统计中的计算方法 · 课后作业（1）

梁子龙（15300180026）

2018 年 3 月 24 日

作业 1 假设 $X=\{\boldsymbol{x}_i\}_{i=1}^n$ 是 n 个来自于 k 个 d 维正态分布的混合分布的独立样本，即

$$f(X)=\sum_{j=1}^k \tau_j f_j(X), \tag{1}$$

其中 $f_i(X)$ 为 d 维空间正态分布的概率密度函数. 试推导出用 *EM* 方法估计 τ_j 及正态分布的均值及协方差矩阵的迭代步骤.

答. 多维正态分布的概率函数为

$$f(\boldsymbol{x};\boldsymbol{\mu},\Sigma)=\frac{1}{\sqrt{(2\pi)^d\,|\Sigma|}}\exp\left(-\frac{1}{2}\left(\boldsymbol{x}-\boldsymbol{\mu}\right)^{\mathrm{T}}\Sigma^{-1}\left(\boldsymbol{x}-\boldsymbol{\mu}\right)\right), \tag{2}$$

我们引入变量

$$z_{ij}=\begin{cases}1, & \text{若样本 } \boldsymbol{x_i} \text{ 由分布 } N\left(\boldsymbol{\mu}_j,\Sigma_j\right) \text{ 得到;}\\ 0, & \text{若样本 } \boldsymbol{x_i} \text{ 不由分布 } N\left(\boldsymbol{\mu}_j,\Sigma_j\right) \text{ 得到;}\end{cases}\quad (j=1,2,\ldots,k). \tag{3}$$

这样，似然函数可以表示为

$$L(\theta;\boldsymbol{x},\boldsymbol{z})=\prod_{i=1}^n\prod_{j=1}^k\left(\tau_j f(\boldsymbol{x}_i;\boldsymbol{\mu}_j,\Sigma_j)\right)^{z_{ij}}; \tag{4}$$

取对数，得到对数似然函数

$$\log L(\theta;\boldsymbol{x},\boldsymbol{z})=\sum_{i=1}^n\sum_{j=1}^k z_{ij}\left(\log\tau_j-\frac{1}{2}\log|\Sigma|-\frac{1}{2}\left(\boldsymbol{x}_i-\boldsymbol{\mu}_j\right)^{\mathrm{T}}\Sigma_j^{-1}\left(\boldsymbol{x}_i-\boldsymbol{\mu}_j\right)-\frac{d}{2}\log 2\pi\right). \tag{5}$$

为使用 EM 算法，需对引入的变量 z_{ij} 求期望进行迭代. 设 $T_{ij}^{(t)}=\mathrm{E}(z_{ij}|\boldsymbol{x_i},\theta^{(t)})$，那么依期

望计算公式，有

$$
\begin{aligned}
T_{ij}^{(t)} &= \mathrm{P}(z_{ij}=1|\boldsymbol{x}_i,\theta^{(t)}) = \frac{\mathrm{P}(z_{ij}=1,\boldsymbol{x}_i,\theta^{(t)})}{\mathrm{P}(\boldsymbol{x}_i,\theta^{(t)})} \\
&= \frac{\tau_j^{(t)} f(\boldsymbol{x}_i;\boldsymbol{\mu}_j^{(t)},\Sigma_j^{(t)})}{\sum_{l=1}^{k}\tau_l^{(t)} f(\boldsymbol{x}_i;\boldsymbol{\mu}_l^{(t)},\Sigma_l^{(t)})}.
\end{aligned} \tag{6}
$$

利用计算得到的 $T_{ij}^{(t)}$，代入对数似然函数 (5)，得到

$$
\begin{aligned}
Q(\theta|\theta^{(t)}) &= \mathrm{E}(\log L(\theta;\boldsymbol{x},\boldsymbol{z})) \\
&= \sum_{i=1}^{n}\sum_{j=1}^{k} T_{ij}^{(t)}\left(\log\tau_j - \frac{1}{2}\log|\Sigma_j| - \frac{1}{2}(\boldsymbol{x}_i-\boldsymbol{\mu}_j)^{\mathrm{T}}\Sigma_j^{-1}(\boldsymbol{x}_i-\boldsymbol{\mu}_j) - \frac{d}{2}\log 2\pi\right).
\end{aligned} \tag{7}
$$

下面逐一考察各个参数的迭代更新．首先考察 τ_j．将已经得到的 $Q(\theta|\theta^{(t)})$ 对 τ_j 求极值，得到一个条件极值问题，并利用 Lagrange 乘数法，得到

$$
\begin{cases}
\dfrac{\partial Q}{\partial\tau_j}+\lambda = \dfrac{1}{\tau_j}\displaystyle\sum_{i=1}^{n} T_{ij}^{(t)}+\lambda = 0, \quad (j=1,2,\ldots,k); \\
\displaystyle\sum_{j=1}^{k}\tau_j = 1.
\end{cases} \tag{8}
$$

