EM 算法和 MM 算法

王璐

EM 算法是一种常用的极大似然估计算法，本章我们介绍如何使用 EM 算法估计混合模型 (mixture models) 或含有隐变量 (latent variables) 的模型，以及更一般的 MM 算法。

1 Gaussian Mixture Model (GMM)

假设数据由 n 个独立同分布的样本组成 $\{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n\}$, 每个样本来自以下模型：

$$\begin{aligned} \boldsymbol{x}_i \mid z_i = j &\sim N(\boldsymbol{\mu}_j, \Sigma_j) \\ z_i &\sim \mathrm{Mult}(1, \phi_1, \ldots, \phi_K) \end{aligned} \tag{1}$$

其中 z_i 是样本 $\boldsymbol{x}_i$ 的隐标签，$z_i \in \{1, 2, \ldots, K\}$, $P(z_i = j) = \phi_j$, $j = 1, \ldots, K$, $\sum_{j=1}^K \phi_j = 1$, 但 z_i 观测不到。在模型(1)中，每个样本 $\boldsymbol{x}_i$ 相当于从 K 个正态分布中随机选一个分布抽样得到，每个分布被选取的概率为 ϕ_j, $j = 1, \ldots, K$, 因此模型(1)被称为 **Gaussian mixture model** (GMM).

在模型(1)中，我们需要估计的参数是 $\boldsymbol{\theta} = \{(\boldsymbol{\mu}_j, \Sigma_j, \phi_j) : j = 1, \ldots, K\}$. 数据的对数似然函数可写为

$$\begin{aligned} l(\boldsymbol{\theta}) &= \sum_{i=1}^n \log p(\boldsymbol{x}_i \mid \boldsymbol{\theta}) \\ &= \sum_{i=1}^n \log \left(\textstyle\sum_{j=1}^K p(\boldsymbol{x}_i, z_i = j \mid \boldsymbol{\theta}) \right) \\ &= \sum_{i=1}^n \log \left(\textstyle\sum_{j=1}^K p(\boldsymbol{x}_i \mid z_i = j, \boldsymbol{\theta}) p(z_i = j \mid \boldsymbol{\theta}) \right) \\ &= \sum_{i=1}^n \log \left(\textstyle\sum_{j=1}^K \phi_j p(\boldsymbol{x}_i \mid \boldsymbol{\mu}_j, \Sigma_j) \right) \end{aligned} \tag{2}$$

其中 $p(\boldsymbol{x}_i \mid \boldsymbol{\mu}_j, \Sigma_j)$ 是正态分布 $N(\boldsymbol{\mu}_j, \Sigma_j)$ 在 $\boldsymbol{x}_i$ 处的概率密度。直接计算 $l(\boldsymbol{\theta})$ 对每个参数的一阶导数并令其等于零无法求出参数 MLE 的解析形式。如果我们能观察到 $\{z_i\}_{i=1}^n$, 则参数的极大似然估计变得很容易，此时对数似然函数可写为

$$l(\boldsymbol{\theta}) = \sum_{i=1}^n \log p(\boldsymbol{x}_i, z_i \mid \boldsymbol{\theta})$$

$$
\begin{aligned}
&= \sum_{i=1}^{n} \left[\log p(\boldsymbol{x}_i \mid z_i, \boldsymbol{\theta}) + \log p(z_i \mid \boldsymbol{\theta})\right] \\
&= \sum_{j=1}^{K} \left[\left(\sum_{i: z_i = j} \log p(\boldsymbol{x}_i \mid \boldsymbol{\mu}_j, \Sigma_j)\right) + n_j \log \phi_j\right] \qquad (3)
\end{aligned}
$$

其中 $n_j = \sum_{i=1}^{n} \mathbf{1}(z_i = j)$, $j = 1, \ldots, K$. 在限制条件 $\sum_{j=1}^{K} \phi_j = 1$ 下, 最大化(3)可得各参数的 MLE 为

$$
\begin{aligned}
\hat{\phi}_j &= \frac{n_j}{n} \\
\hat{\boldsymbol{\mu}}_j &= \sum_{i: z_i = j} \boldsymbol{x}_i / n_j \\
\hat{\Sigma}_j &= \frac{1}{n_j} \sum_{i: z_i = j} (\boldsymbol{x}_i - \hat{\boldsymbol{\mu}}_j)(\boldsymbol{x}_i - \hat{\boldsymbol{\mu}}_j)^{\top}
\end{aligned}
$$

