Practical Work - EM Algorithm

Adib Habbou - Alae Khidour

11-12-2022

Calcul théorique

On considère un mélange de K lois de Poisson :

$$f(x) = \sum_{k=1}^{K} \pi_k \ f_k(x) = \sum_{k=1}^{K} \pi_k \ \frac{\lambda_k^x}{x!} \ e^{-\lambda k}$$

Les valeurs observées sont notées :

$$X = (x_1, x_2, ..., x_n)$$

Pour savoir de quelle composante vient le x_i on dispose de :

$$Z = (z_1, z_2, ..., z_n)$$

On notera nos paramètres :

$$\theta^{(q)} = \{\pi_1^{(q)}, ..., \pi_K^{(q)}, \lambda_1^{(q)}, ..., \lambda_K{}^{(q)}\}$$

On initialise l'algorithme avec :

$$\begin{cases} \pi_k^{(0)} = \frac{1}{k} \quad \forall k \in [\![1; K]\!] \\ \lambda_k^{(0)} = x_i \quad \text{tel que} \quad i \sim \mathcal{U}_{[\![1;n]\!]} \quad \forall k \in [\![1; K]\!] \end{cases}$$

On calcule tout d'abord la vraisemblance :

$$\mathcal{P}(X, Z; \theta) = \prod_{i=1}^{n} \mathcal{P}(x_i, z_i; \theta)$$

$$\mathcal{P}(X, Z; \theta) = \prod_{i=1}^{n} \prod_{k=1}^{K} \mathcal{P}(x_i, z_i; \theta) 1_{\{z_i = k\}}$$

On calcule ensuite la log-vraisemblance :

$$\begin{aligned}
\log(\mathcal{P}(X,Z;\theta)) &= \log(\prod_{i=1}^{n}\prod_{k=1}^{K}\mathcal{P}(x_i,z_i;\theta))1_{\{z_i=k\}} \\
&= \sum_{i=1}^{n}\sum_{k=1}^{K}\log(\mathcal{P}(x_i,z_i;\theta))1_{\{z_i=k\}} \\
&= \sum_{i=1}^{n}\sum_{k=1}^{K}\log(\mathcal{P}(z_i=k,\theta)\mathcal{P}(x_i|z_i=k,\theta))1_{\{z_i=k\}} \\
\log(\mathcal{P}(X,Z;\theta)) &= \sum_{i=1}^{n}\sum_{k=1}^{K}\log(\pi_k f_k(x_i;\theta))1_{\{z_i=k\}}
\end{aligned}$$

On peut donc maintenant calculer la quantité $Q(\theta,\theta^{(q)})$:

$$\begin{aligned}
Q(\theta,\theta^{(q)}) &= E_{Z|X;\theta^{(q)}}[\log\mathcal{P}(X,Z;\theta^{(q)})] \\
&= \sum_{i=1}^{n}\sum_{k=1}^{K}E_{Z|X;\theta^{(q)}}[1_{\{z_i=k\}}]\log(\pi_k f_k(x_i;\theta)) \\
&= \sum_{i=1}^{n}\sum_{k=1}^{K}t_{ik}\log(\pi_k\ \frac{\lambda_k^{x_i}}{x_i!}\ e^{-\lambda k}) \\
Q(\theta,\theta^{(q)}) &= \sum_{i=1}^{n}\sum_{k=1}^{K}t_{ik}[\log(\pi_k)-\log(x_i!)+x_i\log(\lambda_k)-\lambda_k]
\end{aligned}$$

On calcule enfin les estimateurs de nos paramètres :

$$\frac{\partial Q(\theta,\theta^{(q)}}{\partial\lambda_k} = \sum_{i=1}^{n}\sum_{k=1}^{K}t_{ik}^{(q)}(\frac{x_i}{\lambda_k-1}) = 0$$

$$\sum_{i=1}^{n}t_{ik}^{(q)}(\frac{x_i}{\lambda_k}-1) = 0$$

$$\sum_{i=1}^{n}\frac{t_{ik}^{(q)}x_i}{\lambda_k} = \sum_{i=1}^{n}t_{ik}^{(q)}$$

On trouve finalement :

$$\boxed{\lambda_k^{(q+1)} = \frac{\sum_{i=1}^{n}t_{ik}^{(q)}x_i}{\sum_{i=1}^{n}t_{ik}^{(q)}}}$$

