Non-Convex Optimization

Krishna Balasubramanian

May 11, 2020

1 Coping with Non-convexity

Firstly, all the algorithms that we discussed for convex optimization could be run even when the objective function being optimized is non-convex. But the guarantees that we could provide on those algorithms are not as strong as the convex scenario. Essentially, the assumption of *convexity* provides us with a nice structure for optimization purpose as we saw before. In the absence of convexity, things become a bit messy. For example, one **cannot** provide a general result of the form: If we run *an algorithm* for t steps (with t being finite), then we could get ϵ-close to the global solution. To cope with non-convexity, we attack different classes of non-convex problems differently. And we content ourselves with local convergence results as opposed to global convergence results that we did for convex cases.

2 Missing Data and the EM Algorithm

The first class of non-convex problems that we will see involves problems that arise due to missing data. In many cases the observed data contains missing values i.e. $\tilde{\mathbf{x}}_1, \ldots, \tilde{\mathbf{x}}_n \overset{\text{iid}}{\sim} p$ where $\tilde{\mathbf{x}}_i$ can be partitioned to two vectors $\tilde{\mathbf{x}}_i = (\mathbf{x}_i, Z_i)$ where $\mathbf{x}_i$ is observed and Z_i is missing. Assume for simplicity that each $Z_i \in \{1, \ldots, K\}$. In this case, the (log)-likelihood

$$\ell(\boldsymbol{\theta}) = \sum_{i=1}^{n} \log p(\tilde{\mathbf{x}}_i)$$

cannot be computed or maximized. A common alternative is to maximize the likelihood of the observed data

$$\begin{aligned}\hat{\boldsymbol{\theta}} &= \arg\max_{\boldsymbol{\theta}} \; p(\text{observed data}) \\ &= \arg\max_{\theta} \sum_{i=1}^{n} \log p_{\boldsymbol{\theta}}(\mathbf{x}_i) \\ &= \arg\max_{\boldsymbol{\theta}} \sum_{i=1}^{n} \log \sum_{Z_i} p_{\boldsymbol{\theta}}(\mathbf{x}_i, Z_i) \end{aligned} \tag{1}$$

where the sum over Z_i (denoted as $\sum_{Z_i}$ is potentially multidimensional (over all possible values of the missing entries). In many cases the summation over Z_i above is intractable. This is especially true when the amount of missing data grows since the number of terms in the sum grows exponentially with the dimensionality of Z_i. Furthermore, because of the form of the Equation 1, invariably the maximization problem will be non-convex.

2.1 Expectation Maximization Algorithm

The expectation maximization (EM) algorithm is an iterative algorithm to deal with such non-convex problems that arise due to missing data. The EM algorithm maximizes in each round t, a lower bound on the likelihood $\ell(\boldsymbol{\theta})$ above, constructed to be tight at the current guess $\boldsymbol{\theta}^{(t)}$. Repeatedly constructing such bounds and maximizing them converges to a local maximum, often at a much lower computational cost than gradient descent for (1). The EM algorithm is based on maximizing the following bound on the likelihood of the observed data

$$\begin{aligned}\ell(\boldsymbol{\theta}) &= \sum_{i=1}^{n} \log \sum_{Z_i} p_{\boldsymbol{\theta}}(\mathbf{x}_i, Z_i) \\ &= \sum_{i=1}^{n} \log \sum_{Z_i} q_i(Z_i) \frac{p_{\boldsymbol{\theta}}(\mathbf{x}_i, Z_i)}{q_i(Z_i)} \\ &= \sum_{i=1}^{n} \log \mathsf{E}_{q_i} \left(\frac{p_{\boldsymbol{\theta}}(\mathbf{x}_i, Z_i)}{q_i(Z_i)} \right) \\ &\geq \sum_{i=1}^{n} \mathsf{E} \left(\log \frac{p_{\boldsymbol{\theta}}(\mathbf{x}_i, Z_i)}{q_i(Z_i)} \right) \\ &= \sum_{i=1}^{n} \sum_{Z_i} q_i(Z_i) \log \frac{p_{\boldsymbol{\theta}}(\mathbf{x}_i, Z_i)}{q_i(Z_i)}. \\ &= \sum_{i=1}^{n} \sum_{Z_i} \left(q_i(Z_i) \log p_{\boldsymbol{\theta}}(\mathbf{x}_i, Z_i) - q_i(Z_i) \log q_i(Z_i) \right)\end{aligned}$$

