Exam M2: Model-based statistical learning

Charles Bouveyron, Pierre-Alexandre Mattei, Aude Sportisse

28/01/2022

Exercise 1: discriminant analysis Student's t distribution (5 pts)

The idea is to redo the LDA lab we did in class (link here (https://github.com/cbouveyron/MBSL-Course2021/blob/main/Notebook-LDA.Rmd)), but with Student's t classes (with different covariances and different numbers of degrees of freedom) instead of Gaussians.

We assume the following model, where $x \in \mathbb{R}^D$ is a data point, and $k \in \{1, \dots, K\}$ is a class:

$$p(x|Z = k) = \mathcal{S}(x; \mu_k, \Sigma_k, \nu)$$

and $P(Z = k) = \pi_k$. The function $\mathcal{S}$ corresponds to the density of a multivariate Student's t distribution (link here (https://en.wikipedia.org/wiki/Multivariate_t-distribution)).

1. **What is the maximum-likelihood estimate of the class proportions $\pi_1, \dots, \pi_K$?**

We have K classes in a dataset. As such the class is a discrete random variable with K outcomes. This can be modeled with a multinomial distribution with parameters $\forall k \in K, \pi_k$. These parameters are stored in a K-dimensional vector π sucht at $p_i \geq 0$ and π sums to 1 and correspond to the proportions of the classes.

We can draw from the class and set a 1-of-K representation $t = (t_1, \dots t_k)^T$ such that the probability of belonging to a class k is described as:

$$p(t|\pi) = \pi_1^{t_1} * \cdots * \pi_3^{t_k}$$

In this situation, the MLE of π is given by

$$\pi = (\frac{N_1}{N}, \dots, \frac{N_k}{N})$$

I.e. the vector of observed proprtions for all k classes in the dataset.

---

2. **Fit our model using maximum likelihood to the same data we used for the LDA lab in class**.

To help you, here's a quick way to fit a multivariate Student's t distribution to some data using a package. First we load the data:

```
library("fitHeavyTail")
library("mvtnorm")
library("pgmm")

data(wine)

X = wine[,-1]
z = wine$Type
```

Then we use the function fit_mvt:

```
res = fit_mvt(X, nu = "iterative",ftol = 1e-9)
```

*We can evaluate the log-likelihood of the data from the output of fit_mvt, or using dmvt from the mvtnorm package:**

```
print(res$log_likelihood)
```

```
## [1] -11016.75
```

```
print(sum(dmvt(X,delta = res$mu, sigma = res$scatter,df = res$nu, log = TRUE)))
```

```
## [1] -11016.75
```

In the lab, we used a simplified version of the dataset `wine` such that we only kept the `Alcohol` and `Fixed Acidity` columns as part of our features while also reducing the class setup as `Barolo` vs. `Not Barolo` (classification of wine types).

```
X = as.matrix(wine[,c(2,4)])
dim(X)
```

```
## [1] 178   2
```

```
colnames(X)
```

```
## [1] "Alcohol"       "Fixed Acidity"
```

```
z = as.numeric(wine$Type!=1) + 1
```

We now declare our LDA function:

```
lda.learn <- function(X,z){
  ###
  # LINEAR DISCRIMINANT ANALYSIS WITH STUDENT T DISTRIBUTION
  # LEARNING STEP
  ###
  # retrieves parameters
  n = nrow(X); p = ncol(X)
  K = max(z)
  # declares data structures
  prop = rep(NA,K)
  ll = rep(NA, K)
  nu = rep(NA, K)
  mu = matrix(NA,K,p)
  Sigma = matrix(0,p,p)
  Sigma = matrix(0,p,p)
  # loops over classes
  for (k in 1:K){
    nk = sum(z == k)
    prop[k] =  nk / n
    # Fits
    Student_fit = fit_mvt(
      X[z == k,],
      nu = "iterative",
      ftol = 1e-9
    )
    # Records the fitted parameters of the class
    mu[k,] = Student_fit$mu
    Sigma = Sigma + nk / n * Student_fit$cov
    # Records the ll
    ll[k] = Student_fit$log_likelihood
    nu[k] = Student_fit$nu
  }
  return(list(prop = prop, mu = mu, Sigma = Sigma,
              log_likelihood=ll, nu=nu))
}

#lda.predict <- function(X,params){
#  K = length(params$prop)
#  Prob = matrix(NA,nrow(X),K)
#  for (k in 1:K){
#    Prob[,k] = params$prop[k] * dmvt(X,delta = params$mu[k,],
#                                     sigma = params$Sigma,
#                                     df = params$nu[k],
#                                     log = TRUE)
#  }
#  Prob = t(apply(Prob,1,function(x){x / sum(x)}))
#  zstar = apply(Prob,1,which.max)
#  return(list(Prob = Prob,zstar = zstar))
#}

# We fit our LDA with Student's T Distribution to the
# observed data
params = lda.learn(X,z)
```

