Eyes : Normal Mixture Model

Le ZHANG & Shijie XU

26 mars 2025

1 Introduction

Cette étude analyse les données fournies par Bowmaker et al. (1985), qui concernent les longueurs d'onde de sensibilité maximale des photorécepteurs d'un singe. L'ensemble de données contient 48 observations, et l'objectif est de modéliser et d'estimer les moyennes des deux groupes, leur proportion et leur variance à l'aide d'un modèle de mélange normal (Normal Mixture Model).

2 Modèle et Méthodologie

Nous supposons que chaque observation y_i provient de l'une des deux distributions normales :

$$y_i \sim \mathcal{N}(\lambda_{T_i}, \tau)$$

où :

— $T_i \sim \text{Categorical}(P)$ représente le groupe auquel appartient chaque observation ($T_i = 1$ ou 2).

— La moyenne du premier groupe est λ_1, et celle du deuxième groupe est définie comme $\lambda_2 = \lambda_1 + \theta, \quad \theta > 0.$.

— Les deux groupes partagent une même variance contrôlée par le paramètre de précision τ.

λ_1, θ, τ, P sont dotés de prioris indépendants dits "non-informatifs", y compris un priori uniforme pour P sur $(0, 1)$. Ainsi, nous adoptons les lois a priori suivantes :

$$\theta \sim \mathcal{N}(0, \sigma_\theta^2) \cdot \mathbb{I}_{[0,+\infty]}$$

$$\tau \sim \text{Gamma}(\alpha, \beta)$$

$$\lambda_1 \sim \mathcal{N}(0, \sigma_1^2)$$

$$P \sim \text{Dirichlet}(1, 1)$$

Voici les détails mathématiques de l'échantillonnage MCMC.

En introduisant les variables latentes T_i, la vraisemblance complète s'écrit :

$$\pi(\mathbf{y}, \mathbf{T} \mid \lambda_1, \theta, \tau, P) = \prod_{i=1}^{N} P_{T_i} \cdot \phi_\tau(y_i - \lambda_{T_i})$$

avec $\phi_\tau(y - \lambda)$ la densité d'une loi normale de précision τ :

$$\phi_\tau(y - \lambda) = \sqrt{\frac{\tau}{2\pi}} \exp\left(-\frac{\tau}{2}(y - \lambda)^2\right)$$

L'expression de la loi a posteriori jointe est :

$$\pi(\lambda_1, \theta, \tau, P, \mathbf{T} \mid \mathbf{y}) \propto \pi(\mathbf{y}, \mathbf{T} \mid \lambda_1, \theta, \tau, P) \cdot \pi(\lambda_1) \cdot \pi(\theta) \cdot \pi(\tau) \cdot \pi(P)$$

1. Mettre à jour T_i via une loi de Bernoulli :

$$\mathbb{P}(T_i = 1 \mid y_i, \lambda_1, \theta, \tau, P) = \frac{P_1 \cdot \phi_\tau(y_i - \lambda_1)}{P_1 \cdot \phi_\tau(y_i - \lambda_1) + P_2 \cdot \phi_\tau(y_i - (\lambda_1 + \theta))}$$

   On tire alors T_i selon une loi de Bernoulli avec ce paramètre.

2. Mettre à jour P via une loi Beta : Une fois tous les T_i mis à jour, on compte le nombre d'observations par groupe :

$$n_1 = \sum_{i=1}^{N} 1(T_i = 1), \quad n_2 = N - n_1$$

   Puis, la loi conditionnelle de P est :

$$P \sim \text{Dirichlet}(1 + n_1, 1 + n_2)$$

   Soit :

$$P_1 \sim \text{Beta}(1 + n_1, 1 + n_2), \quad P_2 = 1 - P_1.$$

3. Mettre à jour (λ_1, θ) via Metropolis-Hastings : Puisque $\lambda_2 = \lambda_1 + \theta$ et que $\theta > 0$, il n'existe pas de mise à jour conjuguée. On utilise donc un échantillonnage Metropolis-Hastings sur (λ_1, θ) :