Here, q_i are nonzero distributions. Also the inequality is based on Jensen's inequality (stated below) applied to the convex $f(x) = -\log x$

Proposition 1 (Jensen's Inequality)**.** *For a RV X and a convex function f we have $\mathsf{E} f(X) \geq f(\mathsf{E} X)$. Moreover, if f is strongly convex, equality holds iff X is degenerate i.e. i.e. $P(X = \mathsf{E} X) = 1$.*

Note that the term $q_i(Z_i) \log q_i(Z_i)$, does not depend on $\boldsymbol{\theta}$ and therefore can be removed in maximization over $\boldsymbol{\theta}$. Above, we actually have a parameterized family of bounds - one bound for each selection of the distributions $q_1, \ldots, q_n$. Recall that Jensen's inequality is equality for deterministic RV and therefore the selection

$$\begin{aligned} & q_i(Z_i) \propto p_{\boldsymbol{\theta}'}(\mathbf{x}_i, Z_i) \\ \Rightarrow \quad & q_i(Z_i) = \frac{p_{\boldsymbol{\theta}'}(\mathbf{x}_i, Z_i)}{\sum_{Z_i} p_{\boldsymbol{\theta}'}(\mathbf{x}_i, Z_i)} \\ & \qquad = p_{\boldsymbol{\theta}'}(Z_i | \mathbf{x}_i) \end{aligned}$$

would yield a bound with equality at $\boldsymbol{\theta}'$.

Hence, the EM algorithm iterates between the following steps:

Step 1 (E step): Compute $p_{\boldsymbol{\theta}^{(t)}}(Z_i|\mathbf{x}_i)$ and the bound on the observed likelihood

$$Q(\boldsymbol{\theta}, \boldsymbol{\theta}^{(t)}) \stackrel{\text{def}}{=} \sum_{i=1}^{n} \sum_{Z_i} p_{\boldsymbol{\theta}^{(t)}}(Z_i|\mathbf{x}_i) \log p_{\boldsymbol{\theta}}(\mathbf{x}_i, Z_i)$$

Step 2 (M step): Maximize the bound to update new value $\theta^{(t+1)}$

$$\boldsymbol{\theta}^{(t+1)} = \arg\max_{\boldsymbol{\theta}} Q(\boldsymbol{\theta}, \boldsymbol{\theta}^{(t)})$$

Why does the algorithm above works/converge (locally)?
The fact that each iteration in the EM algorithm increases the likelihood may be seen by

$$\begin{aligned} \ell(\boldsymbol{\theta}^{(t+1)}) & \geq \sum_{i=1}^{n} \sum_{Z_i} p_{\boldsymbol{\theta}^{(t)}}(Z_i|\mathbf{x}_i) \log p_{\boldsymbol{\theta}^{(t+1)}}(\mathbf{x}_i, Z_i) \\ & \geq \sum_{i=1}^{n} \sum_{Z_i} p_{\boldsymbol{\theta}^{(t)}}(Z_i|\mathbf{x}_i) \log p_{\boldsymbol{\theta}^{(t)}}(\mathbf{x}_i, Z_i) \\ & = \ell(\boldsymbol{\theta}^{(t)}) \end{aligned}$$

where the first inequality follows from Jensen's inequality (for the specified $q_i = p_{\boldsymbol{\theta}^{(t)}}(Z_i|\mathbf{x}_i)$), the second from the maximization step in EM, and the equality follows from the tightness of the bound at $\boldsymbol{\theta}^{(t)}$. A simple illustration of the EM algorithm in words is available in the PDF below