Now that we have fitted our model to the observations using the Student's t distribution, we can print the resulting parameters:

```
params
```

```
## $prop
## [1] 0.3314607 0.6685393
##
## $mu
##          [,1]     [,2]
## [1,] 13.74410 74.32295
## [2,] 12.62933 90.01734
##
## $Sigma
##                 Alcohol Fixed Acidity
## Alcohol       0.4084682      2.599095
## Fixed Acidity 2.5990955    306.862655
##
## $log_likelihood
## [1] -260.5319 -643.8762
##
## $nu
## [1]  4.371736 99.999928
```

Exercise 2: Gaussian mixtures for MCAR data (20 pts)

Let $X \in \mathbb{R}^{n \times d}$ a dataset which contains missing values. The missing-data pattern $M \in \{0,1\}^{n \times d}$ is a binary matrix which indicates where are the missing values in X: $m_{ij} = 1$ is x_{ij} is missing, $m_{ij} = 0$ otherwise. Let us denote x_i^{obs} (resp. x_i^{mis}) the values of the observed variable(s) (resp. the values of the missing variable(s)) for the individual i.

For example, if we observe $(\text{NA}, 2, 3, \text{NA})$ and the true values are $(1, 2, 3, 4)$, $x_i^{\text{obs}} = (2, 3)$ and $x_i^{\text{mis}} = (1, 4)$.

The aim of model-based clustering is to estimate an (unknown) partition of the data X in K groups. This partition is denoted as $Z = (z_1 | \dots | z_n)^T \in \{0,1\}^{n \times K}$ with $z_i = (z_{i1}, \dots, z_{iK})^T \in \{0,1\}^K$ and where $z_{ik} = 1$ if x_i belongs to the class k, $z_{ik} = 0$ otherwise. In this context, note that both x_i^{mis} and z_i are missing. In the following, we assume that the data are missing completely at random (MCAR).

Introduction (~2 pts)

1. **Give the definition of the MCAR mechanism and what it implies for the statistical analysis.**

We consider i.i.d. data $x_1, \dots, x_n$ ($x_i \in \mathbb{R}^d$) genereted under a Gaussian mixture model with K clusters and the following density:

$$p_\theta(x) = \sum_{k=1}^{K} \pi_k f_k(x; \mu_k, \Sigma_k),$$

with $\theta = (\pi_1, \dots, \pi_K, \mu_1, \dots, \mu_K, \Sigma_1, \dots, \Sigma_K)$ and f_k is the Gaussian density function. $\forall k \in \{1, \dots, K\}, \mu_k \in \mathbb{R}^d$ and $\Sigma_k \in \mathbb{R}^{d \times d}$.

MCAR or missing completely at random is the situation where the probability of a data point missing is case independent (e.g. it is because of bad luck). As such, the missing-data mechanism in the case of MCAR is represented by:

$$P(M|X;\phi)$$

Where $P(X)$ is the distribution of the data such that $P(X, M; \theta, \phi) = P(X; \theta)P(M|X;\phi)$.

In the case of MCAR data, we also have the following:

$$P(M|X;\phi) = P(M;\phi) \text{ (Rubin, 1976)}$$

Furthermore, for statistical analysis of MCAR data, it happens that we can ignore the missing-data mechanism:

$$L_{full,obs}(\theta, \phi; X^{obs}, M) \propto L_{ign}(\theta; X^{obs}) = \int p(X;\theta) dX^{mis} = P(X^{obs};\theta)$$

Ignorability in the case of MCAR to model (X, M) is possible, treating M's parameter ϕ as a nuisance parameter.