On peut également calculer la proportion de notre k-ème loi de Poisson en posant :

$$\mathcal{L}(\theta, \alpha) = Q(\theta, \theta^{(q)} = \alpha(\sum_{k=1}^{K} \pi_k - 1)$$

Parce qu'on sait que :

$$\sum_{k=1}^{K} \pi_k = 1 \Longleftrightarrow \sum_{k=1}^{K} \pi_k - 1 = 0$$

Il suffit donc d'annuler la dérivée :

$$\frac{\partial \mathcal{L}(\theta, \alpha)}{\partial \theta} = 0$$

$$\frac{\partial}{\partial \pi_k}(\sum_{i=1}^{n} t_{ik}^{(q)} \log(\pi_k) + \alpha(\sum_{k=1}^{K} \pi_k - 1) = 0$$

$$\sum_{i=1}^{n} t_{ik}^{(q)} \frac{1}{\pi_k} - \alpha = 0$$

$$\pi_k^{(q+1)} = \frac{1}{\alpha} \sum_{i=1}^{n} t_{ik}^{(q)}$$

$$\alpha = \sum_{i=1}^{n} \sum_{k=1}^{K} t_{ik}^{(q)} = \sum_{i=1}^{n} 1 = n$$

Par conséquent :

$$\boxed{\pi_k^{(q+1)} = \frac{1}{n} \sum_{i=1}^{n} t_{ik}^{(q)}}$$

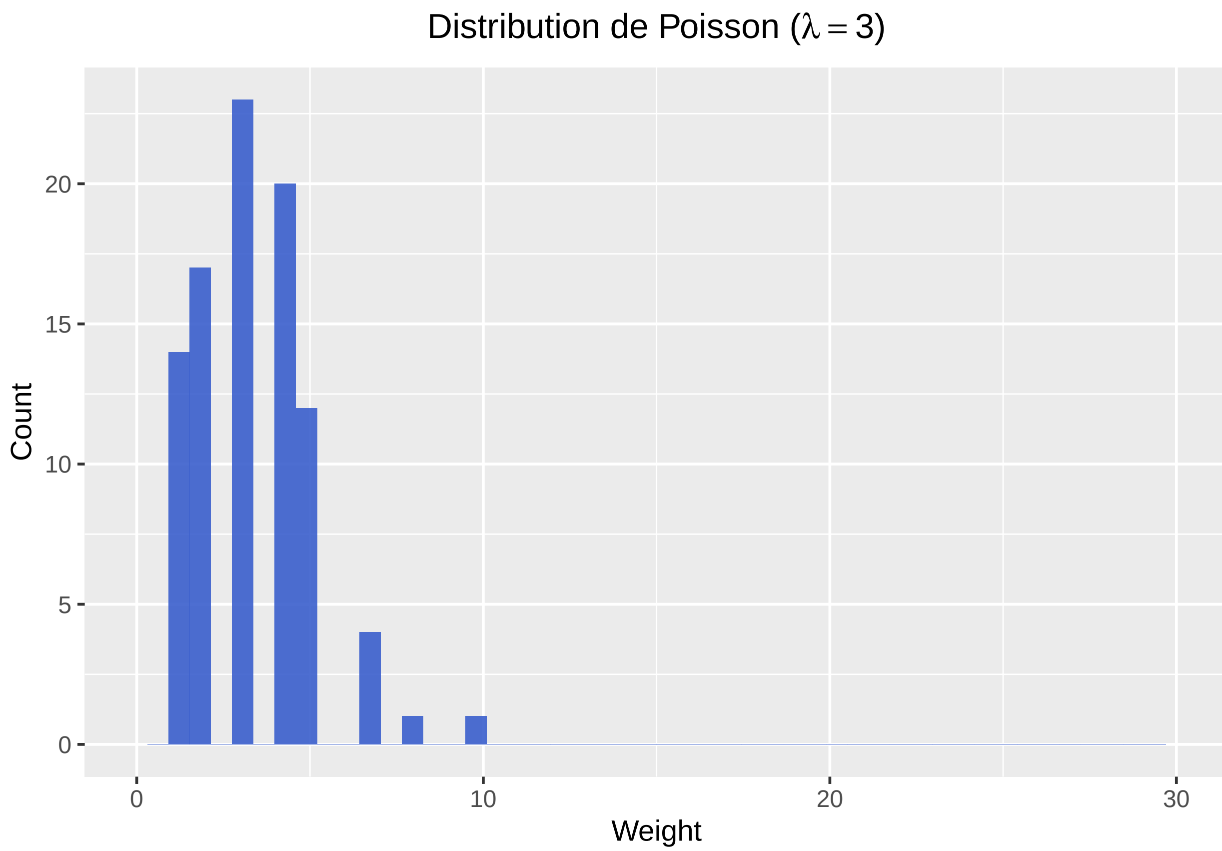
Simulation

Question 1 :