但是 $\{z_i\}_{i=1}^{n}$ 一般是未知的，此时该如何从(2)中计算各参数的 MLE？可以使用 EM 算法。

2 Jensen's Inequality

首先介绍 EM 算法的原理 — Jensen 不等式。

Theorem 1. X 是一个随机变量，f 是一个凸函数，则有

$$E\left[f(X)\right] \geq f\left(E(X)\right).$$

Proof. 因为 f 是凸函数，在 $\mu = E(X)$ 处，总可以找到一条直线 $l: f(\mu) + \lambda(x - \mu)$ 使得 f 处于 l 的上方，即

$$f(x) \geq f(\mu) + \lambda(x - \mu), \ \forall x. \qquad (4)$$

如果 f 在 $x = \mu$ 处可导，则 $\lambda = f'(\mu)$; 如果 f 在 $x = \mu$ 处不可导，则 λ 可取 $f'(\mu-) \leq \lambda \leq f'(\mu+)$ 的任意值。由(4)可得

$$E\left[f(X)\right] \geq E\left[f(\mu) + \lambda(X - \mu)\right] = f(\mu)$$

□

Remarks

1. 如果 f 是严格凸函数 $(f''(x) > 0)$，则 $E\left[f(X)\right] = f\left(E(X)\right)$ 当且仅当 $X = E(X)$ 以概率 1 成立, 即 X 以概率 1 是常数。

2. 如果 f 是凹函数，则 $-f$ 是凸函数，根据 Jensen's inequality, $E\left[f(X)\right] \leq f\left(E(X)\right)$.

3 EM 算法

对于 n 个独立同分布的样本 $\{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n\}$，假设其对数似然函数可写为

$$\begin{aligned} l(\boldsymbol{\theta}) &= \sum_{i=1}^{n} \log p(\boldsymbol{x}_i \mid \boldsymbol{\theta}) \\ &= \sum_{i=1}^{n} \log\left(\int p(\boldsymbol{x}_i, z_i \mid \boldsymbol{\theta}) dz_i\right) \end{aligned} \tag{5}$$

其中 $\{z_i\}_{i=1}^n$ 是隐变量，但是直接最大化(5)很困难。EM 算法的基本想法是：先找到 $l(\boldsymbol{\theta})$ 的一个下界函数 $g(\boldsymbol{\theta})$, 即 $l(\boldsymbol{\theta}) \geq g(\boldsymbol{\theta}), \forall \boldsymbol{\theta}$，且 $g(\boldsymbol{\theta})$ 是较容易优化的函数 (E-step); 然后找到 $g(\boldsymbol{\theta})$ 的最大值点 (M-step); 不断重复这两步直到收敛，如图1所示。

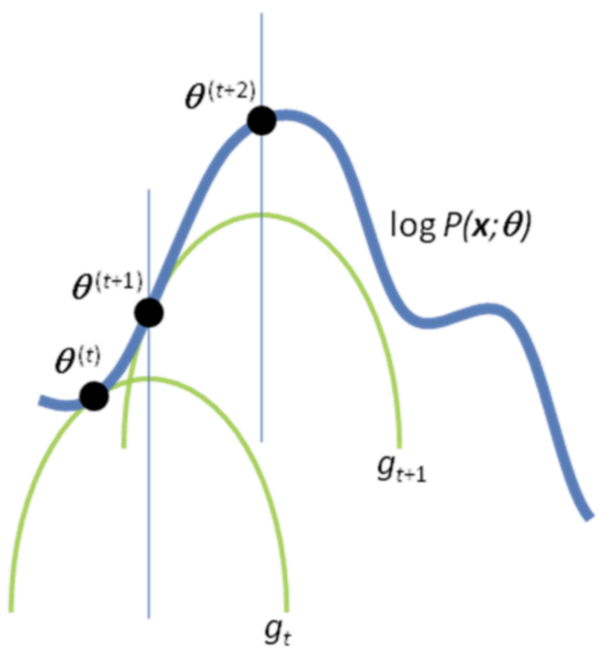

Figure 1: EM 算法的基本想法。

如果隐变量 z_i 是离散变量，$z_i \in \{1, 2, \ldots, K\}, \forall i$, 则(5)可写为

$$l(\boldsymbol{\theta}) = \sum_{i=1}^{n} \log\left(\sum_{j=1}^{K} p(\boldsymbol{x}_i, z_i = j \mid \boldsymbol{\theta})\right). \tag{6}$$

