K-means algorithm

- Suppose that x_1, ..., x_n are n observations in R^d and we would like to find a way of dividing $\{x_1, \ldots, x_n\}$ into disjoint subsets S_1, ..., S_k so that

$$\sum_{\ell=1}^{k} \sum_{x \in S_\ell} \|x - c_\ell\|^2 \tag{1}$$

  is minimized, where $\|\cdot\|$ is the Euclidean norm and c_1, ..., c_k are the centers of S_1, ..., S_k respectively. That is, for $\ell \in \{1, \ldots, k\}$,

$$c_\ell = \frac{\sum_{x \in S_\ell} x}{n_\ell},$$

  where n_ℓ is the number of observations in S_ℓ. Here k is pre-specified.

- Basic algorithm. The algorithm is iterative.
  - Suppose that at the t-th iteration, $c_1^{(t)}$, ..., $c_k^{(t)}$ are estimates for the c_1, ..., c_k in (1). Then an observation is assigned to the i-th group if among $c_1^{(t)}$, ..., $c_k^{(t)}$, $c_i^{(t)}$ is the closest to the observation. After every observation has been assigned to a group, take $c_1^{(t+1)}$, ..., $c_k^{(t+1)}$ to be the new group centers.
  - Usually the above iteration process is stopped when

$$\|(c_1^{(t+1)}, \ldots, c_k^{(t+1)}) - (c_1^{(t)}, \ldots, c_k^{(t)})\| < \delta,$$

    where δ is a pre-specified positive number, or the process is stopped when the number of iterations has reached a pre-specified limit.

- Univariate case ($d = 1$).

  Example 1. Write a function `km` for running the univariate k-means algorithm. The input variables for `km` are `data`, `center.ini`, `delta` and `iter.max`, where
  - `data`: the vector of observations $(x_1, \ldots, x_n)$
  - `center.ini`: vector of initial estimates for the c_1, ..., c_k in (1)
  - `delta`: δ. The default value for δ is 10^{-6}.
  - `iter.max`: the max number of iterations allowed. The default value is 100.

  The function output includes the estimated group centers and a vector indicating group membership for the observations. Run the following lines to generate data `x`:

```
set.seed(1)
x <- c(rnorm(20, mean=10), rnorm(30, mean=-10), rnorm(20, mean=5))
group.true <- c(rep(1, 20), rep(2, 30), rep(3, 20))
```

Test the above function using `x` as the data and using `x[1], x[21], x[51]` as initial centers.

Sol. Define the function `km`:

```
km <- function(data, center.ini, delta=10^(-6), iter.max=100){
  k <- length(center.ini)
  n <- length(data)
  center.t <- center.ini
  for (tt in 1:iter.max){
    group <- rep(0, n)
    for (i in 1:n){
     group[i] <- which.min(abs(data[i] - center.t))[1]
    }
    center.tt <- center.t
    for (j in 1:k){ center.tt[j] <- mean(data[group==j]) }
    if (any(is.na(center.tt))) { return(list("NA appears", group)) }
    if ( sqrt(sum((center.tt-center.t)^2)) < delta) { break }
    center.t <- center.tt
  }
  ans <- list(group, center.tt)
  names(ans) <- c("group", "centers")
  return(ans)
}
```

Test the function

```
set.seed(1)
x <- c(rnorm(20, mean=10), rnorm(30, mean=-10), rnorm(20, mean=5))
group.true <- c(rep(1, 20), rep(2, 30), rep(3, 20))
res <- km(x, c(x[1], x[21], x[51]),delta=10^(-6), iter.max=100)
res
table(res$group, group.true)
ss <- 0
for (i in 1:3){
  ss <- ss + sum((x[res$group==i]-res$centers[i])^2)
}
ss
#Run the function with one iteration gives larger sum of squares
res <- km(x, c(x[1], x[21], x[51]),delta=10^(-6), iter.max=1)
ss <- 0
for (i in 1:3){
```

```
  ss <- ss + sum((x[res$group==i]-res$centers[i])^2)
}
ss

#Run the function with bad initial values
res <- km(x, c(x[1], x[2], x[51]),delta=10^(-6), iter.max=100)
res

#Run the function with bad initial values
set.seed(2)
group.test <- sample(1:3, length(x), replace=T)
center.test <- rep(0,3)
for (i in 1:3){
  center.test[i] <- mean(x[group.test==i])
}
center.test
res <- km(x, center.test,delta=10^(-6), iter.max=100)
res
```

- Practice problems.

  1. Modify the `km` function so that it can compute the k-means algorithm when the observations are in R^d (`data` is a data matrix). Name the modified function `km1`. Test the function by running the following lines:

     ```
     set.seed(1)
     x <- c(rnorm(20, mean=10), rnorm(30, mean=-10), rnorm(20, mean=5))
     x <- cbind(x, c(rnorm(20, mean=10), rnorm(30, mean=-10), rnorm(20, mean=5)))
     res <- km1(x,  x[c(1,21,51),] ,delta=10^(-6), iter.max=100)
     res
     ```

  2. Modify the `km` function so that it can continue to run the k-means algorithm with fewer centers when some centers correspond to empty groups. Name the modified function `km2`. Test the function by running the following lines:

     ```
     set.seed(1)
     x <- c(rnorm(20, mean=10), rnorm(30, mean=-10), rnorm(20, mean=5))
     set.seed(2)
     group.test <- sample(1:3, length(x), replace=T)
     center.test <- rep(0,3)
     for (i in 1:3){
       center.test[i] <- mean(x[group.test==i])
     }
     center.test
     res <- km2(x, center.test,delta=10^(-6), iter.max=100)
     res
     ```