`http://ai.stanford.edu/~chuongdo/papers/em_tutorial.pdf`

2.2 Example 1: EM algorithm for Mixture of Binomials

Suppose you toss two coins (coin 0 and coin 1) with unknown probabilities of heads, denoted by p and q respectively, n times. The observed data is $x_1, \ldots, x_n$ and each $x_i \in \{0, 1\}$. Coin 0 is chosen with probability π (and so, coin 1 is chosen with probability $(1-\pi)$). For each toss, Z_i represents which of the two coins are selected. So, $Z_i \in \{0, 1\}$. The parameter vector is given by $\boldsymbol{\theta} = (\pi, p, q)$. The data is given by $\tilde{\mathbf{x}}_i = (x_i, Z_i)$, for $i = 1, \ldots, n$. Given the full data, $\tilde{\mathbf{x}}_1, \ldots, \tilde{\mathbf{x}}_n$, its trivial to estimate $\boldsymbol{\theta}$. The challenge here is, as before, $Z_1, \ldots, Z_n$ is **not observed**. Hence, we use the EM algorithm to estimate $\boldsymbol{\theta}$ from just $x_1, \ldots, x_n$.

We first compute the **E-step** as follows:

$$\begin{aligned} Q(\boldsymbol{\theta}, \boldsymbol{\theta}^{(t)}) &= Q(\boldsymbol{\theta}, \boldsymbol{\theta}^{(t)}) \\ &= \sum_{i=1}^{n} \Big(\mathbf{F}_{i0}^{(t)} \left(\log \pi + x_i \log p + (1 - x_i) \log(1 - p) \right) \\ &\qquad + \mathbf{F}_{i1}^{(t)} \left(\log(1 - \pi) + x_i \log q + (1 - x_i) \log(1 - q) \right) \Big) \end{aligned}$$

where

$$\begin{aligned} \mathbf{F}_{i0}^{(t)} &= p_{\boldsymbol{\theta}^{(t)}}(Z_i = 0 | x_i) \\ &= \frac{p_{\boldsymbol{\theta}^{(t)}}(x_i | Z_i = 0) p_{\boldsymbol{\theta}^{(t)}}(Z_i = 0)}{p_{\boldsymbol{\theta}^{(t)}}(x_i)} \\ &\qquad \frac{\pi^{(t)} [p^{(t)}]^{x_i} [1 - p^{(t)}]^{1 - x_i}}{\pi^{(t)} [p^{(t)}]^{x_i} [1 - p^{(t)}]^{1 - x_i} + (1 - \pi^{(t)}) [q^{(t)}]^{x_i} [1 - q^{(t)}]^{1 - x_i}} \end{aligned}$$

and $\mathbf{F}_{i1}^{(t)} = 1 - \mathbf{F}_{i0}^{(t)}$. Now, we proceed to compute the **M-step** by taking the derivative of $Q(\boldsymbol{\theta}, \boldsymbol{\theta}^{(t)})$ with respect to π, p and q. Doing so, we get

$$\begin{aligned} \pi^{(t+1)} &= \frac{1}{n} \sum_{i=1}^{n} \mathbf{F}_{i0}^{(t)} \\ p^{(t+1)} &= \frac{\sum_{i=1}^{n} \mathbf{F}_{i0}^{(t)} x_i}{\sum_{i=1}^{n} \mathbf{F}_{i0}^{(t)}} \\ q^{(t+1)} &= \frac{\sum_{i=1}^{n} \mathbf{F}_{i1}^{(t)} x_i}{\sum_{i=1}^{n} \mathbf{F}_{i1}^{(t)}} \end{aligned}$$

2.3 Example 2: EM algorithm for Mixture of Gaussians

The Gaussian mixture model assumes the following generative model for our data

$$\mathbf{x} \sim p_{\boldsymbol{\theta}}(\mathbf{x}) = \sum_{j=1}^{K} p_{\boldsymbol{\theta}}(Z=j) p_{\boldsymbol{\theta}}(\mathbf{x}|Z=j) = \sum_{j=1}^{K} \pi_j N(\mathbf{x}\,; \boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j) \tag{2}$$