为了找到 $l(\boldsymbol{\theta})$ 的一个下界函数，为每个隐变量 z_i 引入一个离散分布 Q_i. 假设 Q_i 是 $\{1, 2, \ldots, K\}$ 上的离散分布，$i = 1, \ldots, n$, 则(6)可写为

$$l(\boldsymbol{\theta}) = \sum_{i=1}^{n} \log\left(\sum_{j=1}^{K} p(\boldsymbol{x}_i, z_i = j \mid \boldsymbol{\theta})\right)$$

$$
\begin{aligned}
&= \sum_{i=1}^{n} \log\left(\sum_{j=1}^{K} Q_i(z_i=j)\frac{p(\boldsymbol{x}_i, z_i=j \mid \boldsymbol{\theta})}{Q_i(z_i=j)}\right) \\
&= \sum_{i=1}^{n} \log\left[E_{z_i\sim Q_i}\left(\frac{p(\boldsymbol{x}_i, z_i \mid \boldsymbol{\theta})}{Q_i(z_i)}\right)\right] && (7) \\
&\geq \sum_{i=1}^{n} E_{z_i\sim Q_i}\left[\log\left(\frac{p(\boldsymbol{x}_i, z_i \mid \boldsymbol{\theta})}{Q_i(z_i)}\right)\right] && (8) \\
&= \sum_{i=1}^{n}\sum_{j=1}^{K} Q_i(z_i=j)\log\left(\frac{p(\boldsymbol{x}_i, z_i=j \mid \boldsymbol{\theta})}{Q_i(z_i=j)}\right) \triangleq g(\boldsymbol{\theta}) && (9)
\end{aligned}
$$

其中由(7)到(8)是根据 Jensen's inequality: $f(x)=\log(x)$ 是凹函数，且是严格凹函数 $f''(x)=-1/x^2<0,\ x\in\mathbb{R}^+$. 对任意一组分布 $\{Q_i: i=1,\dots,n\}$, (9)给出了 $l(\boldsymbol{\theta})$ 的一个下界函数。如果当前对 $\boldsymbol{\theta}$ 的估计是 $\boldsymbol{\theta}^{(t)}$, 如何选取 Q_i's 使得 $g(\boldsymbol{\theta}^{(t)})$ 尽量靠近 $l(\boldsymbol{\theta}^{(t)})$, 最好满足 $g(\boldsymbol{\theta}^{(t)})=l(\boldsymbol{\theta}^{(t)})$?

如果希望(8)中的不等式在 $\boldsymbol{\theta}^{(t)}$ 处变为等式，需要满足

$$
\frac{p(\boldsymbol{x}_i, z_i \mid \boldsymbol{\theta}^{(t)})}{Q_i(z_i)} \equiv c \tag{10}
$$

其中 c 是不依赖于 z_i 的常数。由条件(10)可得，此时应选取

$$
Q_i(z_i) \propto p(\boldsymbol{x}_i, z_i \mid \boldsymbol{\theta}^{(t)}),\ i=1,\dots,n.
$$

考虑到 $\sum_{j=1}^{K} Q_i(z_i=j)=1,\ \forall i$, 则

$$
Q_i(z_i)=\frac{p(\boldsymbol{x}_i, z_i \mid \boldsymbol{\theta}^{(t)})}{\sum_{j=1}^{K} p(\boldsymbol{x}_i, z_i=j \mid \boldsymbol{\theta}^{(t)})}=\frac{p(\boldsymbol{x}_i, z_i \mid \boldsymbol{\theta}^{(t)})}{p(\boldsymbol{x}_i \mid \boldsymbol{\theta}^{(t)})}=p(z_i \mid \boldsymbol{x}_i, \boldsymbol{\theta}^{(t)}) \tag{11}
$$

即 Q_i 应为给定 $\boldsymbol{x}_i, \boldsymbol{\theta}^{(t)}$ 下 z_i 的条件分布。

假设当前对 $\boldsymbol{\theta}$ 的估计值是 $\boldsymbol{\theta}^{(t)}$，在 EM 算法的 E-step 中，按(11)选取 $Q_i,\ i=1,\dots,n$, 得到 $l(\boldsymbol{\theta})$ 的一个下界函数 $g(\boldsymbol{\theta})$; 在 M-step 中，最大化 $g(\boldsymbol{\theta})$，并将 $\boldsymbol{\theta}$ 的估计值更新为最大值点 $\boldsymbol{\theta}^{(t+1)}$. 可以证明

$$
l(\boldsymbol{\theta}^{(t)}) \leq l(\boldsymbol{\theta}^{(t+1)}).
$$

Proof. 按(11)选取 Q_i's 可使(8)中的等号在 $\boldsymbol{\theta}^{(t)}$ 处成立，则有

$$
l(\boldsymbol{\theta}^{(t)}) = g(\boldsymbol{\theta}^{(t)}) \leq \max_{\boldsymbol{\theta}} g(\boldsymbol{\theta}) = g(\boldsymbol{\theta}^{(t+1)}) \leq l(\boldsymbol{\theta}^{(t+1)}).
$$

□

由于似然函数是有界的，因此 EM 算法可以保证 $l(\boldsymbol{\theta}^{(t)})$ 单调递增收敛。EM 算法可总结为 Algorithm 1.

