PB HLTH C240D/STAT C245D: Assignment #3

Rachael Phillips

November 28, 2017

Maximum likelihood estimation of the ABO blood group allele frequencies using the EM algorithm

The ABO *blood groups* were the first to be discovered and are important in assuring safe blood transfusions (Cf. Landsteiner, 1930 Nobel Prize in Physiology and Medicine, `nobelprize.org/nobel_prizes/medicine/laureates/1930`) As indicated in Table 1, the ABO blood groups are characterized by the presence or absence of *antigens* on the surface of red blood cells and *antibodies* in serum. The ABO locus has three *alleles*, A, B, and O, leading to $3^2 = 9$ *phased genotypes*, $3 + 3 \times 2/2 = 6$ *unphased genotypes*, and four *phenotypes*, the blood groups **A**, **B**, **AB**, and **O**.

Let $\pi = (\pi_A, \pi_B, \pi_O)$ denote the ABO allele frequencies in a well-defined population of interest. Under the assumption of *Hardy-Weinberg equilibrium* (HWE), the maternal and paternal alleles are independent, i.e., genotype frequencies are products of allele frequencies. Let $Y = (Y_A, Y_B, Y_{AB}, Y_O)$ denote the ABO phenotype counts for a random sample of n individuals from the population of interest.

The objective of this assignment is to apply the EM algorithm to derive maximum likelihood estimates of the ABO allele frequencies for a dataset from the classical article of Clarke et al. (1959). Specifically, consider the following ABO phenotype counts for a sample of $n = 521$ duodenal ulcer patients (Clarke et al., 1959, Table III): $Y_A = 186$, $Y_B = 38$, $Y_{AB} = 13$, and $Y_O = 284$. For simplicity, you may assume that the $n = 521$ patients are a random sample from a well-defined population, with Hardy-Weinberg equilibrium at the ABO locus.

Table 1: ABO *blood groups*. Phenotypes, genotypes, and genotype frequencies under Hardy-Weinberg equilibrium.

Phenotype ABO blood group	**Antigens**	**Antibodies**	**Unphased genotype**	**Unphased genotype frequency (HWE)**
A	A	Anti-B	AA,AO	$\pi_A^2 + 2\pi_A\pi_O$
B	B	Anti-A	BB, BO	$\pi_B^2 + 2\pi_B\pi_O$
AB	A and B	Neither	AB	$2\pi_A\pi_B$
O	Neither	Anti-A and Anti-B	OO	π_O^2

Question 1. Conditional distribution of multinomial counts.

Suppose $X = (X_k : k = 1, ..., K)$ is a random variable with Multinomial$(n, \pi = (\pi_k : k = 1, ..., K))$ distribution, that is:

$$Pr(X = x) = \frac{n!}{\prod_{k=1}^K x_k!} \prod_{k=1}^K \pi_k^{x_k} \qquad (1)$$

where $x = (x_k : k = 1, ..., K) \in \mathbb{N}^K$, with $\sum_k x_k = n$ and $\pi = (\pi_k : k = 1, ...K) \in [0, 1]^K$, with $\sum_k \pi_k = 1$.

Derive the conditional distribution of X_k given $X_k + X_{k'}$, where $k, k' \in \{1, ..., K\}, k \neq k'$.

In particular, provide $E[X_k | X_k + X_{k'}]$, the conditional expected values of X_k given $X_k + X_k'$.

It follows from the given information that, for distinct k and k', that $(X_k, X_{k'}, \sum_{j\neq k,k'} X_j)$ has a multinomial distribution with parameters $(\pi_k, \pi_{k'}, \sum_{j\neq k,k'} \pi_j) = (\pi_k, \pi_{k'}, 1-\pi_k-\pi_{k'})$. So,

$$P(X_k = x_k \cap X_k + X_{k'} = y) = P(X_k = x_k \cap X_{k'} = y - x_k \cap \sum_{j\neq k,k'} X_j = n - y)$$

$$= \frac{n!}{x_k!(y-x_k)!(n-y)!}\pi_k^{x_k}\pi_{k'}^{y-x_k}(1-\pi_k-\pi_{k'})^{n-y} \quad .$$