On simule un échantillon de 100 observation d'une loi de Poisson de paramètre $\lambda = 3$:

```
poisson100 <- rpois(n = 100, lambda = 3)
```

```
ggplot(data = data.frame(p = poisson100), mapping = aes(x = p)) +
  geom_histogram(bins = 50, fill = "royalblue3", alpha = 0.9) +
  labs(title = TeX("Distribution de Poisson ($\\lambda = 3$)"), x = "Weight", y = "Count") +
  theme(plot.title = element_text(hjust = 0.5)) + xlim(0, 30)
```

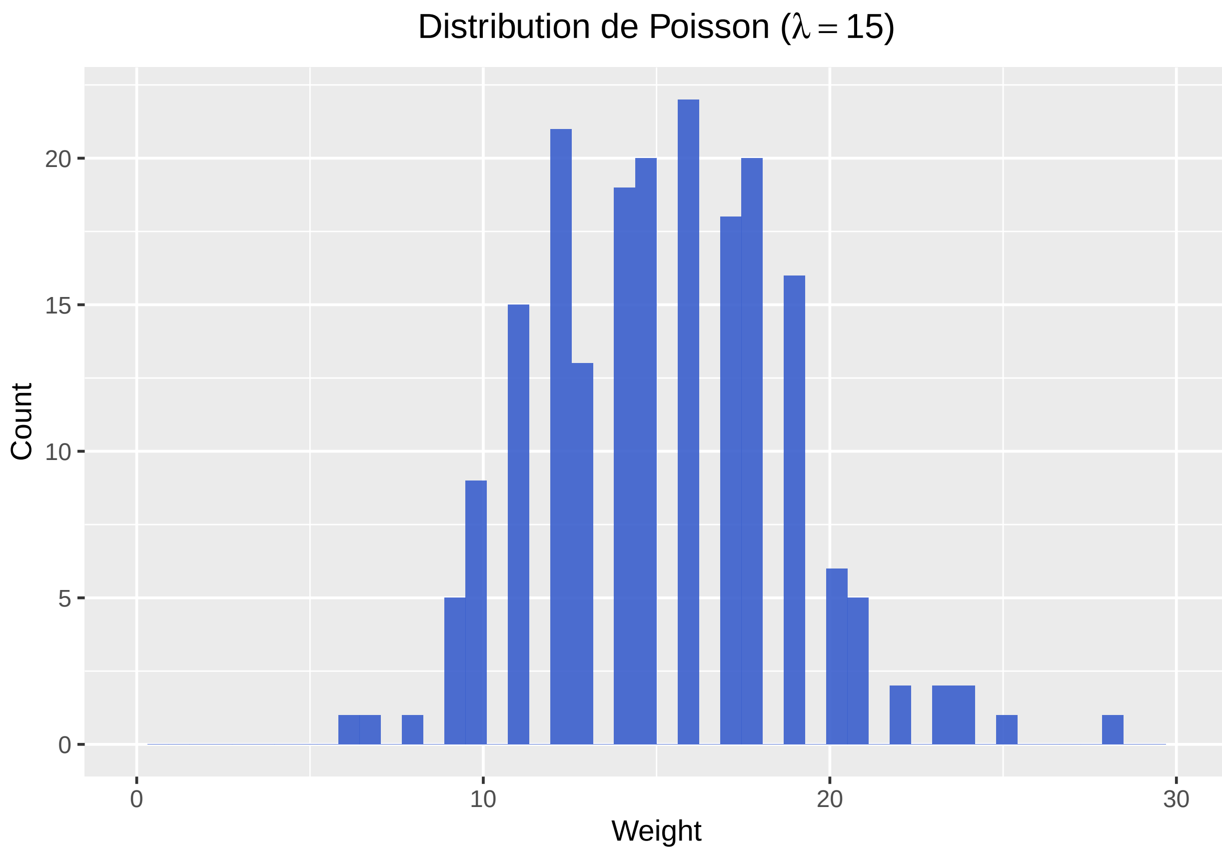


Question 2 :