Algorithm 1 EM Algorithm.

给定数据 $\{\boldsymbol{x}_1,\ldots,\boldsymbol{x}_n\}$ 及 $\boldsymbol{\theta}$ 的初始值 $\boldsymbol{\theta}^{(0)}$.

repeat $t=0,1,\ldots$

(E-step) 将分布 Q_i 选为

$$Q_i(z_i)=p(z_i\mid \boldsymbol{x}_i,\boldsymbol{\theta}^{(t)}),\ i=1,\ldots,n$$

令

$$g(\boldsymbol{\theta})=\sum_{i=1}^n\sum_{j=1}^K Q_i(z_i=j)\log\left(\frac{p(\boldsymbol{x}_i,z_i=j\mid\boldsymbol{\theta})}{Q_i(z_i=j)}\right)$$

(M-step) 计算 $\boldsymbol{\theta}^{(t+1)}=\underset{\boldsymbol{\theta}}{\operatorname{argmax}}\ g(\boldsymbol{\theta})$.

until $l(\boldsymbol{\theta}^{(t+1)})-l(\boldsymbol{\theta}^{(t)})<\epsilon$

return $\boldsymbol{\theta}^{(t+1)}$

4 使用 EM 算法估计 GMM

下面使用 EM 算法估计 GMM 模型(1)的参数 $\boldsymbol{\theta}=\{(\boldsymbol{\mu}_j,\Sigma_j,\phi_j):j=1,\ldots,K\}$.

在 E-step 中，需要先计算每个 z_i 的条件分布

$$\begin{aligned}w_{ij}&=Q_i(z_i=j)=P(z_i=j\mid\boldsymbol{x}_i,\boldsymbol{\theta}^{(t)})\\&\propto p(\boldsymbol{x}_i,z_i=j\mid\boldsymbol{\theta}^{(t)})=p(\boldsymbol{x}_i\mid z_i=j,\boldsymbol{\theta}^{(t)})p(z_i=j\mid\boldsymbol{\theta}^{(t)})\\&\propto p(\boldsymbol{x}_i\mid\boldsymbol{\mu}_j^{(t)},\Sigma_j^{(t)})\phi_j^{(t)},\qquad j=1,\ldots,K;\ i=1,\ldots,n.\end{aligned}$$

其中 $p(\boldsymbol{x}_i\mid\boldsymbol{\mu}_j^{(t)},\Sigma_j^{(t)})$ 是正态分布 $N(\boldsymbol{\mu}_j^{(t)},\Sigma_j^{(t)})$ 在 $\boldsymbol{x}_i$ 处的概率密度。由于对每个 i 有 $\sum_{j=1}^K w_{ij}=\sum_{j=1}^K Q_i(z_i=j)=1$, 因此

$$w_{ij}=\frac{p(\boldsymbol{x}_i\mid\boldsymbol{\mu}_j^{(t)},\Sigma_j^{(t)})\phi_j^{(t)}}{\sum_{k=1}^K p(\boldsymbol{x}_i\mid\boldsymbol{\mu}_k^{(t)},\Sigma_k^{(t)})\phi_k^{(t)}},\quad j=1,\ldots,K;\ i=1,\ldots,n.$$

由此得到 $l(\boldsymbol{\theta})$ 的一个下界函数

$$\begin{aligned}g(\boldsymbol{\theta})&=\sum_{i=1}^n\sum_{j=1}^K w_{ij}\log\left(\frac{p(\boldsymbol{x}_i,z_i=j\mid\boldsymbol{\theta})}{w_{ij}}\right)\\&=\sum_{i=1}^n\sum_{j=1}^K w_{ij}\log\left(\frac{p(\boldsymbol{x}_i\mid z_i=j,\boldsymbol{\theta})p(z_i=j\mid\boldsymbol{\theta})}{w_{ij}}\right)\\&=\sum_{i=1}^n\sum_{j=1}^K w_{ij}\left[\log\left(p(\boldsymbol{x}_i\mid\boldsymbol{\mu}_j,\Sigma_j)\right)+\log(\phi_j)-\log(w_{ij})\right]\end{aligned}$$

$$= \sum_{i=1}^{n}\sum_{j=1}^{K} w_{ij}\left[-\frac{1}{2}\log(|\Sigma_j|) - \frac{1}{2}(\boldsymbol{x}_i-\boldsymbol{\mu}_j)^\top\Sigma_j^{-1}(\boldsymbol{x}_i-\boldsymbol{\mu}_j) + \log(\phi_j) + \cdots\right]$$