---

2. Recall in which spaces the parameters live.

$$\text{Let } X \in \mathbb{R}^{n \times d},\ M \in \{0,1\}^{n \times d}$$
$$\forall K \in \mathbb{N}_+$$
$$\forall k \in \{1, \dots, K\},$$
$$\pi_k \in [0,1],\ \text{given } \sum_{k=1}^{K} \pi_k = 1$$
$$\mu_k \in \mathbb{R}^d$$
$$\Sigma_k \in \mathbb{R}^{d \times d}$$

The goal of this exercise is to implement an EM algorithm for a Gaussian mixture model when the data are missing (MCAR).

The EM algorithm is an iterative algorithm that permits to maximize the likelihood function under missingness. Let be $\theta^{[0]}$ the initialization point. The iteration $[r]$ of the algorithm consists in proceeding:

- the **E-step**: computation of $Q(\theta; \theta^{[r-1]}) = \mathbb{E}[\ell(\theta; X, Z) | X^{\text{obs}}; \theta^{[r-1]}]$, where $\ell(\theta; X, Z)$ is the complete log-likelihood.
- the **M-step**: update of the parameters by maximizing the function $Q(\theta; \theta^{[r-1]})$:

$$\theta^{[r]} = \text{argmax}_\theta Q(\theta; \theta^{[r-1]}).$$

Note that

$$\ell(\theta; X, Z) = \log\left(\prod_{i=1}^{n} p_\theta(x_i)\right) = \sum_{i=1}^{n}\sum_{k=1}^{K} z_{ik} \log(\pi_k f_k(x_i; \mu_k, \Sigma_k))$$

We can show that

$$Q(\theta; \theta^{[r-1]} = \sum_{i=1}^{n}\sum_{k=1}^{K} t_{ik}(\theta^{[r-1]})[\log(\pi_k) + \tau_x(\mu_k, \Sigma_k; x_i^{\text{obs}}, \theta^{[r-1]})]$$

E-step (~ 6pts)

1. Explicit the terms t_{ik} and τ_x as conditional probability and conditional expectation.

As part of the course we saw that:

$$t_{ik}(\theta^{[r-1]}) = \mathbb{E}\left[z_{ik} | \theta^{[r-1]}\right]$$

And

$$\begin{aligned} P(z_{ik}|\theta^{[r-1]}) * P(x_i^{obs}|Z_{ik}, \theta^{[r-1]}) &= \pi_k * \phi(\mu_k, \Sigma_k; x_i^{\text{obs}}, \theta^{[r-1]}) \\ \tau_x &= log(\phi(x_i^{obs}; \mu_k, \Sigma_k, \theta^{[r-1]})) \\ &= P(x_i^{obs}|Z_{ik}, \theta^{[r-1]}) \end{aligned}$$

---

2. Write the term t_{ik}.

$$t_{ik} \propto \pi_k . \phi(x_i^{obs}; \mu_k, \Sigma_k, \theta^{[r-1]})$$

To compute τ_x, we have to make explicit the distribution $f(x_i^{\text{mis}} | x_i^{\text{obs}}, z_{ik} = 1; \theta^{[r-1]})$.

Up to a reorganization of the variables, we have

$$f(x_i | z_{ik} = 1; \theta^{[r-1]}) = f\left(\begin{pmatrix} x_i^{\text{obs}} \\ x_i^{\text{mis}} \end{pmatrix} | z_{ik} = 1; \theta^{[r-1]}\right) = \mathcal{N}\left(\begin{pmatrix} (\mu_{ik}^{\text{obs}})^{[r-1]} \\ (\mu_{ik}^{\text{mis}})^{[r-1]} \end{pmatrix}, \begin{pmatrix} (\Sigma_{ik}^{\text{obs,obs}})^{[r-1]} & (\Sigma_{ik}^{\text{obs,mis}})^{[r-1]} \\ (\Sigma_{ik}^{\text{obs,mis}})^{[r-1]} & (\Sigma_{ik}^{\text{mis,mis}})^{[r-1]} \end{pmatrix}\right),$$

where

- $(\mu_{ik}^{\text{obs}})^{[r-1]}$ (resp. $(\mu_{ik}^{\text{mis}})^{[r-1]}$) is the values of the vector $(\mu_{ik})^{[r-1]}$ for the observed variables (resp. for the missing variables),
- $(\Sigma_{ik}^{\text{obs,obs}})^{[r-1]}$ is sub-matrix of $\Sigma_{ik}^{[r-1]}$ for the observed variables only, …

