03_EM_iteration

Frances Lin

11/24/2020

Description

This .Rmd file iterates between the E-step and M-step untill convergence using log likelihood at each iteration. All related results can be found in the results folder.

Load packages

```
library(here)
library(tidyverse)
library(tibble)
library(stats)
library(ggplot2)
library(gridExtra)
devtools::load_all() # load functions e_step() and m_step()
```

Load data

```
data <- readRDS(here("results", "data.rds"))
head(data)
```

```
## # A tibble: 6 x 2
##   component  value
##   <chr>      <dbl>
## 1 A         -1.21
## 2 A          0.277
## 3 A          1.08
## 4 A         -2.35
## 5 A          0.429
## 6 A          0.506
```

Load initial estimates from K-means clustering and rename to df

```
df_summary_kmeans <- readRDS(here("results", "df_summary_kmeans.rds"))
df <- df_summary_kmeans
df
```

```
## # A tibble: 2 x 6
##   cluster   mean   var    sd  size    pi
##     <int>  <dbl> <dbl> <dbl> <int> <dbl>
## 1       1  2.10  0.737 0.859   110  0.55
## 2       2 -0.471 0.445 0.667    90  0.45
```

Check to see if the e_step() and the m_step() function work

E-step: Calculate posterior probability (or soft labelling) using Bayes Rule and pass it to M-step & store log likelihood to check for convergence

```r
#?e_step
#good

#e_step(x = data$value, mu = df$mean, sd = df$sd, pi = df$pi)
#good
```

M-step: Replace hard labelling with posterior probability (or soft labelling) and optimize the parameters using MLE & return the final estimates if convergence happens

```r
E_step <- e_step(x = data$value, mu = df$mean, sd = df$sd, pi = df$pi)
#E_step
```

```r
#?m_step
#good

#m_step(x = data$value, posterior = E_step$posterior_prob)
#good
```

Putting it all together

Convergence: Iterate until convergence (i.e. change is minimal) using log likelihood

```r
# Set the #s of iterations
iterations <- 50

# Iterate between EM step untill convergence
for(i in 1:iterations){
  if (i == 1){
  # Initialization
  # Pass the initial estimates as a result of K-means
  e_out <- e_step(x = data$value, mu = df$mean, sd = df$sd, pi = df$pi)
  m_out <- m_step(x = data$value, posterior = e_out$posterior_prob)

  # Set to current log likelihood
  current_log_likelihood <- e_out$log_likelihood

  # Store log likelihood vector for plotting
  log_likelihood <- e_out$log_likelihood

  } else {
  # Repeat E and M steps until convergence
  # Pass the estimates as a result of the 1st (and current) EM iteration
  e_out <- e_step(x = data$value, mu = m_out$mu, sd = m_out$sd, pi = m_out$pi)
  m_out <- m_step(x = data$value, posterior = e_out$posterior_prob)

  # Incrementally store log likelihood vector for plotting
  log_likelihood <- c(log_likelihood, current_log_likelihood)

  # Check for convergence
  # Compare current log likelihood to current + 1 log likelihood
  check <- abs(current_log_likelihood - e_out$log_likelihood)

  if(check < 1e-3){
    # Converge
    break
  } else {
    # Do not converge
    # Reset current + 1 to current and repeat E and M steps
    current_log_likelihood <- e_out$log_likelihood
  }
  }
}

# Return log likelihood vector for plotting
log_likelihood
```

```
##  [1] -354.7605 -354.7605 -353.1962 -352.7744 -352.5211 -352.3219 -352.1559
##  [8] -352.0171 -351.9019 -351.8073 -351.7301 -351.6675 -351.6169 -351.5761
## [15] -351.5433 -351.5169 -351.4957 -351.4786 -351.4648 -351.4537 -351.4447
## [22] -351.4375 -351.4316 -351.4268 -351.4230 -351.4199 -351.4173 -351.4152
## [29] -351.4136 -351.4122 -351.4111
```

```
# Return current (or final) log likelihood element for checking
current_log_likelihood
```

```
## [1] -351.4111
```

```
# Return #s of iteractions for plotting
n_iterations <- length(log_likelihood)
n_iterations
```

```
## [1] 31
```

```
# Return convergence for checking
check
```

```
## [1] 0.0009184255
```

```
# Return for reporting
#e_out

# Return for reporting
m_out
```

```
## $mu
## [1]  1.7856173 -0.6610847
##
## $sd
## [1] 1.0783195 0.5909651
##
## $pi
## [1] 0.6553007 0.3446993
```

Estimates improve with N(0,1) and N(4, 1), as compared to N(0,1) and N(2,1)

```
# Combine EM results
result_1_parameters <- tibble(
  "mean" = c(m_out$mu[1], m_out$mu[2]),
  "sd" = c(m_out$sd[1], m_out$sd[2]),
  "pi" = c(m_out$pi[1], m_out$pi[2])
)
result_1_parameters
```

