```r
# modèles de mélange en 1D , K nombre de composants du modèle
gmixtmono <- function(donnees, K, param=NULL, fCEM=FALSE)
{
    # modèle de mélange gaussien unidimensionnel
    if (is.vector(donnees))
    {
        # initialisation des varibles locales
        n <- length(donnees)
        t_ik <- matrix(NA, n, K)
        c_ik <- matrix(NA, n, K)
        epsilon <- 10^(-12)
        nbiteration <- 0
        nL <- epsilon
        L <- 0

        # initialisation arbitraire des paramètres
        if (is.null(param))
        {
            for( k in 1:K )
            {
                pi_k <- 1/K
                mu_k <- mean( donnees[round((k-1)*n/K+1) : round(k*n/K)] )
                sigma_k <- var( donnees[round((k-1)*n/K+1) : round(k*n/K)] )

                param[[k]] <- list(p = pi_k, mu = mu_k, sigma = sigma_k)
            }
        }

        # tant que le point de convergence n'est pas atteint
        while( abs(nL - L) >= epsilon && nbiteration < 100 )
        {
            # Mise a jour des parametres
            nbiteration = nbiteration + 1
              print(nbiteration)
            L <- nL

            # étape E
            for( k in 1:K )
                t_ik[,k] <-param[[k]]$p*dnorm(donnees, param[[k]]$mu, param[[k]]$sigma)

            t_ik = t_ik / rowSums(t_ik)

            # étape C
            if( fCEM )
                for( i in 1:n )
                {
                        c_ik[i,] <- 0
                        c_ik[i,which.max(t_ik[i,])] <- 1
                }
            else
                c_ik <- t_ik

            # étape M
            for( k in 1:K )
            {
                pi_k <- 1/n * sum(c_ik[,k])
                mu_k <- ( t(c_ik[,k]) %*% donnees ) / sum(c_ik[,k])
                sigma_k <- sqrt( ( t(c_ik[,k]) %*% (donnees - mu_k)^2 ) / sum(c_ik[,k]) )

                param[[k]] <-list(p=pi_k, mu=as.numeric(mu_k), sigma=as.numeric(sigma_k))
            }

            # Calcul de la nouvelle vraissemblance
            fk <- matrix(0,n)
```