SML HW4

29-176004 奥村 恭平 [*,†]

May 22, 2018

宿題 1

記法は断りのない限りスライド中のものをそのまま用いる．$\phi_{ij} := \phi(x_i; \mu_j, \sigma_j)$, $S_e := \sum_{j=1}^m \exp[\gamma_j]$.

$$l(\theta) := \log L(\theta) = \sum_{i=1}^n \log \left[\sum_{j=1}^m w_j(\gamma_1, \dots, \gamma_m) \phi(x_i; \mu_j, \sigma_j) \right]$$

(i) $\widehat{w}_k = \frac{1}{n} \sum_{i=1}^n \widehat{\eta}_{ik}$

$\because$)

$$\frac{\partial l}{\partial \gamma_k} = \sum_i \frac{\sum_j \frac{\partial w_j}{\partial \gamma_k} \cdot \phi_{ij}}{\sum_j w_j \phi_{ij}}$$

ここで,

$$\begin{aligned}
\sum_j \frac{\partial w_j}{\partial \gamma_k} \cdot \phi_{ij} &= -\sum_{j \neq k} \frac{\exp[\gamma_j + \gamma_k]}{S_e^2} \phi_{ij} + \frac{\exp[\gamma_k] S_e - \exp[2\gamma_k]}{S_e^2} \phi_{ik} \\
&= -\sum_{j \neq k} w_j w_k \phi_{ij} + (w_k - w_k^2) \phi_{ik} \\
&= w_k \left[-\sum_{j=1}^m w_j \phi_{ij} + \phi_{ik} \right]
\end{aligned}$$

であることより,

$$\begin{aligned}
& \frac{\partial l}{\partial \gamma_k} = 0 \\
\iff & \sum_i \frac{-\sum_j w_j \phi_{ij} + \phi_{ik}}{\sum_j w_j \phi_{ij}} = 0 \\
\iff & \sum_i \left[-1 + \frac{\phi_{ik}}{\sum_j w_j \phi_{ij}} \right] = 0 \\
\iff & n = \sum_i \frac{\phi_{ik}}{\sum_j w_j \phi_{ij}} \\
\iff & n \cdot w_k = \sum_i \frac{w_k \phi_{ik}}{\sum_j w_j \phi_{ij}} \\
\iff & w_k = \frac{1}{n} \sum_i \eta_{ik}
\end{aligned}$$

以上より，(i) の式が示された．

[*] E-mail: kyohei.okumura@gmail.com
[†] 東京大学大学院 経済学研究科 M2

(ii) $\widehat{\sigma}_k = \sqrt{\frac{1}{d} \cdot \frac{\sum_i \widehat{\eta}_{ik}(x_i - \widehat{\mu}_j)^\top (x_i - \widehat{\mu}_j)}{\sum_i \widehat{\eta}_{ik}}}$

$\because$) $s_k := \sigma_k^2$ とする.

$$\frac{\partial l}{\partial s_k} = \sum_i \frac{w_k \frac{\partial \phi}{\partial s_k}}{\sum_j w_j \phi_{ij}}$$

ここで,

$$\begin{aligned}
\frac{\partial \phi}{\partial s_k} &= \frac{\partial}{\partial s_k}\left[\frac{1}{(2\pi s_k)^{\frac{d}{2}}} \exp\left[-\frac{(x_i-\mu_k)^\top (x_i-\mu_k)}{2s_k}\right]\right] \\
&= \left[2\pi\left(-\frac{d}{2}\right)\frac{1}{(2\pi s_k)^{\frac{d}{2}+1}} \exp\left[-\frac{(x_i-\mu_k)^\top (x_i-\mu_k)}{2s_k}\right]\right] \\
&\quad + \left[\frac{1}{(2\pi s_k)^{\frac{d}{2}}} \exp\left[-\frac{(x_i-\mu_k)^\top (x_i-\mu_k)}{2s_k}\right] \frac{2(x_i-\mu_k)^\top (x_i-\mu_k)}{4s_k^2}\right] \\
&= -\frac{d}{2s_k}\phi_{ik} + \frac{(x_i-\mu_k)^\top (x_i-\mu_k)}{2s_k^2}\phi_{ik}
\end{aligned}$$