where Z is a *hidden/latent* variable representing the Gaussian generating X and $\pi_j = p(Z = j)$ and we use the notation $N(\mathbf{y}\,; \boldsymbol{\mu}, \boldsymbol{\Sigma})$ to represent the fact that the random variable $\mathbf{y}$ is normally distributed with mean $\boldsymbol{\mu}$ and covariance $\boldsymbol{\Sigma}$. Here, the unknown parameter $\boldsymbol{\theta}$ contains $\{\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k, \pi_k\}_{k=1}^K$, but in some special case may contain only $\boldsymbol{\mu}, \boldsymbol{\Sigma}$ assuming π is known, or only $\boldsymbol{\mu}$ assuming $\boldsymbol{\Sigma}, \pi$ are known. Our task is the estimate the parameter $\boldsymbol{\theta}$ given n samples $\mathbf{x}_1, \ldots, \mathbf{x}_n$ from the above model. Once the parameter $\boldsymbol{\theta}$ is estimated by maximizing the likelihood of the observed data we can cluster by assigning each $\mathbf{x}^{(i)}$ to the Gaussian most likely to have generated it.

The likelihood is

$$\ell(\boldsymbol{\theta}) = \sum_{i=1}^{n} \log \sum_{j=1}^{K} \pi_j N(\mathbf{x}_i\,; \boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j)$$

and the corresponding EM is

$$\begin{aligned}
\textbf{E Step:} \quad Q(\boldsymbol{\theta}, \boldsymbol{\theta}^{(t)}) &= Q(\boldsymbol{\theta}, \boldsymbol{\theta}^{(t)}) \\
&= Q((\pi, \boldsymbol{\mu}, \boldsymbol{\Sigma}), (\pi^{(t)}, \boldsymbol{\mu}^{(t)}, \boldsymbol{\Sigma}^{(t)})) \\
&= \sum_{i=1}^{n} \sum_{j=1}^{k} \mathbf{F}_{ij}^{(t)} \log \pi_j N(\mathbf{x}_i\,; \boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j)
\end{aligned}$$

$$\mathbf{F}_{ij}^{(t)} = p_{\boldsymbol{\theta}^{(t)}}(Z_i = j|\mathbf{x}_i) = \frac{N(\mathbf{x}_i\,; \boldsymbol{\mu}_j^{(t)}, \boldsymbol{\Sigma}_j^{(t)}) \pi_j^{(t)}}{\sum_{j'=1}^{K} N(\mathbf{x}_i\,; \boldsymbol{\mu}_{j'}^{(t)}, \boldsymbol{\Sigma}_{j'}^{(t)}) \pi_{j'}^{(t)}}$$

$$\textbf{M Step:} \qquad \boldsymbol{\theta}^{(t+1)} = (\pi^{(t+1)}, \boldsymbol{\mu}^{(t+1)}, \boldsymbol{\Sigma}^{(t+1)}) = \arg\max_{\pi, \boldsymbol{\mu}, \boldsymbol{\Sigma}} \quad Q((\pi, \boldsymbol{\mu}, \boldsymbol{\Sigma}), (\pi^{(t)}, \boldsymbol{\mu}^{(t)}, \boldsymbol{\Sigma}^{(t)})).$$

It is straightforward to show that the above maximization has the following closed form. Maximizing for π is similar to deriving the multinomial MLE and maximizing for μ, Σ is

similar to the Gaussian MLE.

$$\pi^{(t+1)} = \sum_{i=1}^{n} \mathbf{F}_{ij}^{(t)} / \sum_{i=1}^{n} \sum_{j'=1}^{K} \mathbf{F}_{ij'} = \sum_{i=1}^{n} \mathbf{F}_{ij}^{(t)} / n$$
$$\boldsymbol{\mu}_j^{(t+1)} = \sum_{i=1}^{n} \mathbf{F}_{ij}^{(t)} \mathbf{x}_i / \sum_{i=1}^{n} \mathbf{F}_{ij}^{(t)}$$
$$\boldsymbol{\Sigma}_j^{(t+1)} = \sum_{i=1}^{n} \mathbf{F}_{ij}^{(t)} (\mathbf{x}_i - \boldsymbol{\mu}_j^{(t+1)})(\mathbf{x}_i - \boldsymbol{\mu}_j^{(t+1)})^\top / \sum_{i=1}^{n} \mathbf{F}_{ij}^{(t)}.$$

Note: Here, for computing $\boldsymbol{\Sigma}_j^{(t+1)}$, if we follow the EM algorithm derivation exactly, we need to use $\boldsymbol{\mu}_j^{(t)}$. But in practice, using $\boldsymbol{\mu}_j^{(t+1)}$ is beneficial.