Therefore, we have:

$$\left(x_i^{\text{mis}} \mid x_i^{\text{obs}}, z_{ik} = 1; \theta^{[r-1]}\right) \sim \mathcal{N}\left((\tilde{\mu}_{ik}^{\text{mis}})^{[r-1]}, (\tilde{\Sigma}_{ik}^{\text{mis}})^{[r-1]}\right)$$

3. Explicit $(\tilde{\mu}_{ik}^{\text{mis}})^{[r-1]}$ and $(\tilde{\Sigma}_{ik}^{\text{mis}})^{[r-1]}$.

Hint: use the classic formulae for the conditional expectation of Gaussian variables https://en.wikipedia.org/wiki/Multivariate_normal_distribution#Conditional_distributions (https://en.wikipedia.org/wiki/Multivariate_normal_distribution#Conditional_distributions).

The term τ_x is written as follows:

$$\tau_x(\mu_k, \Sigma_k; x_i^{\text{obs}}, \theta^{[r-1]}) = \frac{1}{2}[n\log(2\pi) + \log((\mid \Sigma_k \mid))] - \frac{1}{2}\mathbb{E}\left[(x_i - \mu_k)^T(\Sigma_k)^{-1}(x_i - \mu_k) \mid x_i^{\text{obs}}, z_{ik} = 1; \theta^{[r-1]}\right].$$

This last term could be expressed using the commutativity and linearity of the trace function:

$$\begin{aligned}&\mathbb{E}\left[(x_i - \mu_k)^T(\Sigma_k)^{-1}(x_i - \mu_k) \mid x_i^{\text{obs}}, z_{ik} = 1; \theta^{[r-1]}\right] \\ &= \text{tr}(\mathbb{E}\left[(x_i - \mu_k)(x_i - \mu_k)^T \mid x_i^{\text{obs}}, z_{ik} = 1, c_i; \theta^{[r-1]}\right](\Sigma_k)^{-1}).\end{aligned}$$

Therefore, to compute τ_x, only $\mathbb{E}\left[(x_i - \mu_k)(x_i - \mu_k)^T \mid x_i^{\text{obs}}, z_{ik} = 1; \theta^{[r-1]}\right]$ has to be calculated.

4. Compute τ_x.

Hint: up to a reorganization of the variables, we have

$$(x_i - \mu_k)(x_i - \mu_k)^T = \begin{pmatrix} (x_i^{\text{obs}} - \mu_{ik}^{\text{obs}})^T(x_i^{\text{obs}} - \mu_{ik}^{\text{obs}}) & (x_i^{\text{obs}} - \mu_{ik}^{\text{obs}})^T(x_i^{\text{mis}} - \mu_{ik}^{\text{mis}}) \\ (x_i^{\text{mis}} - \mu_{ik}^{\text{mis}})^T(x_i^{\text{obs}} - \mu_{ik}^{\text{obs}}) & (x_i^{\text{mis}} - \mu_{ik}^{\text{mis}})^T(x_i^{\text{mis}} - \mu_{ik}^{\text{mis}}) \end{pmatrix}.$$

You can compute the expected value for each block.

To conclude, the E-step consists of computing the following quantities: $(\tilde{\mu}_{ik}^{\text{mis}})^{[r-1]}$, $(\tilde{\Sigma}_{ik}^{\text{mis}})^{[r-1]}$ and $t_{ik}(\theta^{[r-1]}$.