$$\pi(\lambda_1, \theta \mid T, y_i, \tau) \propto \pi(y_i \mid \lambda_1, \theta, \tau, T) \cdot \pi(\lambda_1) \cdot \pi(\theta)$$

$$\log \pi(\lambda_1, \theta \mid T, y_i, \tau) \propto \underbrace{\log \pi(\lambda_1) + \log \pi(\theta)}_{\text{petit terme}} + \log \pi(y_i \mid \lambda_1, \theta, \tau, T)$$

$$\propto \sum_{i:T_i=1} \left[-\frac{\tau}{2}(y_i - \lambda_1)^2\right] + \sum_{i:T_i=2} \left[-\frac{\tau}{2}(y_i - (\lambda_1 + \theta))^2\right]$$

   On échantillonne (λ_1, θ) conjointement via Metropolis-Hastings, en imposant $\theta > 0$.

4. Mettre à jour τ via une loi Gamma :

$$\pi(\tau \mid \lambda, \theta, T, y) \propto \pi(y \mid \tau, \lambda, \theta, T) \cdot \pi(\tau)$$

$$\propto \prod_{i=1}^{N} \left[\tau^{\frac{1}{2}} \exp\left(-\frac{\tau}{2}(y_i - \lambda_{T_i})^2 \right) \right] \cdot \tau^{\alpha-1} e^{-\beta\tau}$$

$$\propto \tau^{\alpha+\frac{N}{2}-1} \exp\left[-\tau \left(\beta + \frac{1}{2} \sum_{i=1}^{N} (y_i - \lambda_{T_i})^2 \right) \right]$$

$$\tau \sim \text{Gamma}\left(\alpha + \frac{N}{2}, \beta + \frac{1}{2} \sum_{i=1}^{N} (y_i - \lambda_{T_i})^2 \right)$$

5. Mettre à jour σ via $\sigma = 1/\sqrt{\tau}$.

3 Résultats et Analyse

Les résultats numériques et les graphiques des chaînes simulées sont présentés en annexe.

Après 10 000 itérations avec 1 000 burn-in, les estimations postérieures sont stables et bien définies. Les moyennes estimées $\lambda_1 = 536.70$ et $\lambda_2 = 548.96$ confirment la distinction entre deux sous-populations. La proportion estimée $P_1 = 0.6018$ indique qu'environ 60 % des observations relèvent du premier groupe, contre $P_2 = 0.3982$ pour le second. L'écart-type commun $\sigma = 3.733$ suggère une dispersion modérée. Ces résultats, cohérents avec la littérature, montrent une bonne convergence du MCMC et des estimations robustes.

L'inspection visuelle des chaînes simulées confirme cette stabilité : toutes les chaînes présentent une bonne mélangeabilité et une stationnarité apparente après le burn-in, sans dérive ni piégeage local. Les paramètres λ_1, θ et P_1 oscillent de manière régulière autour de leur moyenne postérieure.

Les erreurs de Monte Carlo étant très inférieures à l'écart-type (ex. : MC_error = 0.0095 vs. SD = 0.9058 pour λ_1), la précision des estimations est élevée et l'échantillon postérieur suffisant. La combinaison des diagnostics graphiques et numériques renforce la fiabilité de l'inférence bayésienne.

Enfin, la re-paramétrisation de Robert (1994), $\lambda_2 = \lambda_1 + \theta$ avec $\theta > 0$, permet d'éviter le switching de labels. Nos résultats valident cette approche : les chaînes de λ_1 et λ_2 sont bien séparées et aucune attribution excessive des observations à un seul groupe n'a été observée. Cela améliore clairement l'identifiabilité du modèle et la stabilité du MCMC.