The above parameters have a natural meaning. Note that they are just the weighted means and the covariance matrices, with the weights being how likely the sample i is to belong to the cluster j.

2.4 k-Means Clustering Algorithm

Recall that in clustering the task is to partition a dataset $X^{(1)}, \ldots, X^{(n)} \in \mathbb{R}^d$ into K disjoint sets so that each set has a coherent set of points. Note that this is an unsupervised task i.e., labels are not available during the training phase. In the previous case, we assumed each cluster is is distributed normally so that the overall data is described by a mixture model. K-means algorithm is a clustering technique that does not explicitly such a probabilistic model assumption. Hence it is a model-free clustering technique.

The method works as follows: start by randomly initializing the cluster centroids $\mu_k^{(0)} \in \mathbb{R}^d, k = 1 \ldots, K$, and follow by iterating over the following two steps till convergence:

1. assign each $X^{(i)}$ to a cluster corresponding to the nearest centroid among $\mu_1^{(t)}, \ldots, \mu_K^{(t)}$,
2. update the cluster centroids based on the cluster membership obtained in (i) i.e.,

$$\mu_k^{(t+1)} = \text{average}(\{X^{(i)} : \|X^{(i)} - \mu_k^{(t)}\| = \min_{k'=1,\ldots,K} \|X^{(i)} - \mu_{k'}^{(t)}\|\}).$$

The first step has the following interpretation. Recall from EM algorithm for mixture of Gaussians, we have (ignoring the iterations count t):

$$F_{ij} = p_\theta(Z^{(i)} = j | X^{(i)}) = \frac{N(X^{(i)}; \mu_j, \Sigma_j) \cdot \pi_j}{\sum_{j'=1}^{K} N(X^{(i)}; \mu_{j'}, \Sigma_{j'}) \cdot \pi_{j'}}$$

Assume that $\Sigma_j = \sigma^2 \mathbf{I}_{d\times d}$ where $\mathbf{I}_{d\times d}$ is the identity matrix. Now, if we let σ^2 go to zero, one can show that we have

$$F_{ij} = \begin{cases} 1 & \text{if } \|X^{(i)} - \mu_j\|_2^2 \leq \|X^{(i)} - \mu_\ell\|_2^2 \text{ for all } \ell \neq j \\ 0 & \text{otherwise} \end{cases}$$

So, even though K-means is a model-free algorithm, the way it is introduced, one way to interpret it through the above model based interpretation.

3 Method of Moment based Initializations

Yet another technique for provable non-convex optimization is that of using special initialization along with gradient descent (or other optimization algorithms that we discussed before). This technique again, works for some class of optimization problems that are common in statistical problems. We will discuss two examples now.

3.1 Mixture of Gaussians

Consider the mixture of Gaussians model in Equation 2, with $K = 2$ for simplicity and $\mathbf{\Sigma}_1 = \mathbf{\Sigma}_2 = \boldsymbol{I}_d$, where $\boldsymbol{I}_d \in \mathbb{R}^{d\times d}$ is the identity matrix and π_1 and π_2 are known. Furthermore, assume that $\boldsymbol{\mu}_1$ and $\boldsymbol{\mu}_2$ linearly independent. Note that the random vector $\mathbf{x}$ could be written as follows:

$$\mathbf{x} = \boldsymbol{\mu}_Z + \mathbf{z}$$

where $\mathbf{z} \sim N(0, \boldsymbol{I}_d)$ and $Z \in \{1, 2\}$ with $P(Z = 1) = \pi_1$ and $P(Z = 2) = \pi_2$. Now, let us try computing the mean and covariance matrix of $\mathbf{x}$. The expected value of $\mathbf{x}$ is given by

$$\bar{\boldsymbol{\mu}} = \mathsf{E}[\mathbf{x}] = \mathsf{E}[\boldsymbol{\mu}_Z + \mathbf{z}] = \mathsf{E}[\boldsymbol{\mu}_Z] = \mathsf{E}[\boldsymbol{\mu}_1 \cdot \mathbf{1}_{\{Z=1\}}] + \mathsf{E}[\boldsymbol{\mu}_2 \cdot \mathbf{1}_{\{Z=2\}}] = \pi_1\boldsymbol{\mu}_1 + \pi_2\boldsymbol{\mu}_2$$