M-step

Let us denote $(\tilde{x}_{ik})^{[r-1]} = (x_i^{\text{obs}}, (\tilde{\mu}_{ik}^{\text{mis}})^{[r-1]})$ and $\tilde{\Sigma}_{ik}^{[r-1]} = \begin{pmatrix} 0_i^{\text{obs,obs}} & 0_i^{\text{obs,mis}} \\ 0_i^{\text{mis,obs}} & (\tilde{\Sigma}_{ik}^{\text{mis}})^{[r-1]} \end{pmatrix}$

The maximization of $Q(\theta; \theta^{[r-1]})$ over (π, μ, Σ) leads to, for $k = 1, \ldots, K$,

$$\pi_k^{[r]} = \frac{1}{n}\sum_{i=1}^{n} t_{ik}(\theta^{[r-1]})$$

$$\mu_k^{[r]} = \frac{\sum_{i=1}^{n} t_{ik}(\theta^{[r-1]})(\tilde{x}_{ik})^{[r-1]}}{\sum_{i=1}^{n} t_{ik}(\theta^{[r-1]})}$$

$$\Sigma_k^{[r]} = \frac{\sum_{i=1}^{n}\left[t_{ik}(\theta^{[r-1]})\left((\tilde{x}_{ik})^{[r-1]} - \mu_k^{[r]})((\tilde{x}_{ik})^{[r-1]} - \mu_k^{[r]})^T + \tilde{\Sigma}_{ik}^{[r-1]}\right)\right]}{\sum_{i=1}^{n} t_{ik}(\theta^{[r-1]})}$$

Code (~ 9pts)

We consider a bivariate Gaussian mixture ($d = 2$) with 2 classes ($K = 2$):

$$X \sim \pi_1\mathcal{N}(\mu_1, \Sigma_1) + \pi_2\mathcal{N}(\mu_2, \Sigma_2),$$

with $\mu_1 = (0, 0)$, $\mu_2 = (3, 3)$, $\pi_1 = 0.3$, $\pi_2 = 0.7$ and $\Sigma_1 = I_{2\times 2}$, $\Sigma_2 = I_{2\times 2}$

We then introduce MCAR values in X (both variables are missing).

```
library(MASS)
library(mvtnorm)
```

```
SimuZ <- function(n,pik){
  #######Arguments
  ##n: number of observations
  ##pik: vector of size K, proportion of the K classes
  #######Values
  ##Return a matrix of size (n,K)

  K <- length(pik)
  Z <- matrix(0,nrow=n,ncol=length(pik))
  obs_pos <- 1:n
  for (k in 1:(K-1)){
    obs_k <- sample(obs_pos,pik[k]*n)
    Z[obs_k,k] <- 1
    obs_pos <- setdiff(obs_pos,obs_k)
  }
  Z[obs_pos,K] <- 1
  return(Z)
}
```

```
set.seed(2)

n <- 100
d <- 2

mu.true <- list(rep(0,d),rep(3,d))
sigma.true <- list(diag(1,d),diag(1,d))

K <- 2
pik.true <-  c(0.3,0.7)

## Generation of the true partition
Z.true <- SimuZ(n=n,pik=pik.true)

Partition_true <- apply(Z.true, 1, function(z) which(z==1))

## Generation of the complete dataset
X <- matrix(NA,nrow=n,ncol=d)
for (k in 1:K){
    obs_k <- which(Z.true[,k]==1)
    X[obs_k, ] <- mvrnorm(n*pik.true[k] , mu.true[[k]] , sigma.true[[k]])
}

## Introduction of missing values in X
prop.miss <- 0.4
nb.miss <- floor(n*d*prop.miss)
missing_idx.mcar <- sample(n*d, nb.miss, replace = FALSE)
XNA <- X
XNA[missing_idx.mcar] <- NA

for (i in 1:n){ #to avoid that one row contains only NA
    if (sum(is.na(XNA[i,]))==d){
      num <- sample(1:d,1)
      XNA[i,num] <- X[i,num]
    }
}
```

```
head(XNA)
```

```
##            [,1]       [,2]
## [1,]  4.5994441         NA
## [2,]  2.5131302  4.9682624
## [3,] -0.2427924  0.8234145
## [4,]         NA  3.0975859
## [5,]  3.1487466  1.7245412
## [6,]  0.1171338 -0.3877827
```

```
sum(is.na(XNA))/(n*d)
```

```
## [1] 0.315
```