Annexe

A1 DAG

Voici la DAG correspondant :

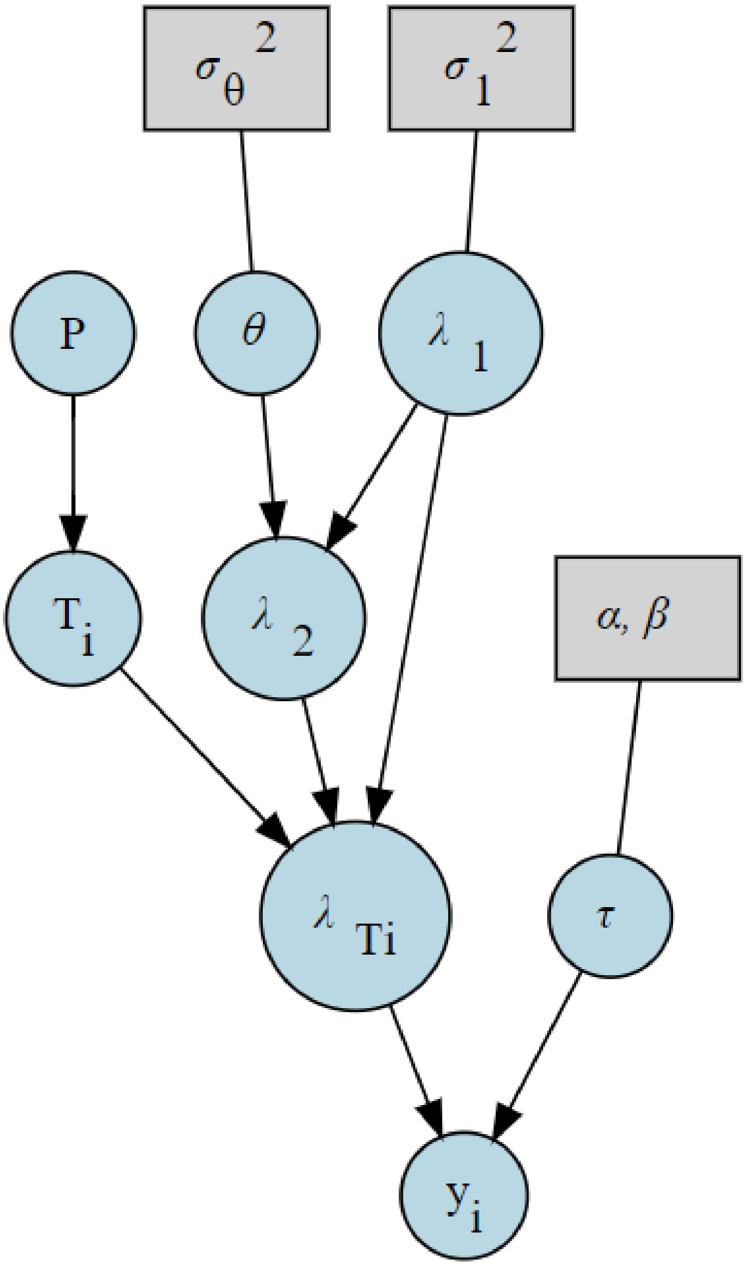

FIGURE 1 – DAG correspondante

A2 Résultat

Parameter	Mean	SD	MC_error	2.5%	Median	97.5%	Start	Sample
P[1]	0.6018	0.0826	0.000870	0.4392	0.6036	0.7586	1001	10000
P[2]	0.3982	0.0826	0.000870	0.2414	0.3964	0.5608	1001	10000
lambda[1]	536.7021	0.9058	0.009548	534.8288	536.6955	538.4622	1001	10000
lambda[2]	548.9633	1.1541	0.012165	546.6067	548.9466	551.3026	1001	10000
sigma	3.7330	0.5412	0.005704	2.9152	3.6545	5.0253	1001	10000