On the other hand, $X_k + X_{k'}$ has binomial distribution with parameters n and $\pi_k + \pi_{k'}$, so

$$P(X_k + X_{k'} = y) = \frac{n!}{y!(n-y)!}(\pi_k+\pi_{k'})^y(1-\pi_k-\pi_{k'})^{n-y} \quad .$$

Therefore,

$$P(X_k = x_k | X_k + X_{k'} = y) = \frac{P(X_k = x_k \cap X_k + X_{k'} = y)}{P(X_k + X_{k'} = y)}$$

$$= \frac{\frac{n!}{x_k!(y-x_k)!(n-y)!}\pi_k^{x_k}\pi_{k'}^{y-x_k}(1-\pi_k-\pi_{k'})^{n-y}}{\frac{n!}{y!(n-y)!}(\pi_k+\pi_{k'})^y(1-\pi_k-\pi_{k'})^{n-y}}$$

$$= \frac{y!}{x_k!(y-x_k)!}\Big(\frac{\pi_k}{\pi_k+\pi_{k'}}\Big)^{x_k}\Big(\frac{\pi_{k'}}{\pi_k+\pi_{k'}}\Big)^{y-x_k} \quad .$$

So. the conditional distribution of X_k given $X_k + X_{k'}$, is binomial with parameters $X_k + X_{k'}$ and $\frac{\pi_k}{\pi_k+\pi_{k'}}$. Thus,

$$E[X_k | X_k + X_{k'}] = (X_k + X_{k'})\frac{\pi_k}{\pi_k+\pi_{k'}} \quad .$$

Question 2. Log-likelehood surface for trinomial probabilities

Propose and implement in R graphical displays for a *log-likelehood surface* when the parameter of interest is a vector of trinomial probabilities.

Display and comment on the log-likelehood surface for the ABO blood group phenotype counts in the Clarke et al. (1959) dataset.

Hint. Use barycentric coordinates.

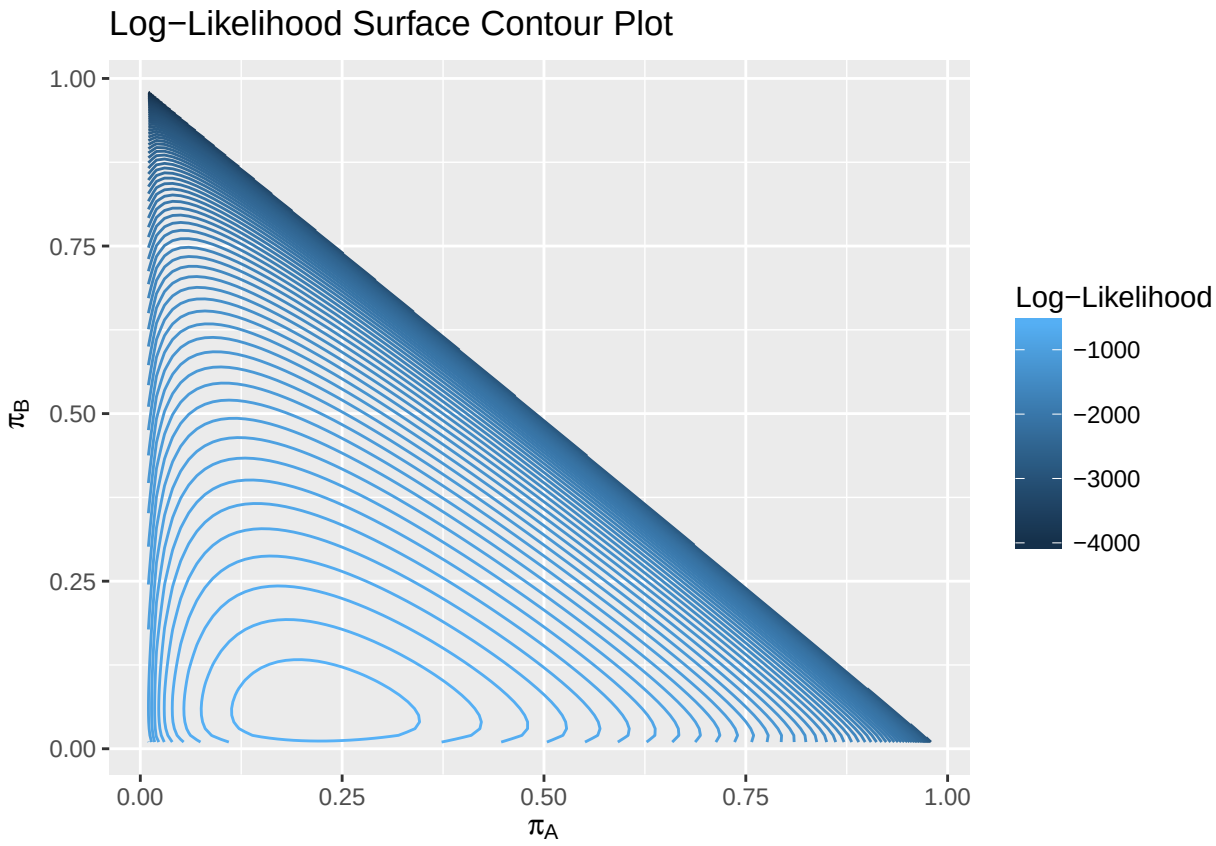
If we allow the trinomial probabilities to individually vary under the constraint that the sum of these three probabilities sums to one and, for each combination of values, we recalculate the log-likelihood, then we can plot the results as a goodness (or badness)-of-fit surface. The surface depicts the value of the log-likelihood for every combination of parameter values evaluated gives us an understanding of how the likelihood depends on one or more parameters.