求解该方程组，得到

$$
\tau_j^{(t+1)} = \frac{\sum_{i=1}^{n} T_{ij}^{(t)}}{n}. \tag{9}
$$

接下来考察 $\boldsymbol{\mu}_j$．为对其求极值，令

$$
\begin{aligned}
\frac{\partial Q}{\partial\boldsymbol{\mu}_j} &= \frac{\partial}{\partial\boldsymbol{\mu}_j}\left(\sum_{i=1}^{n} -\frac{1}{2}T_{ij}^{(t)}(\boldsymbol{x}_i-\boldsymbol{\mu}_j)^{\mathrm{T}}\Sigma_j^{-1}(\boldsymbol{x}_i-\boldsymbol{\mu}_j)\right) \\
&= -\frac{1}{2}\sum_{i=1}^{n} T_{ij}^{(t)}\frac{\partial}{\partial\boldsymbol{\mu}_j}\left((\boldsymbol{x}_i-\boldsymbol{\mu}_j)^{\mathrm{T}}\Sigma_j^{-1}(\boldsymbol{x}_i-\boldsymbol{\mu}_j)\right) \\
&= \left(\sum_{i=1}^{n} T_{ij}^{(t)}(\boldsymbol{x}_i-\boldsymbol{\mu}_j)^{\mathrm{T}}\right)\Sigma_j^{-1} = 0,
\end{aligned} \tag{10}
$$

得到

$$
\boldsymbol{\mu}_j^{(t+1)} = \frac{\sum_{i=1}^{n} T_{ij}^{(t)}\boldsymbol{x}_i}{\sum_{i=1}^{n} T_{ij}^{(t)}}. \tag{11}
$$

最后考察 Σ_j．类似地，令

$$\begin{aligned}
\frac{\partial Q}{\partial \Sigma_j} &= \sum_{i=1}^{n} T_{ij}^{(t)} \left(-\frac{1}{2}\frac{1}{|\Sigma_j|}\frac{\partial |\Sigma_j|}{\partial \Sigma_j} - \frac{1}{2}\frac{\partial}{\partial \Sigma_j}\left((\boldsymbol{x}_i - \boldsymbol{\mu}_j)^{\mathrm{T}} \Sigma_j^{-1} (\boldsymbol{x}_i - \boldsymbol{\mu}_j) \right) \right) \\
&= \sum_{i=1}^{n} T_{ij}^{(t)} \left(-\frac{1}{2}\frac{1}{|\Sigma_j|} |\Sigma_j| \Sigma_j^{-1} + \frac{1}{2} (\boldsymbol{x}_i - \boldsymbol{\mu}_j)(\boldsymbol{x}_i - \boldsymbol{\mu}_j)^{\mathrm{T}} \Sigma_j^{-2} \right) \\
&= \sum_{i=1}^{n} T_{ij}^{(t)} \left(-\frac{1}{2}\Sigma_j^{-1} + \frac{1}{2} (\boldsymbol{x}_i - \boldsymbol{\mu}_j)(\boldsymbol{x}_i - \boldsymbol{\mu}_j)^{\mathrm{T}} \Sigma_j^{-2} \right) = 0,
\end{aligned} \tag{12}$$

得到

$$\Sigma_j^{(t+1)} = \frac{\sum_{i=1}^{n} T_{ij}^{(t)} \left(\boldsymbol{x}_i - \boldsymbol{\mu}_j^{(t+1)} \right) \left(\boldsymbol{x}_i - \boldsymbol{\mu}_j^{(t+1)} \right)^{\mathrm{T}}}{\sum_{i=1}^{n} T_{ij}^{(t)}}. \tag{13}$$

至此，所需估计参数的迭代过程已全部构建完毕．现将以上 EM 算法总结如下：

1. 适当选取参数的初始值；
2. E-Step：依 (6) 求得 $T_{ij}^{(t)}$；
3. M-Step：依 (9), (11), (13) 求得 $\tau_j^{(t+1)}, \boldsymbol{\mu}_j^{(t+1)}, \Sigma_j^{(t+1)}$；
4. 重复 E-Step 与 M-Step，直至参数结果收敛． □

作业 2 *将上述方法应用于数据 Data1.csv 来估计参数．*

表 1: EM 算法实验结果

	$j=1$	$j=2$	$j=3$
τ_j	0.497	0.203	0.300
$\boldsymbol{\mu}_j$	$(5.024, -2.048)$	$(2.105, 0.959)$	$(-1.980 - 3.035)$
Σ_j	$\begin{pmatrix} 2.166 & 0.154 \\ 0.154 & 0.868 \end{pmatrix}$	$\begin{pmatrix} 2.179 & 0.005 \\ 0.005 & 0.898 \end{pmatrix}$	$\begin{pmatrix} 1.312 & 0.017 \\ 0.017 & 0.908 \end{pmatrix}$