On simule un échantillon de 200 observation d'une loi de Poisson de paramètre $\lambda = 15$:

```
poisson200 <- rpois(n = 200, lambda = 15)
```

```
ggplot(data = data.frame(p = poisson200), mapping = aes(x = p)) +
  geom_histogram(bins = 50, fill = "royalblue3", alpha = 0.9) +
  labs(title = TeX("Distribution de Poisson ($\\lambda = 15$)"),
       x = "Weight", y = "Count") +
  theme(plot.title = element_text(hjust = 0.5)) + xlim(0, 30)
```



Question 3 :

On crée un vecteur de 300 valeurs entières composé de 100 fois 1 et 200 fois 2 :

```r
vect <- c(rep(x = 1, times = 100), rep(x = 2, times = 200))
```

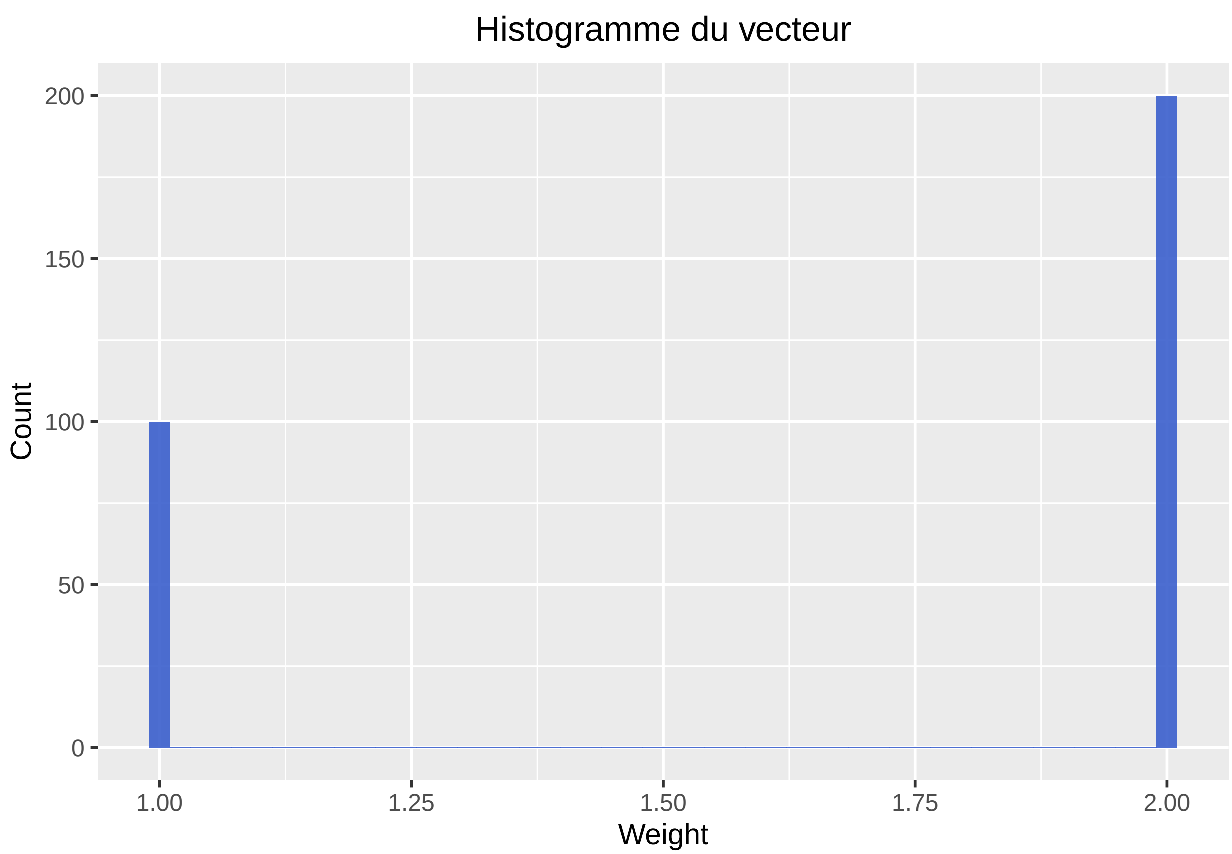
```r
vect[1:100]
```

```
  [1] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
 [38] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
 [75] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
```

```r
vect[101:300]
```

```
  [1] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
 [38] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
 [75] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
[112] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
[149] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
[186] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
```

```r
ggplot(data = data.frame(v = vect), mapping = aes(x = v)) +
  geom_histogram(bins = 50, fill = "royalblue3", alpha = 0.9) +
  labs(title = "Histogramme du vecteur", x = "Weight", y = "Count") +
  theme(plot.title = element_text(hjust = 0.5))
```