此处省略了与 $\{(\boldsymbol{\mu}_j,\Sigma_j,\phi_j): j=1,\ldots,K\}$ 无关的项。

在 M-step 中，我们希望选取 $\boldsymbol{\theta}=\{(\boldsymbol{\mu}_j,\Sigma_j,\phi_j): j=1,\ldots,K\}$ 使 $g(\boldsymbol{\theta})$ 达到最大。首先对 $g(\boldsymbol{\theta})$ 关于 $\{\phi_j\}_{j=1}^K$ 优化，此时最大化 $g(\boldsymbol{\theta})$ 等价于

$$\max_{\phi_1,\ldots,\phi_K}\sum_{i=1}^{n}\sum_{j=1}^{K} w_{ij}\log(\phi_j)$$

注意到 $\{\phi_j\}_{j=1}^K$ 还需满足条件 $\sum_{j=1}^K\phi_j=1$，因此建立如下 Lagrangian:

$$L(\phi_1,\ldots,\phi_K) = \sum_{i=1}^{n}\sum_{j=1}^{K} w_{ij}\log(\phi_j) + \lambda\left(\textstyle\sum_{j=1}^{K}\phi_j - 1\right) \tag{12}$$

Lagrangian (12)关于每个 ϕ_j 的偏导数为

$$\frac{\partial L}{\partial\phi_j} = \sum_{i=1}^{n}\frac{w_{ij}}{\phi_j} + \lambda, \quad j=1,\ldots,K.$$

令上式等于 0 解得

$$\phi_j = -\frac{\sum_{i=1}^{n} w_{ij}}{\lambda}, \quad j=1,\ldots,K.$$

利用限制条件 $\sum_{j=1}^K\phi_j=1$ 解得 $\hat\lambda = -\sum_{i=1}^n\sum_{j=1}^K w_{ij} = -n$. 代入上式得到对 ϕ_j's 的新的估计:

$$\phi_j^{(t+1)} = \frac{1}{n}\sum_{i=1}^{n} w_{ij}, \quad j=1,\ldots,K.$$

注意此时得到的最优解一定满足 $\phi_j^{(t+1)}\ge 0$, $\forall j$, 因此并不需要在 Lagrangian (12) 中加入限制条件 $\phi_j\ge 0$, $j=1,\ldots,K$.

接下来对 $g(\boldsymbol{\theta})$ 关于 $\boldsymbol{\mu}_j$ 优化, $j=1,\ldots,K$. $g(\boldsymbol{\theta})$ 关于 $\boldsymbol{\mu}_j$ 的梯度为:

$$\nabla_{\boldsymbol{\mu}_j} g(\boldsymbol{\theta}) = -\sum_{i=1}^{n} w_{ij}\Sigma_j^{-1}(\boldsymbol{x}_i-\boldsymbol{\mu}_j) = \Sigma_j^{-1}\left(\boldsymbol{\mu}_j\sum_{i=1}^{n} w_{ij} - \sum_{i=1}^{n} w_{ij}\boldsymbol{x}_i\right)$$

令其等于零，解得最优的 $\boldsymbol{\mu}_j$ 为

$$\boldsymbol{\mu}_j^{(t+1)} = \frac{\sum_{i=1}^{n} w_{ij}\boldsymbol{x}_i}{\sum_{i=1}^{n} w_{ij}}, \quad j=1,\ldots,K.$$

利用矩阵微积分或仿照 Wishart 分布 MLE 的证明可得最优的 Σ_j 为

$$\Sigma_j^{(t+1)} = \frac{\sum_{i=1}^{n} w_{ij}(\boldsymbol{x}_i-\boldsymbol{\mu}_j^{(t+1)})(\boldsymbol{x}_i-\boldsymbol{\mu}_j^{(t+1)})^\top}{\sum_{i=1}^{n} w_{ij}}, \quad j=1,\ldots,K.$$

5 MM 算法

EM 算法可以看作更一般的 MM 算法 (Lange et al., 2000) 的一个特例。MM 算法是 minorization-maximization principle 的简称，它最大化目标函数 $f(\boldsymbol{\theta})$ 的思路是：先在当前估计点 $\boldsymbol{\theta}^{(t)}$ 附近寻找一个代理函数 (surrogate function) $g(\boldsymbol{\theta} \mid \boldsymbol{\theta}^{(t)})$，$g$ 需要满足两个条件：

$$
\begin{aligned}
f(\boldsymbol{\theta}^{(t)}) &= g(\boldsymbol{\theta}^{(t)} \mid \boldsymbol{\theta}^{(t)}) \\
f(\boldsymbol{\theta}) &\geq g(\boldsymbol{\theta} \mid \boldsymbol{\theta}^{(t)}),\ \forall \boldsymbol{\theta}
\end{aligned}
\tag{13}
$$