FIGURE 2 – Estimations postérieures après 10 000 itérations

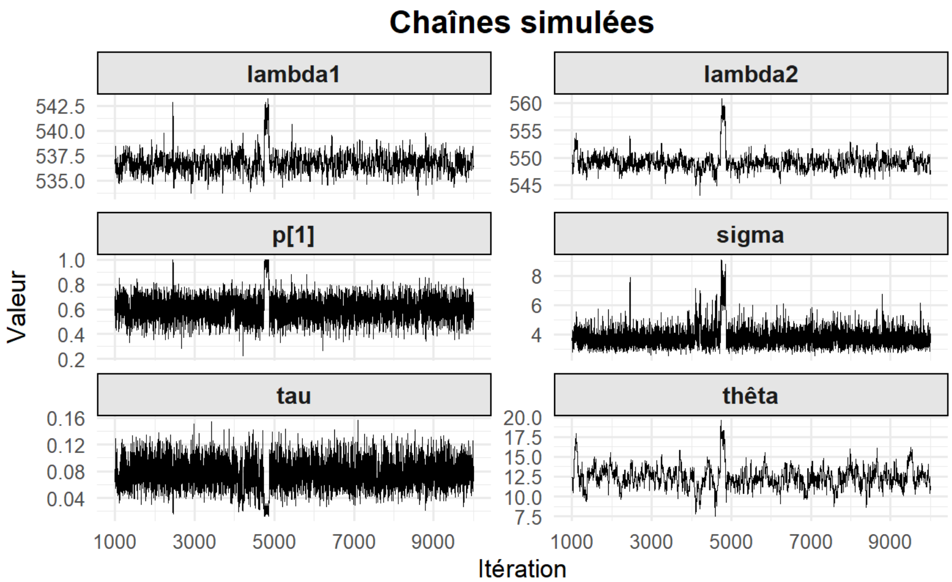

FIGURE 3 – Chaînes simulées

A3 code

```
```{r}
N <- 48
y <- c(529, 530, 532, 533.1, 533.4, 533.6, 533.7, 534.1, 534.8, 535.3,
       535.4, 535.9, 536.1, 536.3, 536.4, 536.6, 537, 537.4, 537.5, 538.3,
       538.5, 538.6, 539.4, 539.6, 540.4, 540.8, 542, 542.8, 543, 543.5,
       543.8, 543.9, 545.3, 546.2, 548.8, 548.7, 548.9, 549, 549.4, 549.9,
       550.6, 551.2, 551.4, 551.5, 551.6, 552.8, 552.9, 553.2)

alpha <- c(1,1)    # Dirichlet(1,1)

##MCMC
n_iter <- 10000
burn_in <- 1000
thin <- 1
set.seed(123)

```
```

```
## initiation
lambda1 <- 535
theta   <- 5
tau     <- 1/10   # sigmasq = 10 donc tau = 0.1
p       <- c(0.5, 0.5)

T_ <- sample(1:2, N, replace=TRUE)

##MCMC sampling
samples_lambda1 <- numeric(n_iter)
samples_theta   <- numeric(n_iter)
samples_tau     <- numeric(n_iter)
samples_p1      <- numeric(n_iter)