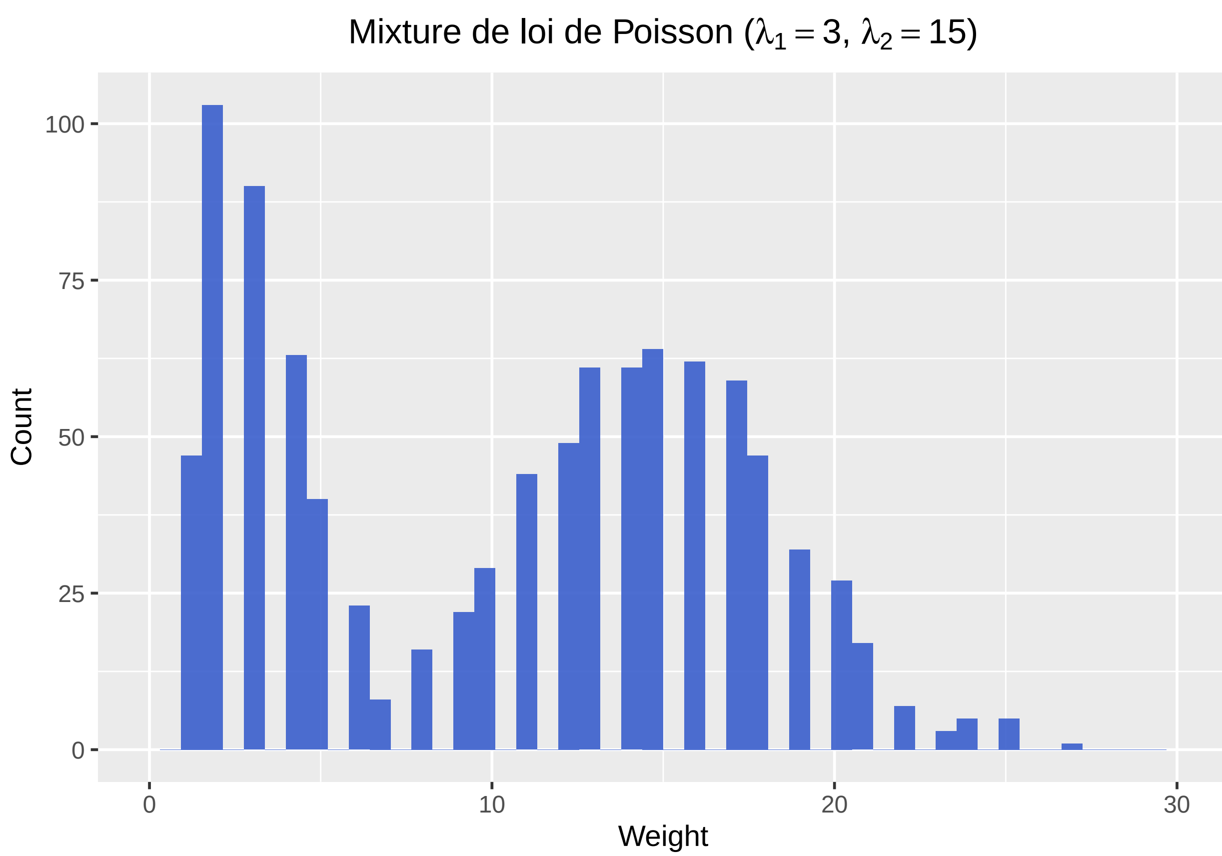
Question 4 :

On simule une mixture de lois de poissons à deux composantes $\lambda_1 = 3$ et $\lambda_2 = 15$ avec les proportions $\pi_1 = 0.4$ et $\pi_2 = 0.6$.

```
n <- 1000
lambda <- c(3, 15)
pi <- c(0.4, 0.6)
```

```
poisson <- rep(NA, n)
vect <- sample(1:2, size = n, replace = TRUE, prob = pi)
for (k in 1:n)
{
  poisson[k] <- rpois(1, lambda = lambda[vect[k]])
}
```

```
ggplot(data = data.frame(p = poisson), mapping = aes(x = p)) +
  geom_histogram(bins = 50, fill = "royalblue3", alpha = 0.9) +
  labs(title = TeX("Mixture de loi de Poisson ($\\lambda_1 = 3$, $\\lambda_2 = 15$)"),
       x = "Weight", y = "Count") +
  theme(plot.title = element_text(hjust = 0.5)) + xlim(0, 30)
```