So with just the first moment, we cannot estimate $\boldsymbol{\mu}_1$ and $\boldsymbol{\mu}_2$. Hence, let us proceed to compute the second moment.

$$\begin{aligned} \mathsf{E}[\mathbf{x}\mathbf{x}^\top] &= \mathsf{E}[(\boldsymbol{\mu}_Z + \mathbf{z})(\boldsymbol{\mu}_Z + \mathbf{z})^\top] \\ &= \mathsf{E}[\boldsymbol{\mu}_Z\boldsymbol{\mu}_Z^\top] + \mathsf{E}[\mathbf{z}\mathbf{z}^\top] \\ &= \mathsf{E}[\boldsymbol{\mu}_Z\boldsymbol{\mu}_Z^\top] + \boldsymbol{I}_d \\ &= \pi_1(\boldsymbol{\mu}_1\boldsymbol{\mu}_1^\top) + \pi_2(\boldsymbol{\mu}_2\boldsymbol{\mu}_2^\top) + \boldsymbol{I}_d \end{aligned}$$

From the above expression, we see that if we define a matrix $\mathbf{M}_2$ as

$$\mathbf{M}_2 = \mathsf{E}[\mathbf{x}\mathbf{x}^\top] - \boldsymbol{I}_d,$$

then, we can find the subspace spanned by $\boldsymbol{\mu}_1$ and $\boldsymbol{\mu}_2$ by finding the eigenvectors of the matrix $\mathbf{M}_2$. In the special case that $\boldsymbol{\mu}_1$ and $\boldsymbol{\mu}_2$ themselves are the basis for that subspace, we can recover them means themselves.

While the above discussion was in the population setting, suppose we observe n i.i.d. samples $\mathbf{x}_1, \ldots, \mathbf{x}_n \sim p_{\boldsymbol{\theta}}(\mathbf{x})$. Then, we can estimate the matrix $\mathbf{M}_2$ as

$$\hat{\mathbf{M}}_2 = \frac{1}{n} \sum_{i=1}^{n} \mathbf{x}_i \mathbf{x}_i^\top - \boldsymbol{I}_d$$

Just to be clear, we did not use the missing data Z_i anywhere in our procedure above similar to the EM-algorithm.

One can ask if we can recover the vectors $\boldsymbol{\mu}_1$ and $\boldsymbol{\mu}_2$ without the assumption that the vectors are the basis for the subspace. It turns out that one can look at higher-order moments of the data and it is possible to recover the vectors themselves. For the details, if you are curious, you are referred to [HK13]. Such an algorithm is called as **method of moments** and provides an alternative to likelihood based approaches for parameter estimation.

The bottom line of this approach is that, since eigenvalue decompositions are provably computable (as opposed to the solutions of the EM-algorithm), the above approach provides a provable way of estimating the means of the mixture of Gaussian model (under the required assumptions). Typically, in practice neither EM algorithm nor the above algorithm works well individually. But when the EM algorithm is initialized with the output of method of moment estimator, it supposedly works well. So, we have the two step approach for **provable non-convex** optimization as follows:

1. Use the method of moments approach to an rough estimate of $\boldsymbol{\mu}_1$ and $\boldsymbol{\mu}_2$.
2. Use that estimate as an initializer for EM algorithm.

3.2 Index Models

3.2.1 Some Intuition

Before presenting the model, consider the following toy example for some $a > 0$.

$$\theta = \arg\min_{\theta} \ (\theta^2 - a)^2$$

First, note that the function $f(\theta) = (\theta^2 - a)^2$ is not a convex function of θ. Setting the derivative to zero, we see that

$$\nabla f(\theta) = 2(\theta^2 - a) \cdot 2\theta = 0$$

from which we see that either $\theta^2 - a = 0$ or $\theta = 0$. But we also see that the minimizers are $\theta = \pm\sqrt{a}$. So if we run the gradient descent algorithm for this function, as long as we don't initialize the algorithm at 0, the algorithm will converge to $\pm\sqrt{a}$. In terms of the function values, both solutions are global solutions.