该过程被称为 minorization, 它代表 MM 算法的第一个 M, 函数 g 也被称为 minorizing function. MM 算法的第二个 M 是指最大化 (maximize) 代理函数 $g(\boldsymbol{\theta} \mid \boldsymbol{\theta}^{(t)})$, 令

$$
\boldsymbol{\theta}^{(t+1)} = \underset{\boldsymbol{\theta}}{\operatorname{argmax}}\ g(\boldsymbol{\theta} \mid \boldsymbol{\theta}^{(t)}).
$$

则有

$$
f(\boldsymbol{\theta}^{(t+1)}) \geq g(\boldsymbol{\theta}^{(t+1)} \mid \boldsymbol{\theta}^{(t)}) \geq g(\boldsymbol{\theta}^{(t)} \mid \boldsymbol{\theta}^{(t)}) = f(\boldsymbol{\theta}^{(t)}). \tag{14}
$$

这种单调递增性保证了 MM 算法的收敛。事实上，MM 算法只需要保证每步 $g(\boldsymbol{\theta}^{(t+1)} \mid \boldsymbol{\theta}^{(t)}) \geq g(\boldsymbol{\theta}^{(t)} \mid \boldsymbol{\theta}^{(t)})$，不一定要找到 $g(\boldsymbol{\theta} \mid \boldsymbol{\theta}^{(t)})$ 的最大值点。

5.1 Variance Components Model

方差成分 (variance components) 模型在基因研究和生物医学领域有广泛应用。对于含有 n 个样本的数据，用 $n \times 1$ 向量 $\boldsymbol{y}$ 储存所有观察结果 (response vector), 用 $n \times p$ 矩阵 X 储存所有预测变量的值，最简单的 variance components model 假设

$$
\begin{aligned}
\boldsymbol{y} &\sim N_n(X\boldsymbol{\beta}, \Omega) \\
\Omega &= \sum_{j=1}^{m} \sigma_j^2 V_j
\end{aligned}
\tag{15}
$$

其中 $V_1, \ldots, V_m$ 是 m 个已知的对称正定矩阵，在实际问题中一般由不同群体的历史数据估计得到。模型(15)要估计的参数是系数向量 $\boldsymbol{\beta} \in \mathbb{R}^p$ 和方差成分权重 $\boldsymbol{\sigma}^2 = (\sigma_1^2, \ldots, \sigma_m^2)$.

模型(15)的对数似然函数为

$$
L(\boldsymbol{\beta}, \boldsymbol{\sigma}^2) = -\frac{1}{2} \ln [\det(\Omega)] - \frac{1}{2} (\boldsymbol{y} - X\boldsymbol{\beta})^\top \Omega^{-1} (\boldsymbol{y} - X\boldsymbol{\beta}). \tag{16}
$$

Zhou et al. (2019) 使用 MM 算法最大化(16)估计模型(15)的 MLE. 使用迭代策略轮流更新参数 $\boldsymbol{\beta}$ 和 $\boldsymbol{\sigma}^2$ 时, 给定 $\boldsymbol{\sigma}^2$ 当前的估计量 $\boldsymbol{\sigma}^2_{(t)}$，很容易得到 $\boldsymbol{\beta}$ 的最优解：

$$
\boldsymbol{\beta}^{(t+1)} = \left(X^\top \Omega_{(t)}^{-1} X \right)^{-1} X^\top \Omega_{(t)}^{-1} \boldsymbol{y}. \tag{17}
$$

但是给定 $\boldsymbol{\beta}^{(t+1)}$, 很难找到使(16)最大化的 $\boldsymbol{\sigma}^2$ 的解析解。Zhou et al. (2019) 使用以下两个引理构造了 $\boldsymbol{\sigma}^2$ 的一个容易优化的 minorization function.

Lemma 1.

$$-\ln\left[\det(\Omega)\right] \geq -\ln\left[\det(\Omega_{(t)})\right] - tr\left[\Omega_{(t)}^{-1}(\Omega - \Omega_{(t)})\right]. \tag{18}$$

Proof.

$$\begin{aligned}
\ln\det(\Omega) &= \ln\det(\Omega_{(t)} + \Omega - \Omega_{(t)}) \\
&= \ln\det\left[\Omega_{(t)}^{1/2}\left(I + \Omega_{(t)}^{-1/2}(\Omega - \Omega_{(t)})\Omega_{(t)}^{-1/2}\right)\Omega_{(t)}^{1/2}\right] \\
&= \ln\det(\Omega_{(t)}) + \ln\det\left[I + \Omega_{(t)}^{-1/2}(\Omega - \Omega_{(t)})\Omega_{(t)}^{-1/2}\right]
\end{aligned} \tag{19}$$