であることより,

$$\begin{aligned}
&\frac{\partial l}{\partial s_k} = 0 \\
\iff &\sum_i \frac{-\frac{d}{2s_k} w_k \phi_{ik} + \frac{(x_i-\mu_k)^\top (x_i-\mu_k)}{2s_k^2} w_k \phi_{ik}}{\sum_j w_j \phi_{ij}} = 0 \\
\iff &\sum_i \frac{-d s_k w_k \phi_{ik} + (x_i-\mu_k)^\top (x_i-\mu_k) w_k \phi_{ik}}{\sum_j w_j \phi_{ij}} = 0 \\
\iff &\sum_i (-d s_k + (x_i-\mu_k)^\top (x_i-\mu_k))\eta_{ik} = 0 \\
\iff & d s_k \sum_i \eta_{ik} = \sum_i (x_i-\mu_k)^\top (x_i-\mu_k)\eta_{ik} \\
\iff & s_k = \frac{1}{d} \cdot \frac{\sum_i \eta_{ik}(x_i-\mu_k)^\top (x_i-\mu_k)}{\sum_i \eta_{ik}} \\
\therefore \quad & \sigma_k = \sqrt{\frac{1}{d} \cdot \frac{\sum_i \eta_{ik}(x_i-\mu_k)^\top (x_i-\mu_k)}{\sum_i \eta_{ik}}}
\end{aligned}$$

となり, (ii) の結果を得る.

(iii) $\widehat{\mu}_k = \frac{\sum_i \widehat{\eta}_{ik} x_i}{\sum_i \widehat{\eta}_{ik}}$

$\because$)

$$\frac{\partial l}{\partial \mu_k} = \sum_i \frac{w_k \frac{\partial \phi}{\partial \mu_k}}{\sum_j w_j \phi_{ij}}$$

ここで,

$$\frac{\partial \phi}{\partial \mu_k} = \phi_{ik}\left[-\frac{1}{2\sigma_k^2} 2(x_i - \mu_k)\right]$$