1. Propose a naive initialization step for the parameters $\pi^{[0]}$, $\mu^{[0]}$ and $\Sigma^{[0]}$.

```
## Answer (code):
pi.init <- rep(1/K, K)
mu.init <- apply(X, 2, mean, na.rm=TRUE)
sigma.init <- cov(X, use="complete.obs")
```

2. Compute the missing-data pattern M which indicates where are the missing values in X.

Given the implementation of Booleans in R (`1==T` and `0 == FALSE`) we can simply implement M via the `is.na` function.

```
M = is.na(XNA)
```

3. Code the function which returns the observed log-likelihood and t_{ik}.

Recall that the observed log-likelihood is written as

$$\ell(\theta; X^{\text{obs}}) = \sum_{i=1}^{n} \log(f(x_i^{\text{obs}}; \theta)) = \sum_{i=1}^{n} \log\left(\sum_{k=1}^{K} f_k(x_i^{\text{obs}}; \mu_k, \Sigma_k)\pi_k\right)$$

Use the following function **logsumexp** to write the function **Loglikehood_obs_Gaussian** which returns the observed log-likelihood and $t_{ik}, \forall i, \forall k$.

Hint: you can use https://github.com/cbouveyron/MBSL-Course2021/blob/main/GMM_1d_notebook.ipynb (https://github.com/cbouveyron/MBSL-Course2021/blob/main/GMM_1d_notebook.ipynb)

```
logsumexp <- function (x) {
    y = max(x)
    y + log(sum(exp(x - y)))
}
```

```
## Answer (code):

# Arguments:
## XNA: matrix containing missing values (data matrix of size n,d)
## mu: current value of the means (list of length K, with vectors of size d)
## sigma: current value of the covariance matrices (list of length K, with matrices of size d,d)
## prop.pi: current proportion per class (vector of length K)

Loglikelihood_obs_Gaussian <- function(XNA,mu,sigma,prop.pi){

    n <- dim(XNA)[1]
    d <- dim(XNA)[2]
    M <- is.na(XNA)
    K <- length(prop.pi)

    log_tik_numerator <- matrix(0, n, K) #log_tik_numerator[i,k] = log(f(zik=1,y_i^obs))

    Pattern <- matrix(M[!duplicated(M),],nrow=sum(!duplicated(M)))

    for (i in 1:nrow(Pattern)){
        wherePat <- which(sapply(1:nrow(M),function(x) prod(M[x,]==Pattern[i,])==1))
        X_Pat <- rbind(XNA[wherePat,])
        M_Pat <- rbind(M[wherePat,])

        var_obs <- which(Pattern[i,]==0)
        X_obs <- as.matrix(X_Pat[,var_obs])
        mu_obs <- lapply(mu, function(x) x[var_obs])
        sigma_obs <- lapply(sigma , function(x) matrix(x[var_obs,var_obs],ncol=length(var_obs),nrow=lengt
h(var_obs)))

        for (k in 1:K){
            log_tik_numerator[wherePat,k] = log(prop.pi[k]) + dnorm(X_pat, mu_obs[k], sigma_obs[k], log=
T)
        }
    }

    log_tik = log_tik_numerator - apply(log_tik_numerator , 1, logsumexp)
    tik = exp(log_tik)
    loglik_obs = sum(apply(log_tik_numerator , 1, logsumexp))
    return(list(loglik_obs=loglik_obs,tik=tik))
}
```

4. In the function **Loglikelihood_obs_Gaussian**, explain what is the object called Pattern.

Pattern is a matrix (boolean-valued or $\{0,1\}$-valued) that contains the list of missing patterns in the dataset.

```
Pattern <- matrix(M[!duplicated(M),],nrow=sum(!duplicated(M)))
Pattern
```

```
##       [,1]  [,2]
## [1,] FALSE  TRUE
## [2,] FALSE FALSE
## [3,]  TRUE FALSE
```

Here we can see that we have either no missing value, or a missing value in either of the columns (but never both at once, i.e. an empty row).