令矩阵 $\Omega_{(t)}^{-1/2}(\Omega - \Omega_{(t)})\Omega_{(t)}^{-1/2}$ 的特征值为 $\lambda_1, \ldots, \lambda_n$, 则

$$\ln\det\left[I + \Omega_{(t)}^{-1/2}(\Omega - \Omega_{(t)})\Omega_{(t)}^{-1/2}\right] = \sum_{i=1}^{n}\ln(1+\lambda_i). \tag{20}$$

利用不等式 $\log(1+x) \leq x, \forall x > -1$ 可得

$$\begin{aligned}
\ln\det(\Omega) &= \ln\det(\Omega_{(t)}) + \sum_{i=1}^{n}\ln(1+\lambda_i) \\
&\leq \ln\det(\Omega_{(t)}) + \sum_{i=1}^{n}\lambda_i \\
&= \ln\det(\Omega_{(t)}) + \operatorname{tr}\left[\Omega_{(t)}^{-1/2}(\Omega - \Omega_{(t)})\Omega_{(t)}^{-1/2}\right] \\
&= \ln\det(\Omega_{(t)}) + \operatorname{tr}\left[\Omega_{(t)}^{-1}(\Omega - \Omega_{(t)})\right].
\end{aligned}$$

□

引理1为(16)中右式第一项找到了满足条件的下界函数。(16)中右式第二项可以写为:

$$(\boldsymbol{y} - X\boldsymbol{\beta})^\top\Omega^{-1}(\boldsymbol{y} - X\boldsymbol{\beta}) = (\boldsymbol{y} - X\boldsymbol{\beta})^\top\Omega_{(t)}^{-1}\left[\Omega_{(t)}\Omega^{-1}\Omega_{(t)}\right]\Omega_{(t)}^{-1}(\boldsymbol{y} - X\boldsymbol{\beta}). \tag{21}$$

引入一个新符号 $\preceq$, 如果矩阵 $(B - A)$ 半正定，记为 $A \preceq B$. Boyd and Vandenberghe (2004) 证明了矩阵函数 $f(A, B) = A^\top B^{-1}A$ 对任意 $m \times n$ 矩阵 A 及 $m \times m$ 的正定矩阵 B 是凸 (convex) 函数，即 $\forall\lambda \in [0, 1]$,

$$\begin{aligned}
f\left[\lambda A_1 + (1-\lambda)A_2, \lambda B_1 + (1-\lambda)B_2\right] &\preceq \lambda f(A_1, B_1) + (1-\lambda)f(A_2, B_2) \\
\left[\lambda A_1 + (1-\lambda)A_2\right]^\top\left[\lambda B_1 + (1-\lambda)B_2\right]^{-1}\left[\lambda A_1 + (1-\lambda)A_2\right] &\preceq \lambda A_1^\top B_1^{-1}A_1 + (1-\lambda)A_2^\top B_2^{-1}A_2.
\end{aligned} \tag{22}$$

利用该凸函数的性质可以证得以下引理:

Lemma 2.

$$\Omega_{(t)}\Omega^{-1}\Omega_{(t)} \preceq \sum_{j=1}^{m} \frac{\sigma_{j,t}^4}{\sigma_j^2} V_j \tag{23}$$

其中 $\sigma_{j,t}$ 表示 σ_j 在第 t 步的估计量。

Proof. 根据凸函数 $f(A,B) = A^\top B^{-1} A$ 的性质(22)可得

$$\begin{aligned}
\Omega_{(t)}\Omega^{-1}\Omega_{(t)} &= \left(\sum_{j=1}^{m} \sigma_{j,t}^2 V_j\right)\left(\sum_{j=1}^{m} \sigma_j^2 V_j\right)^{-1}\left(\sum_{j=1}^{m} \sigma_{j,t}^2 V_j\right) \\
&= \left(\sum_{j=1}^{m} \frac{\sigma_{j,t}^2}{\sum_k \sigma_{k,t}^2} \frac{\sum_k \sigma_{k,t}^2}{\sigma_{j,t}^2} \sigma_{j,t}^2 V_j\right)\left(\sum_{j=1}^{m} \frac{\sigma_{j,t}^2}{\sum_k \sigma_{k,t}^2} \frac{\sum_k \sigma_{k,t}^2}{\sigma_{j,t}^2} \sigma_j^2 V_j\right)^{-1}\left(\sum_{j=1}^{m} \frac{\sigma_{j,t}^2}{\sum_k \sigma_{k,t}^2} \frac{\sum_k \sigma_{k,t}^2}{\sigma_{j,t}^2} \sigma_{j,t}^2 V_j\right) \\
&\preceq \sum_{j=1}^{m} \frac{\sigma_{j,t}^2}{\sum_k \sigma_{k,t}^2}\left(\frac{\sum_k \sigma_{k,t}^2}{\sigma_{j,t}^2} \sigma_{j,t}^2 V_j\right)\left(\frac{\sum_k \sigma_{k,t}^2}{\sigma_{j,t}^2} \sigma_j^2 V_j\right)^{-1}\left(\frac{\sum_k \sigma_{k,t}^2}{\sigma_{j,t}^2} \sigma_{j,t}^2 V_j\right) \\
&= \sum_{j=1}^{m} \frac{\sigma_{j,t}^4}{\sigma_j^2} V_j
\end{aligned}$$