Barycentric coordinates are triples of numbers corresponding to masses placed at the vertices of a reference triangle. These masses then determine a point, which is the geometric centroid of the three masses and is identified with coordinates. Thus barycentric coordinates are a method of introducing coordinates into a space.

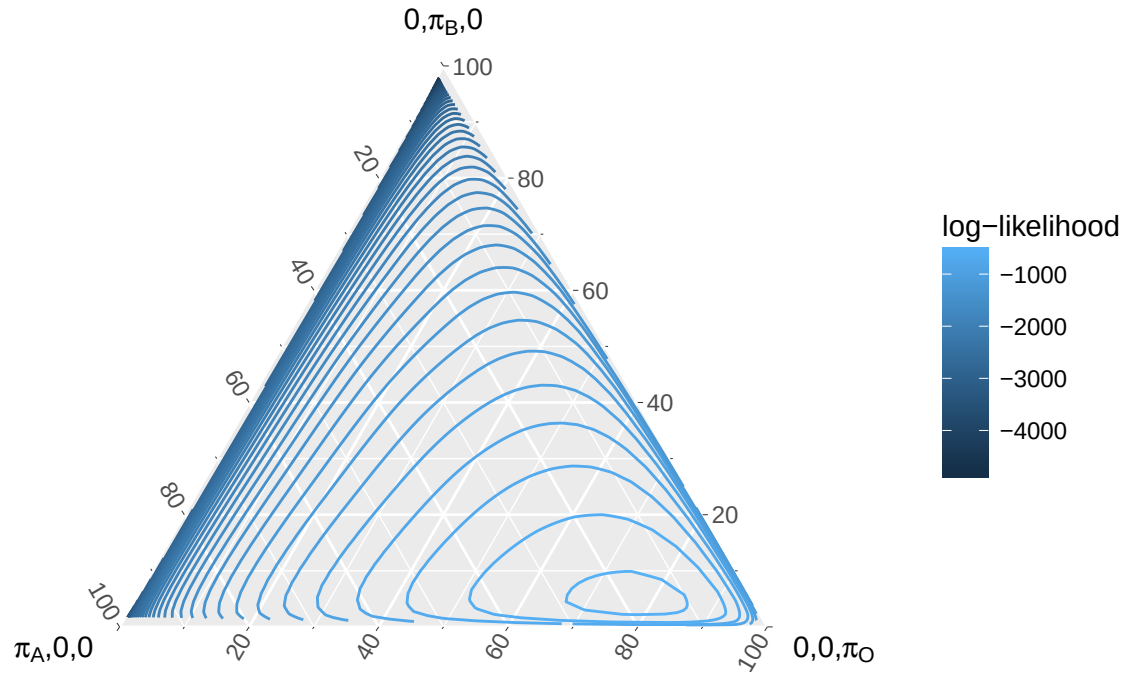
```r
# we simulate combinations of the trinomial probabilities
pi <- expand.grid(pi_A = seq(from = 0, to = 1, by = 0.01),
                  pi_B = seq(from = 0, to = 1, by = 0.01))
# setting up contraints assuring:
# 1. all three probabilities > 0
pi <- pi[pi$pi_A > 0
                 & pi$pi_B > 0
                 & pi$pi_A + pi$pi_B < 1,]
# 2. they sum to one
pi$pi_O <- 1 - (pi$pi_A + pi$pi_B)

# log-likelihood function
# ignoring the multinomial constant
lnL <- function(pi, Y){
  Y[1]*log(pi[1]^2+2*pi[1]*pi[3]) +
  Y[2]*log(pi[2]^2+2*pi[2]*pi[3]) +
  Y[3]*log(2*pi[1]*pi[2]) +
  2*Y[4]*log(pi[3])
}

# observed data
Y <- c(186,38,13,284)

# calculate the log-likelihood for each
# simulated combination of trinomial probabilities
logLk <- apply(pi, 1, function(x) lnL(Y=Y, pi=unlist(x)))
pi$logLikelihood <- logLk

# which values of pi maximize log-likelihood?
pi[pi$logLikelihood == max(pi$logLikelihood), ]
```

```
##     pi_A pi_B pi_O logLikelihood
## 527 0.21 0.05 0.74     -511.6083
```

```r
# contour plot
pi %>%
  ggplot() +
  stat_contour(aes(x = pi_A, y = pi_B, z = logLikelihood,
                   color = ..level..), binwidth=50) +
  labs( color = "Log-Likelihood",
        x = expression(~pi[A]), y = expression(~pi[B]),
        title = "Log-Likelihood Surface Contour Plot")
```




```
# barycentric plot
pi %>%
  ggtern(aes(x = pi_A, y = pi_B, z = pi_O, value = logLikelihood)) +
  geom_interpolate_tern(geom = "polygon", method = "loess",
                        formula = value ~ x+y,