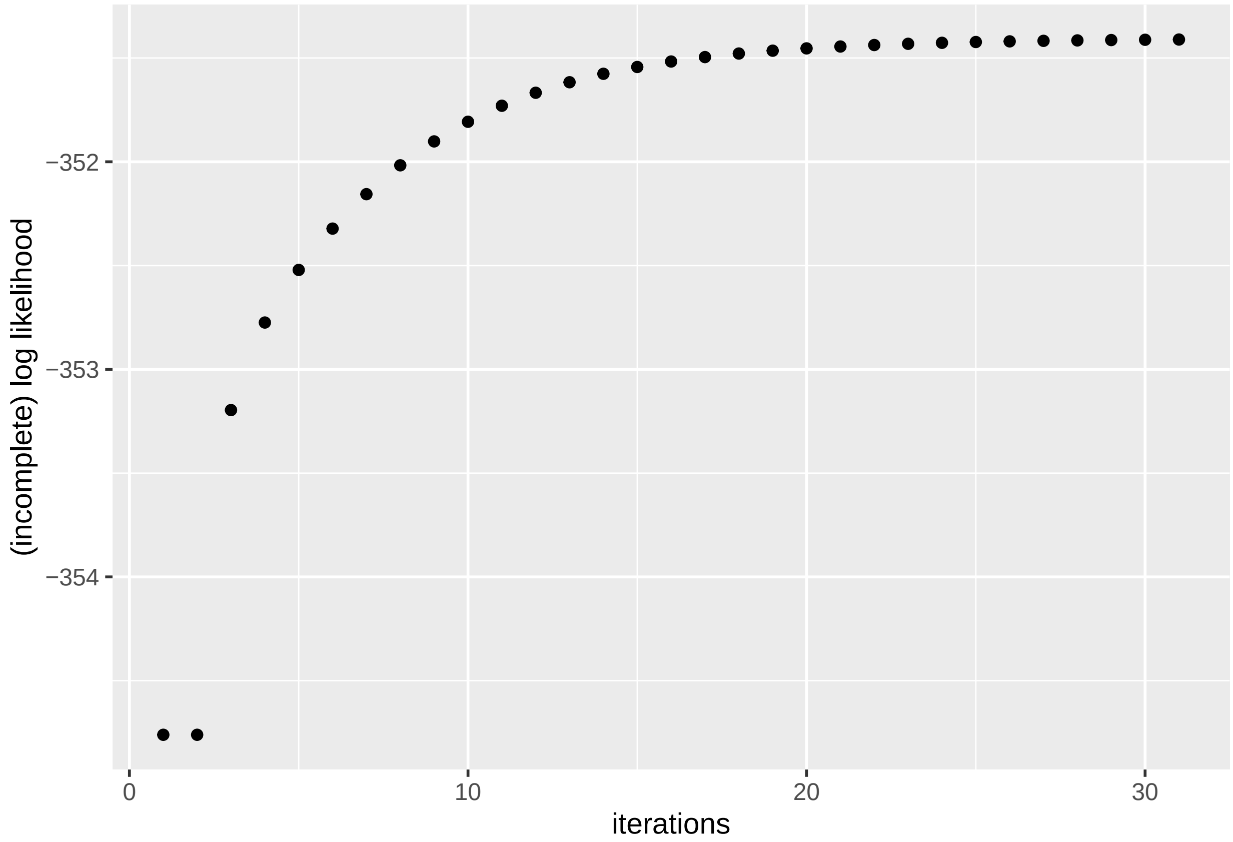
```
## # A tibble: 2 x 3
##     mean    sd    pi
##    <dbl> <dbl> <dbl>
## 1  1.79  1.08  0.655
## 2 -0.661 0.591 0.345
```

Converge now at 12th iteraction with N(0,1) and N(4, 1), as compared to 31st iteraction

```
# Combine EM results
result_2_max_log_like <- tibble(
  "max_log_likelihood" = current_log_likelihood,
  "#s of iteractions" = n_iterations
)
result_2_max_log_like
```

```
## # A tibble: 1 x 2
##   max_log_likelihood `#s of iteractions`
##                <dbl>               <int>
## 1              -351.                  31
```

```
# Plot (incomplete) log likelihood
result_3_plot_log_likelihood <- qplot(x = 1:n_iterations, y = log_likelihood,
                              xlab = "iterations",
                              ylab = "(incomplete) log likelihood")
result_3_plot_log_likelihood
```



If time permits, plot simulated data in histogram and overlay a density curve ### Save out results

```
write_rds(result_1_parameters, here("results", "result_1_parameters.rds"))
write_rds(result_2_max_log_like, here("results", "result_2_max_log_like.rds"))
write_rds(result_3_plot_log_likelihood, here("results", "result_3_plot_log_likelihood.rds"))
write_rds(e_out, here("results", "e_out.rds"))
write_rds(m_out, here("results", "m_out.rds"))

write_rds(log_likelihood, here("results", "log_likelihood.rds")) # fix reporting error
write_rds(current_log_likelihood, here("results", "current_log_likelihood.rds"))
write_rds(n_iterations, here("results", "n_iterations.rds"))
write_rds(check, here("results", "check.rds"))

#write_rds(plot_EM, here("results", "result_4_plot_EM.rds"))
```