□

由引理(2)及(21)得

$$(\boldsymbol{y} - X\boldsymbol{\beta})^\top \Omega^{-1} (\boldsymbol{y} - X\boldsymbol{\beta}) \leq (\boldsymbol{y} - X\boldsymbol{\beta})^\top \Omega_{(t)}^{-1} \left(\sum_{j=1}^{m} \frac{\sigma_{j,t}^4}{\sigma_j^2} V_j\right) \Omega_{(t)}^{-1} (\boldsymbol{y} - X\boldsymbol{\beta}). \tag{24}$$

根据不等式(18)和(24)，我们找到了对数似然函数 $L(\boldsymbol{\sigma}^2)$ 的一个下界函数 (minorization):

$$\begin{aligned}
g(\boldsymbol{\sigma}^2 \mid \boldsymbol{\sigma}_{(t)}^2) &= -\frac{1}{2}\mathrm{tr}\left(\Omega_{(t)}^{-1}\Omega\right) - \frac{1}{2}(\boldsymbol{y} - X\boldsymbol{\beta}^{(t+1)})^\top \Omega_{(t)}^{-1}\left(\sum_{j=1}^{m} \frac{\sigma_{j,t}^4}{\sigma_j^2} V_j\right)\Omega_{(t)}^{-1}(\boldsymbol{y} - X\boldsymbol{\beta}^{(t+1)}) + c^{(t)} \\
&= \sum_{j=1}^{m}\left[-\frac{\sigma_j^2}{2}\mathrm{tr}\left(\Omega_{(t)}^{-1} V_j\right) - \frac{\sigma_{j,t}^4}{2\sigma_j^2}(\boldsymbol{y} - X\boldsymbol{\beta}^{(t+1)})^\top \Omega_{(t)}^{-1} V_j \Omega_{(t)}^{-1}(\boldsymbol{y} - X\boldsymbol{\beta}^{(t+1)})\right] + c^{(t)}
\end{aligned} \tag{25}$$

其中 $c^{(t)}$ 是与 $\boldsymbol{\sigma}^2$ 无关的常数。利用一阶导数条件，很容易找出使(25)最大的 $\boldsymbol{\sigma}^2$ 的解析解:

$$\sigma_{j,t+1}^2 = \sigma_{j,t}^2 \sqrt{\frac{(\boldsymbol{y} - X\boldsymbol{\beta}^{(t+1)})^\top \Omega_{(t)}^{-1} V_j \Omega_{(t)}^{-1} (\boldsymbol{y} - X\boldsymbol{\beta}^{(t+1)})}{\mathrm{tr}\left(\Omega_{(t)}^{-1} V_j\right)}},\ j = 1, \ldots, m.$$

习题: 编程实现 EM 算法，并用如下数据和初始值估计一个 two-component Gaussian mixture model. 使用 contour plot 展示估计的正态分布。

```
# create dataset
library(MASS)
set.seed(123)
n=1000
mu1 = c(0,4)
mu2 = c(-2,0)
Sigma1 = matrix(c(3,0,0,0.5),nr=2,nc=2)
Sigma2 = matrix(c(1,0,0,2),nr=2,nc=2)
phi = c(0.6,0.4)
X = matrix(0,nr=2,nc=n)

for (i in 1:n){
  if (runif(1)<=phi[1]){
    X[,i] = mvrnorm(1,mu=mu1,Sigma=Sigma1)
  }else{
    X[,i] = mvrnorm(1,mu=mu2,Sigma=Sigma2)
  }
}

# initial guess for parameters
mu10 = runif(2)
mu20 = runif(2)
Sigma10 = diag(2)
Sigma20 = diag(2)
phi0 = runif(2)
phi0 = phi0/sum(phi0)
```

References

Boyd, S. and Vandenberghe, L. (2004). *Convex optimization*. Cambridge university press.

Lange, K., Hunter, D. R., and Yang, I. (2000). Optimization transfer using surrogate objective functions. *Journal of computational and graphical statistics*, 9(1):1–20.

Zhou, H., Hu, L., Zhou, J., and Lange, K. (2019). Mm algorithms for variance components models. *Journal of Computational and Graphical Statistics*, 28(2):350–361.