Algorithme EM

Question 1-2-3 :

```
algo_EM <- function(data, K)
{
  ## INITIALISATION

  # on stocke la taille de nos données
  n <- length(data)

  # on initialise pi on utilisant une distribution uniforme
  pi <- runif(K, 0, 1)

  # on initialise lambda on utilisant une distribution uniforme
  lambda <- runif(K, 0, 20)

  # on initialise la matrice T
  T <- matrix(0, nrow = n, ncol = K)

  # on initialise la matrice normalisé de T
  norm_T <- matrix(0, nrow = n, ncol = K)

  # on initialise la vraisemblance pour une itération
  likelihood <- rep(0, times = n)

  # on initialise la log-vraisemblance pour une itération
  log_likelihood <- rep(0, times = n)

  # on initialise la somme les log-vraisemblances
  log_likelihood_sum <- 0

  # limite des critères de convergence
  eps <- 10^(-6)

  ## CALCUL

  while (TRUE)
  {

    # sauvegarde du vecteur pi avant l'itération
    pi_old <- pi

    # sauvegarde du vecteur lambda avant l'itération
    lambda_old <- lambda

    ## Etape E

    # calcul de la matrice T

    for (i in 1:n)
    {
```

```r
      for (k in 1:K)
      {
        # calcul de chaque élément de la matrice T
        T[i, k] <- dpois(as.integer(data[i]), lambda[k]) * pi[k]
      }
    }

    for(i in 1:n)
    {
      # normalisation de la matrice T
      norm_T[i,] <- T[i,] / sum(T[i,])

      # calcul de la vraisemblance
      likelihood[i] <- sum(T[i,])
    }

    # calcul de la log-vraisemblance
    log_likelihood <- log(likelihood, base = exp(1))

    # calcul de la somme des log-vraisemblances
    log_likelihood_sum <- sum(log_likelihood)

    ## Etape M

    # calcul des proportions pi
    pi <- colSums(norm_T) / n

    # calcul des paramètres lambda
    lambda <- colSums(norm_T * as.integer(data)) / colSums(norm_T)

    ## CONVERGENCE

    # critères de convergence sur le vecteur pi
    norm2_pi <- Norm(pi_old - pi, 2) / Norm(pi_old, 2)

    # critères de convergence sur le vecteur lambda
    norm2_lambda <- Norm(lambda_old - lambda, 2) / Norm(lambda_old, 2)

    # condition d'arrêt
    if (norm2_pi < eps && norm2_lambda < eps) break

  }

  ## RESULTAT

  return(list(pi = pi, # Distribution de probabilité de Poisson
              lambda = lambda, # Paramètres de Poisson
              log_likelihood_sum = log_likelihood_sum # Somme des log-vraisemblances
              ))
}
```

Question 4 :

Test sur une distrubition

```
poisson_test <- rpois(n = 100, lambda = 5)
algo_EM(data = poisson_test, K = 1)
```

```
$pi
[1] 1

$lambda
[1] 5.07

$log_likelihood_sum
[1] -224.2798
```

L'algorithme retrouve bien une valeur proche de $\lambda = 5$.

Test sur une mixture de deux distrubitions

On applique l'algorithme EM sur la mixture de lois de poisson simulée :

```
algo_EM(data = poisson, K = 2)
```

```
$pi
[1] 0.6158513 0.3841487

$lambda
[1] 14.905561  3.010269

$log_likelihood_sum
[1] -3031.656
```

L'algorithme retrouve bien les paramètres $\lambda_1 = 3$ et $\lambda_2 = 15$ et les proportions $\pi_1 = 0.4$ et $\pi_2 = 0.6$.

Test sur une mixture de trois distrubitions

```
poisson_triple <- rmixpois(n = 1000, lambda = c(5, 10, 15), alpha = c(0.2, 0.3, 0.5))
algo_EM(data = poisson_triple, K = 3)
```

```
$pi
[1] 0.4671947 0.1331097 0.3996956

$lambda
[1] 15.223364  4.134760  9.548137

$log_likelihood_sum
[1] -3041.518
```

L'algorithme retrouve bien les paramètres : $\lambda_1 = 5$, $\lambda_2 = 10$, $\lambda_2 = 15$ et $\pi_1 = 0.2$, $\pi_2 = 0.3$, $\pi_3 = 0.5$.