3.2.2 Single Index Models

We now consider a class of single-index models. Let $\mathbf{x} \sim N(0, \boldsymbol{I}_d)$ and let

$$y = g(\mathbf{x}^\top \boldsymbol{\theta}^*) + \epsilon \tag{3}$$

where $\epsilon \sim N(0,1)$ and independent of $\mathbf{x}$. Here, $g(t) : \mathbb{R}^d \to \mathbb{R}$ is called as the link function. When $g(t) = t$, we recover the standard linear regression. When $g(t) = t^2$, the model has a similar behaviour as the toy example considered in Section 3.2.1. Specifically, given n i.i.d. samples $\{(\mathbf{x}_i, y_i)\}_{i=1}^n$ from the model in Equation 3, the least-squares estimator for $\boldsymbol{\theta}$ is given by

$$\arg\min_{\boldsymbol{\theta} \in \mathbb{R}^d} \frac{1}{n} \sum_{i=1}^{n} (y_i - (\mathbf{x}_i^\top \boldsymbol{\theta})^2)^2 \tag{4}$$

This problem has a similar structure to the toy problem in Section 3.2.1 for the purpose of optimization: As long as we don't initialize in some bad regions of $\boldsymbol{\theta}$, we can converge to the global solution. The point to emphasize here is the problem is a non-convex optimization problems for which we are converging to a global solution!

Such an initialization turns out to be given by a method of moments approach. Specificaly, consider the matrix

$$\mathbf{M} = \mathsf{E}\left[y(\mathbf{x}\mathbf{x}^\top - \boldsymbol{I}_d)\right]$$

To see why that is the case, we recollect the following fact about Gaussian random vectors.

Fact 3.0.1 (Stein's Identity)**.** For random vector $\mathbf{u} \in \mathbb{R}^d$ that is standard Gaussian, we have

$$\mathsf{E}_{\mathbf{u}}[(\mathbf{u}\mathbf{u}^\top - \mathbf{I})\ h(\mathbf{u})] = \mathsf{E}\left[\nabla^2_{\mathbf{u}}\ h(\mathbf{u})\right]$$

for any twice differentiable function $h : \mathbb{R}^d \to \mathbb{R}$.

One can show that, $\mathbf{M} = \lambda(\boldsymbol{\theta}^* \boldsymbol{\theta}^{*\top})$ by using Fact 3.0.1 as follows. Indeed substituting the model from Equation 3 (with $g(t) = t^2$) in the definition of $\mathbf{M}$, we have

$$\begin{aligned}
\mathbf{M} &= \mathsf{E}\left[y(\mathbf{x}\mathbf{x}^\top - \boldsymbol{I}_d)\right] \\
&= \mathsf{E}\left[((\mathbf{x}^\top \boldsymbol{\theta}^*)^2 + \epsilon)(\mathbf{x}\mathbf{x}^\top - \boldsymbol{I}_d)\right] \\
&= \mathsf{E}\left[(\mathbf{x}^\top \boldsymbol{\theta}^*)^2 (\mathbf{x}\mathbf{x}^\top - \boldsymbol{I}_d)\right] + \mathsf{E}\left[\epsilon(\mathbf{x}\mathbf{x}^\top - \boldsymbol{I}_d)\right] \\
&= \mathsf{E}\left[(\mathbf{x}^\top \boldsymbol{\theta}^*)^2 (\mathbf{x}\mathbf{x}^\top - \boldsymbol{I}_d)\right] \\
&= \lambda(\boldsymbol{\theta}^* \boldsymbol{\theta}^{*\top})
\end{aligned}$$

where the last line follows from Fact 3.0.1. So, we have the two step approach for **provable non-convex** optimization as follows:

1. Use the method of moments approach to an rough estimate of $\boldsymbol{\theta}^*$.
2. Use that estimate as an initializer for the gradient descent algorithm on the objective function given in Equation 4.

A MLE for Multivariate Gaussian

In this appendix, for the sake of completion, we give a derivation for computing the MLE estimates of a multivariate Gaussian model.