```
```

```
```{r}
sigma_lambda <- 0.5
sigma_theta  <- 0.2

accept_count <- 0
adjust_interval <- 100

for(iter in 1:n_iter) {

  ## update T[i]
  lam2 <- lambda1 + theta
  for(i in 1:N) {
    prob1 <- p[1] * dnorm(y[i], lambda1, sqrt(1/tau))
    prob2 <- p[2] * dnorm(y[i], lam2, sqrt(1/tau))
    T_[i] <- sample(1:2, 1, prob=c(prob1, prob2))
  }

  ## update P
  n1 <- sum(T_ == 1)
  p1_new <- rbeta(1, 1 + n1, 1 + (N - n1))
  p <- c(p1_new, 1 - p1_new)

  ## update Metropolis (lambda1, theta)
  lam1_prop <- lambda1 + rnorm(1, mean=0, sd=sigma_lambda)
  theta_prop <- abs(theta + rnorm(1, mean=0, sd=sigma_theta))

  log_accept_ratio <- sum(dnorm(y[T_==1], mean=lam1_prop, sd=sqrt(1/tau), log=TRUE)) +
    sum(dnorm(y[T_==2], mean=lam1_prop + theta_prop, sd=sqrt(1/tau), log=
```

```
        TRUE)) -
    sum(dnorm(y[T_==1], mean=lambda1, sd=sqrt(1/tau), log=TRUE)) -
    sum(dnorm(y[T_==2], mean=lambda1 + theta, sd=sqrt(1/tau), log=TRUE))

  if(log(runif(1)) < log_accept_ratio) {
    lambda1 <- lam1_prop
    theta   <- theta_prop
    accept_count <- accept_count + 1
  }

  ## update tau
  ssq <- sum((y[T_==1] - lambda1)^2) + sum((y[T_==2] - (lambda1 + theta))^2)
  tau <- rgamma(1, shape=0.001 + N/2, rate=0.001 + 0.5 * ssq)

  ## save the samples
  samples_lambda1[iter] <- lambda1
  samples_theta[iter]   <- theta
  samples_tau[iter]     <- tau
  samples_p1[iter]      <- p[1]

  if(iter %% adjust_interval == 0) {
    accept_rate <- accept_count / adjust_interval  # calculate accept rate
    if(accept_rate < 0.2) {
      sigma_lambda <- sigma_lambda * 0.9
      sigma_theta  <- sigma_theta * 0.9
    } else if(accept_rate > 0.5) {
      sigma_lambda <- sigma_lambda * 1.1
      sigma_theta  <- sigma_theta * 1.1
    }
    accept_count <- 0
  }
}

## analyse the result
burned_samples <- (burn_in+1):n_iter
cat("Posterior Mean Estimates:\n")
cat("lambda1 =", mean(samples_lambda1[burned_samples]), "\n")
cat("theta   =", mean(samples_theta[burned_samples]), "\n")
cat("tau     =", mean(samples_tau[burned_samples]), "\n")
cat("p1      =", mean(samples_p1[burned_samples]), "\n")
```

```{r}

posterior_summary <- function(samples, name) {
```

````r
  mean_val <- mean(samples)
  sd_val <- sd(samples)
  mc_error <- sd_val / sqrt(length(samples))
  quantiles <- quantile(samples, probs = c(0.025, 0.5, 0.975))

  cat(sprintf("%-10s %8.4f %8.4f %10.6f %8.4f %8.4f %8.4f %6d %6d\n",
              name, mean_val, sd_val, mc_error,
              quantiles[1], quantiles[2], quantiles[3], burn_in+1, n_iter))
}

cat(sprintf("%-10s %8s %8s %10s %8s %8s %8s %6s %6s\n",
            "Parameter", "Mean", "SD", "MC_error", "2.5%", "Median", "97.5%", "Start", "Sample"))
cat(rep("-", 70), "\n")

posterior_summary(samples_p1[burned_samples], "P[1]")
posterior_summary(1 - samples_p1[burned_samples], "P[2]")
posterior_summary(samples_lambda1[burned_samples], "lambda[1]")
posterior_summary((samples_lambda1 + samples_theta)[burned_samples], "lambda[2]")
posterior_summary((1 / sqrt(samples_tau))[burned_samples], "sigma")  # sigma = 1/sqrt(tau)
```
```{r}
library(ggplot2)
library(tidyr)
library(dplyr)

df_trace <- data.frame(
  Itération = burned_samples,
  lambda1 = samples_lambda1[burned_samples],
  theta   = samples_theta[burned_samples],
  tau     = samples_tau[burned_samples],
  p1      = samples_p1[burned_samples],
  lambda2 = samples_lambda1[burned_samples] + samples_theta[burned_samples],
  sigma   = 1 / sqrt(samples_tau[burned_samples])
)

df_long <- df_trace %>%
  pivot_longer(-Itération, names_to = "Paramètre", values_to = "Valeur") %>%
  mutate(Paramètre = recode(Paramètre,
    lambda1 = "lambda1",
    lambda2 = "lambda2",
    p1 = "p[1]",
````

```
    sigma = "sigma",
    tau = "tau",
    theta = "thêta"
  ))

ggplot(df_long, aes(x = Itération, y = Valeur)) +
  geom_line(size = 0.3) +
  facet_wrap(~Paramètre, scales = "free_y", ncol = 2) +
  theme_minimal(base_size = 13) +
  labs(title = "Chaînes␣simulées",
       y = "Valeur", x = "Itération") +
  scale_x_continuous(breaks = seq(1000, 10000, by = 2000)) +
  theme(
    strip.background = element_rect(fill = "grey90"),
    strip.text = element_text(face = "bold", size = 12),
    axis.text.x = element_text(size = 10),
    axis.text.y = element_text(size = 10),
    plot.title = element_text(face = "bold", hjust = 0.5, size = 16)
  )
```
```
```