答. 依据上述方法，本文使用 R 完成了这个实验，代码实现已附在本文末尾．在 EM 算法的主要函数 `gmm` 中使用了一些必要的向量化方法以加快运行速度；在取初始值时，选用了均匀分布的随机数生成 $\{\tau_j^{(0)}\}_{j=1}^{k}$ 与 $\{\boldsymbol{\mu}_j^{(0)}\}_{j=1}^{k}$，并选用单位矩阵作为初始协方差矩阵 $\{\Sigma_j^{(0)}\}_{j=1}^{k}$．实验选用诸参数的范数的相对误差来判断收敛；使用程序迭代约 35 次后，估计得到实验数据如表 1 所示，大致的概率密度如图 1 所示． □

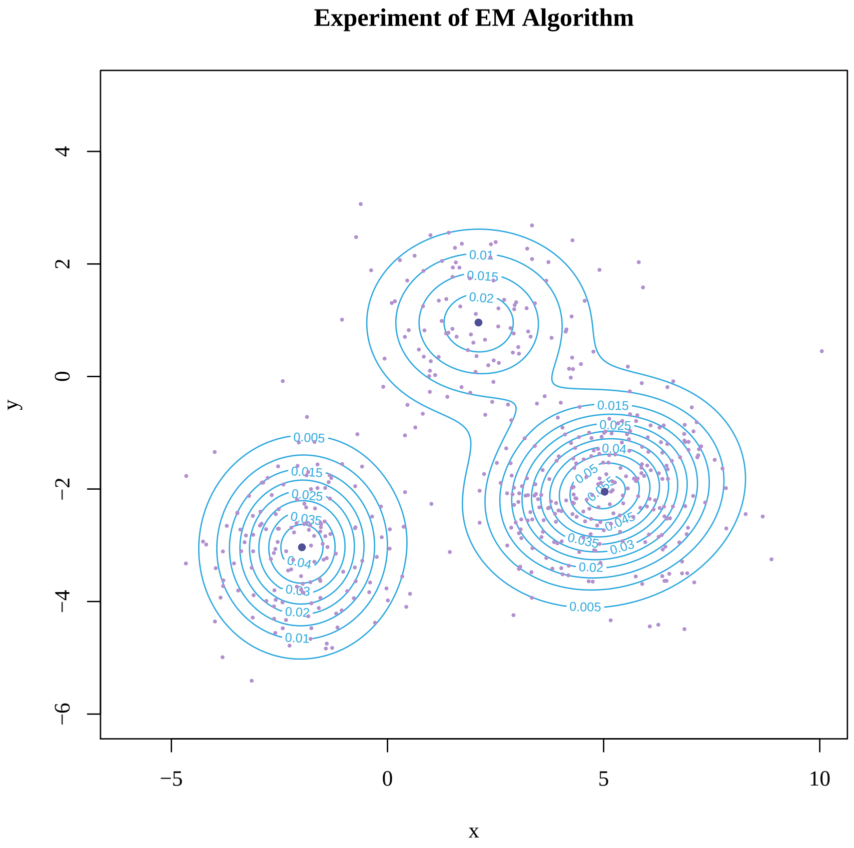

图 1: EM 算法实验结果．图中紫色散点为原始样本的数据点，蓝色散点为组成混合分布的三个正态分布的均值点，等高线为混合分布的概率密度函数．

附录：作业中用到的矩阵求导技术

1. 行列式的求导．记 $A=(a_{ij})\in\mathbb{R}^{n\times n}$，$A^*=(A_{ji})_{n\times n}$ 为 A 的伴随矩阵，有

$$\frac{\partial\,|A|}{\partial A}=\left(\frac{\partial\,|A|}{\partial a_{ij}}\right)_{n\times n}=(A_{ij})_{n\times n}=(A^*)^{\mathrm{T}};$$

   当 A 为对称矩阵时，有

$$(A^*)^{\mathrm{T}}=A^*=|A|\,A^{-1}.$$

2. 二次型对向量求导．又记 $x=\{x_i\}_{i=1}^n\in\mathbb{R}^n$，有

$$\begin{aligned}\frac{\partial x^{\mathrm{T}}Ax}{\partial x_i}&=\frac{\partial\left(\sum_{k=1}^n\sum_{l=1}^n a_{lk}x_lx_k\right)}{\partial x_i}\\&=\sum_{k=1}^n a_{ik}x_k+\sum_{l=1}^n a_{li}x_l,\end{aligned}$$

