loss function to be convex. We can consider the following loss functions:

- **Mean squared error (MSE)**: $L(y; \hat{y}) = \frac{1}{2}(y - \hat{y})^2$
- **Cross-entropy loss**: $L(y; \hat{y}) = -y \log(\hat{y}) - (1 - y) \log(1 - \hat{y})$
- **Hinge loss**: $L(y; \hat{y}) = \max(0, 1 - y\hat{y})$

We can illustrate an example with the *perceptron learning algorithm*. In this example we have a single neuron with two inputs, x_1 and x_2, and one output, y. The neuron has two weights, w_1 and w_2, and one bias, b. The neuron is activated by the hard limit function $\mathrm{hardlim}(x)$, which is defined as

$$\mathrm{hardlim}(x) = \begin{cases} 1 & x \geq 0 \\ 0 & x < 0 \end{cases} \tag{13}$$

The perceptron output is given by

$$y = \mathrm{hardlim}(Wx + b) = \mathrm{hardlim}(w_1 x_1 + w_2 x_2 + b) \tag{14}$$

We can use binary cross entropy loss function to train the perceptron. We then have the following optimization problem

$$\begin{aligned} \text{minimize over } w_1, w_2, b \quad & \sum_{i=1}^{N} \left(-y_i \log(\hat{y}_i) - (1 - y_i) \log(1 - \hat{y}_i)\right) \\ \text{subject to} \quad & \hat{y}_i = \mathrm{hardlim}(w_1 x_{i1} + w_2 x_{i2} + b) \\ & w_1, w_2, b \in \mathbb{R} \end{aligned} \tag{15}$$

If the weights and the bias are initialized to a random value, the prediction of the perceptron will not be correct. The loss function will measure the distance between the prediction and the ground truth, and the gradient of the loss function will point in the direction of the correct prediction. The weights and the bias will be updated in the direction of the correct prediction, and the prediction will be closer to the ground truth. This process will be repeated until the prediction is correct. This is the perceptron learning algorithm.

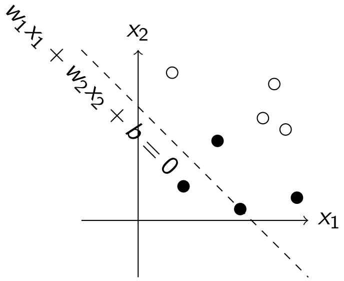


Figure: Bad classification

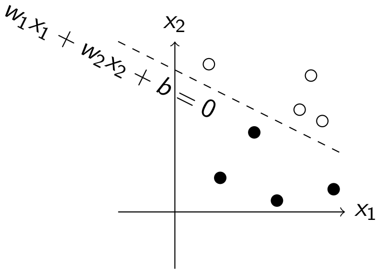


Figure: Good classsification

Usually, however, the optimization of a neural network is not a convex problem. For example, YOLO is a convolutional neural network for object detection. It is trained using the following loss function

$$
\begin{aligned}
L = {} & \lambda_{\text{coord}} \sum_{i=0}^{S^2} \sum_{i=0}^{B} \mathbf{1}_{ij}^{\text{obj}} \left[(x_i - \hat{x}_i)^2 + (y_i - \hat{y}_i)^2 \right] \\
& + \lambda_{\text{coord}} \sum_{i=0}^{S^2} \sum_{i=0}^{B} \mathbf{1}_{ij}^{\text{obj}} \left[(\sqrt{w_i} - \sqrt{\hat{w}_i})^2 + (\sqrt{h_i} - \sqrt{\hat{h}_i})^2 \right] \\
& + \sum_{i=0}^{S^2} \sum_{i=0}^{B} \mathbf{1}_{ij}^{\text{obj}} \left(C_i - \hat{C}_i \right)^2 \\
& + \lambda_{\text{noobj}} \sum_{i=0}^{S^2} \sum_{i=0}^{B} \mathbf{1}_{ij}^{\text{noobj}} \left(C_i - \hat{C}_i \right)^2 \\
& + \sum_{i=0}^{S^2} \mathbf{1}_{i}^{\text{obj}} \sum_{c \in \text{classes}} (p_i(c) - \hat{p}_i(c))^2
\end{aligned}
\tag{16}
$$

where $\hat{x}, \hat{y}, \hat{w}, \hat{h}, \hat{c}$ are the output from the convolutional neural network.

Although this function is a modification of sum squared error, it is not convex. Optimization algorithms such as gradient descent can be used to find a local minimum of this function, but it is not guaranteed that the local minimum is a global minimum.

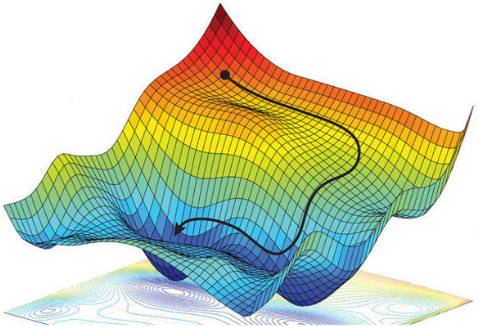


Figure: Optimization of the loss function.