In the case where the covariance matrix are diagonal, note that the M-step can be written as follows:

$\forall k \in \{1, \dots, K\}$:

- $\pi_k^{[r]} = \frac{1}{n} \sum_{i=1}^{n} t_{ik}(\theta^{[r-1]})$.
- $\forall i \in \{1, \dots, n\}, \quad \tilde{x}_{ik}^{[r-1]} = (\tilde{x}_{ik1}^{[r-1]}, \dots, \tilde{x}_{ikd}^{[r-1]}) \in \mathbb{R}^d$ such that $\forall j \in \{1, \dots, d\}$:

$$\tilde{x}_{ikj}^{[r-1]} = X_{ij} \text{ if } M_{ij} = 0 \text{ (observed) and } \tilde{x}_{ikj}^{[r-1]} = \mu_{kj}^{[r-1]} \text{ if } M_{ij} = 1 \text{ (missing)}.$$

- $\mu_k^{[r]} = (\mu_{k1}^{[r]}, \ldots, \mu_{kd}^{[r]}) \in \mathbb{R}^d$ such that

$$\forall j \in \{1, \ldots, d\}, \quad \mu_{kj}^{[r]} = \frac{\sum_{i=1}^n \tilde{x}_{ikj}^{[r-1]} t_{ik}(\theta^{[r-1]})}{\sum_{i=1}^n t_{ik}(\theta^{[r-1]})}$$

- $\forall i \in \{1, \ldots, n\}, \quad \tilde{\Sigma}_{ik}^{[r-1]} = (\tilde{\Sigma}_{ikjj'}^{[r-1]})_{j \in \{1,\ldots,d\}, j' \in \{1,\ldots,d\}} \in \mathbb{R}^{d \times d}$ such that:

$$\forall j \neq j' \in \{1, \ldots, d\}, \quad \tilde{\Sigma}_{ikjj'}^{[r-1]} = 0$$

and

$$\forall j \in \{1, \ldots, d\}, \quad \tilde{\Sigma}_{ikjj}^{[r-1]} = 0 \text{ if } M_{ij} = 0 \text{ (observed) and } \tilde{\Sigma}_{ikjj}^{[r-1]} = \Sigma_{ikjj}^{[r-1]} \text{ if } M_{ij} = 1 \text{ (missing)}.$$

- $\Sigma_k^{[r]} = (\Sigma_{kjj'}^{[r]})_{j \in \{1,\ldots,d\}, j' \in \{1,\ldots,d\}} \in \mathbb{R}^{d \times d}$ such that:

$$\forall j \neq j' \in \{1, \ldots, d\}, \quad \Sigma_{kjj'}^{[r]} = 0$$

and

$$\forall j \in \{1, \ldots, d\}, \quad \Sigma_{kjj'}^{[r]} = \frac{\sum_{i=1}^n ((\tilde{x}_{ikj} - \mu_{kj}^{[r]})(\tilde{x}_{ikj} - \mu_{kj}^{[r]}) + \tilde{\Sigma}_{ikjj}^{[r-1]}) t_{ik}(\theta^{[r-1]})}{\sum_{i=1}^n t_{ik}(\theta^{[r-1]})}$$

5. Apply the EM algorithm for 50 iterations and plot the observed log-likelihood.

Additional questions (~ 3pts)

1. Describe a procedure to select the number of classes.

Using the Akaike Information Criterion (https://en.wikipedia.org/wiki/Akaike_information_criterion), we can perform the EM algorithm with an increasing number of classes and check with the resulting log likelihood where the process yielded the lowest metric (AIC). I.e.

1. We start with a set of candidate models (1 per class size)
2. We perform the EM algorithm and yield the corresponding Log-likelihood
3. We compute per candidate model the corresponding AIC metric $\eta(M) - log\mathcal{L}(X, \hat{\theta})$ with $\eta(M)$ the number of fine scalar parameters in the model and $log\mathcal{L}$ the resulting log-likelihood
4. We perform model selection by selecting the model with the lowest AIC value

---

2. Write the complete log-likelihood if the mechanism is MNAR.

The likelihood approach with missing data (i.e. maximizing the full observed likelihood) is:

$$\mathcal{L}_{\text{full, obs}}(\theta, \phi; X^{obs}, M) = \int \mathcal{L}_{full}(\theta, \phi; X, M) dX^{mis}$$

In the case of MNAR, we cannot ignore the missing-data mechanism.