Given n i.i.d. sample from a Gaussian distribution, i.e., $\{\mathbf{x}_i\}_{i=1}^n \in \mathbb{R}^d \sim N(\boldsymbol{\mu}, \boldsymbol{\Sigma})$, the likelihood function is given by

$$\ell(\boldsymbol{\mu}, \boldsymbol{\Sigma}) = \frac{1}{(2\pi)^{\frac{nd}{2}} (\det \boldsymbol{\Sigma})^{n/2}} \exp^{-\frac{1}{2}\sum_{i=1}^n (\mathbf{x}_i - \mu)^\top \boldsymbol{\Sigma}^{-1} (\mathbf{x}_i - \mu)},$$

and the log-likelihood function is given by

$$\begin{aligned} \ell(\boldsymbol{\mu}, \boldsymbol{\Sigma}) &= \log \ell(\boldsymbol{\mu}, \boldsymbol{\Sigma}) \\ &= \frac{n}{2} \log \det \boldsymbol{\Sigma}^{-1} - \frac{1}{2} \sum_{i=1}^n \textsc{trace}\left((\mathbf{x}_i - \mu)(\mathbf{x}_i - \mu)^\top \boldsymbol{\Sigma}^{-1} \right) + C, \end{aligned}$$

where we have used the fact that for a vector a and a matrix A, we have $a^\top A a = \textsc{trace}(a^\top A a) = \textsc{trace}(a a^\top A) = \textsc{trace}(A a a^\top)$ and the constant C does not involve $\boldsymbol{\mu}$ or $\boldsymbol{\Sigma}$. Now we need to take derivative with respect to the vector $\boldsymbol{\mu}$ and the matrix $\boldsymbol{\Sigma}^{-1}$. Imagine them to be just a single variable and you are taking partial derivative with respect to $\boldsymbol{\mu}$ and $\boldsymbol{\Sigma}^{-1}$. Now, we need the following facts (without proofs):

1. $\frac{\partial b^\top a}{\partial a} = b$ for two vectors a and b.
2. $\frac{\partial a^\top A a}{\partial a} = (A + A^\top) a$ for a vector a and matrix A.
3. $\frac{\partial \textsc{trace}(BA)}{\partial A} = B^\top$ for two matrices A and B.
4. $\frac{\partial \det A}{\partial A} = (A^{-1})^\top$ for a matrix A.

Take the partial derivative of $\ell(\boldsymbol{\mu}, \boldsymbol{\Sigma})$ with respect to $\boldsymbol{\mu}$ and setting it to zero, we have

$$\frac{\partial\, \ell(\boldsymbol{\mu}, \boldsymbol{\Sigma})}{\partial \boldsymbol{\mu}} = -\frac{1}{2} \sum_{i=1}^n -2\boldsymbol{\Sigma}^{-1}(\mathbf{x}_i - \boldsymbol{\mu}) = 0.$$

Hence we have $\sum_{i=1}^n (\mathbf{x}_i - \boldsymbol{\mu}) = 0$ from which we have

$$\hat{\boldsymbol{\mu}} = \frac{1}{n} \sum_{i=1}^n \mathbf{x}_i.$$

Thus the sample mean is also the MLE for multivariate Gaussian distribution. Now we take the derivative with respect to $\boldsymbol{\Sigma}^{-1}$. In this case, we have

$$\frac{\partial\, \ell(\boldsymbol{\mu}, \boldsymbol{\Sigma})}{\partial \boldsymbol{\Sigma}^{-1}} = \frac{n\boldsymbol{\Sigma}}{2} - \frac{1}{2} \sum_{i=1}^n \left((\mathbf{x}_i - \boldsymbol{\mu})(\mathbf{x}_i - \boldsymbol{\mu})^\top \right),$$

from which we have

$$\hat{\mathbf{\Sigma}} = \frac{1}{n}\sum_{i=1}^{n}(\mathbf{x}_i - \mu)(\mathbf{x}_i - \mu)^\top.$$

Thus the sample variance (ignoring the n and $n-1$ business) is also the MLE in multivariate Gaussian distribution.

References

[HK13] Daniel Hsu and Sham M Kakade, *Learning mixtures of spherical gaussians: moment methods and spectral decompositions*, Proceedings of the 4th conference on Innovations in Theoretical Computer Science, ACM, 2013, pp. 11–20.