则

$$\frac{\partial x^{\mathrm{T}}Ax}{\partial x} = x^{\mathrm{T}}(A + A^{\mathrm{T}});$$

当 A 为对称矩阵时，有

$$\frac{\partial x^{\mathrm{T}}Ax}{\partial x} = 2x^{\mathrm{T}}A.$$

3. 二次型对矩阵求导．对 A 的每一个元素 a_{ij}，有

$$\frac{\partial x^{\mathrm{T}}Ax}{\partial a_{ij}} = \frac{\partial \left(\sum_{k=1}^{n}\sum_{l=1}^{n} a_{lk}x_l x_k\right)}{\partial a_{ij}} = x_i x_j,$$

则

$$\frac{\partial x^{\mathrm{T}}Ax}{\partial A} = xx^{\mathrm{T}}.$$

附录：EM 算法实验的 R 代码（`assignment01.R`）

```r
# ------------------------------------------------------------
# Statistical Computing - Assignment
# ------------------------------------------------------------
# EM Algorithm for Multivariate Gaussian Mixture Distribution
# Author: Liang Zilong (ID: 15300180026)
# Date: 2018-03-23
# ------------------------------------------------------------

library("mvtnorm")

# -----------------------------
# Main function of EM Algorithm
# -----------------------------

gmm <- function(x, k, tol = 1e-6, iter.max = 200) {
  # Check the samples
  d <- ncol(x)
  n <- nrow(x)

  # Initialize the parameters
  tau <- runif(k - 1, 0, 1 / k); tau <- c(tau, 1 - sum(tau))
  mu <- matrix(runif(d * k, 0, 5), nrow = k, ncol = d)
  sigma <- array(rep(diag(d), k), dim = c(d, d, k))

  # EM iterations
  iter <- 0  # Iteration index
  repeat {
    # E-Step
    f <- matrix(nrow = n, ncol = k)
    for (j in 1:k) {  # TODO: vectorization?
      f[, j] <- dmvnorm(x, mu[j, ], sigma[, , j])
    }
    e.weighted <- f %*% tau
```

```
    e <- matrix(0, nrow = n, ncol = k)
    for (j in 1:k) {  # TODO: vectorization?
      e[, j] <- f[, j] * tau[j]
    }
    e <- e / rowSums(e.weighted)

    # M-Step
    sum.e <- colSums(e)
    tau.new <- sum.e / n
    mu.new <- t(e) %*% x / sum.e
    sigma.new <- array(rep(0, d * d* k), dim = c(d, d, k))
    for (j in 1:k) {  # TODO: vectorization?
      for (i in 1:n) {
        sigma.new[, , j] <- sigma.new[, , j] + e[i, j] *
                            ((x[i, ] - mu.new[j, ]) %o% (x[i, ] - mu.new[j, ]))
      }
      sigma.new[, , j] <- sigma.new[, , j] / sum.e[j]
    }

    # Judge convergence
    iter <- iter + 1
    if (iter > iter.max) { break }
    err.tau <- norm(rbind(tau.new - tau), "I") / rbind(tau)
    err.mu <- norm(mu.new - mu, "I") / norm(mu, "I")
    err.sigma <- norm(colSums(sigma.new - sigma), "I") / norm(colSums(sigma), "I")
    err.max <- max(c(err.tau, err.mu, err.sigma))
    if (err.max < tol) { break }

    # Iterate the parameters
    tau <- tau.new
    mu <- mu.new
    sigma <- sigma.new
  }

  return (list(tau, mu, sigma, iter))
}


# ----------
# Experiment
# ----------

# Read sample data
x <- read.csv("assignment01-data.csv")
x <- as.matrix(x[, 2:3])

# Estimate parameters
estimates <- gmm(x, 3)
tau <- estimates[[1]]
mu <- estimates[[2]]
```

```r
sigma <- estimates[[3]]

# Prepare plotting
pfunc <- function(pp, tau, mu, sigma) {
  # PDF of Gaussian Mixture Distribution
  k = length(tau)
  zz <- 0
  for (j in 1:k) {
    zz <- zz + tau[j] * dmvnorm(pp, mu[j, ], sigma[, , j])
  }
  return (zz)
}
pnum <- 300
xx <- seq(-6, 10, length.out = pnum)
yy <- seq(-6, 5, length.out = pnum)
pp <- cbind(rep(xx, pnum), sort(rep(yy, pnum)))  # Plotting points
zz <- matrix(pfunc(pp, tau, mu, sigma), pnum, pnum)

# Plotting
contour(xx, yy, zz, xlab = "x", ylab = "y", family = "serif", col = "#2fa9df")
points(x, pch = 20, cex = 0.4, col = "#b28fce")
points(estimates[[2]], pch = 20, cex = 1, col ="#4e4f97")
title("Experiment of EM Algorithm", family = "serif")
```