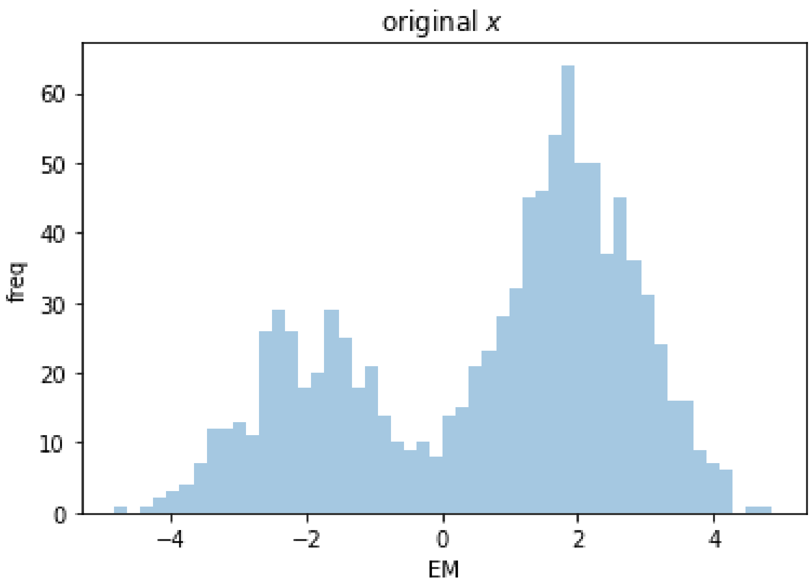
であるので，

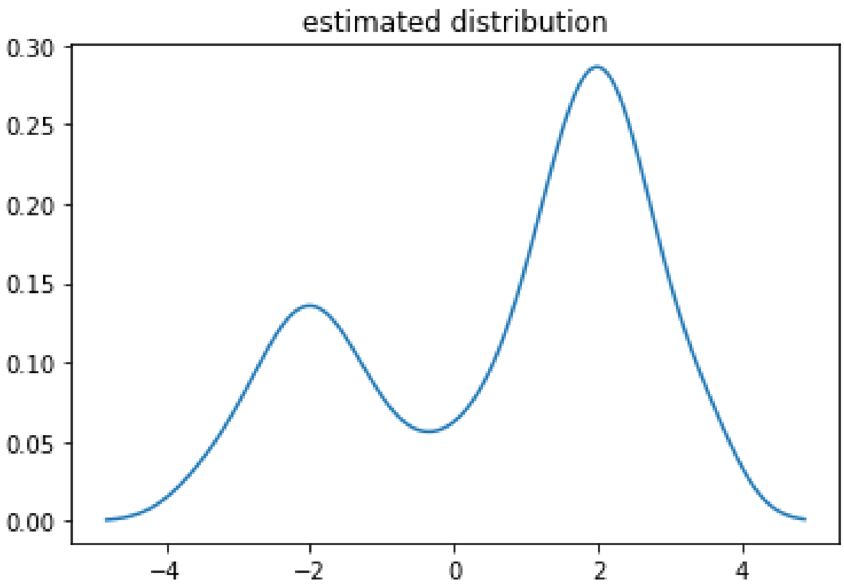
$$
\begin{aligned}
&\frac{\partial l}{\partial \mu_k} = 0 \\
\iff &\sum_i \frac{(x_i - \mu_k) w_k \phi_{ik}}{\sum_j w_j \phi_{ij}} = 0 \\
\iff &\sum_i (x_i - \mu_k) \eta_{ik} = 0 \\
\therefore \quad &\mu_k = \frac{\sum_i \eta_{ik} x_i}{\sum_i \eta_{ik}}
\end{aligned}
$$

となり，(iii) の結果を得る． □

宿題 2

末尾のコード (言語は python) を用いてシミュレーションした．(プロットなどに必要な関数は省略．) 上手く元の分布を推定できていることがわかる．





```python
import numpy as np
import scipy as sp
from numpy.random import randn, rand
from scipy.stats import norm

def em_gauss_sim(n_sample=1000, n_component=5, iter_lim=20, seed=7):
    # set seed
    np.random.seed(seed)

    x = randn(n_sample) + (rand(n_sample) > 0.3) * 4 - 2

    L = - np.inf

    # initial values
    weights = np.ones(n_component) / n_component
    means = np.linspace(np.min(x), np.max(x), n_component)
    covs = np.ones(n_component)/10
    eta = np.zeros([n_sample, n_component])

    # main loop
    n_iter = 0
    while n_iter < iter_lim:
        # E step
        for i in range(n_sample):
            for l in range(n_component):
                eta[i][l] = (weights[l] * norm.pdf(x=x[i], loc=means[l], scale=np.sqrt(covs[l]))) / np.dot(weights, norm.pdf(x=x[i], loc=means, scale=np.sqrt(covs)))

        # M step
        means_prev = means.copy()
        weights_prev = weights.copy()
        covs_prev = covs.copy()
        for l in range(n_component):
            weights[l] = np.sum(eta[:,l]) / n_sample
            covs[l] = np.dot(eta[:,l], (x - means_prev[l])**2) / (1 * np.sum(eta[:,l]))
            means[l] = np.dot(eta[:,l], x) / np.sum(eta[:, l])

        # stop condition
        if np.sum((weights - weights_prev)**2) < 0.00001:
            break

        n_iter += 1

    # for debugging
    print(n_iter)

    sim_x = np.linspace(np.min(x), np.max(x), 1000)
    y = np.zeros_like(sim_x)
    for l in range(n_component):
        tmp = weights[l] * norm.pdf(x=sim_x, loc=means[l], scale=np.sqrt(covs[l]))
        y += tmp

    return x, sim_x, y
```