                        aes(color =..level..),
                        show.legend = TRUE,
                        breaks = seq(-5000,100,by=100),
                        expand = 1) +
  labs(fill="log-likelihood", color="log-likelihood",
       title  = "Log-Likelihood Surface Barycentric Plot") +
  Llab(expression(paste(pi[A],",0,0"))) +
  Tlab(expression(paste("0,",pi[B],",0"))) +
  Rlab(expression(paste("0,0,",pi[O])))
```



From the contour plot of the log-likelihood surface, we see that the log-likelihood is maximized for π_A slightly less than .25 and π_B less than .1 but we do not see how this relationship changes in regards to π_O. The Barycentric plot of the log-likelihood surface displays the trinomial log-likelihood surface from 3D cartesian coordinates as barycentric coordinates therevy preserving the original coordinate space. The Barycentric plot gives us the advantage over the contour plot of visualizing all three of the allele frequencies. We clearly see the relationship needed between all three allele frequencies for the log-likelihood to be maximized. Specifically, we see that π_O must be quite large in comparison to the other two allele frequencies. Even though we could have extrapolated this information regarding π_O from the contour plot, it is still a better visual to consider the Barycentric plot where all three allele frequencies are displayed.

Question 3. Derivation of EM algorithm.

Derive the *expectation-maximization algorithm* (EM) for *maximum likelihood estimation* (MLE) of the ABO allele frequencies $\pi = (\pi_A, \pi_B, \pi_O)$, based on ABO phenotype counts from a random sample of n individuals from a well-defined population, with Hardy-Weinberg equilibrium at the ABO locus.

Specifically, define the observed incomplete and unobserved complete data structures, provide the incomplete and complete data log-likelihood functions, supply the main EM Q-function, and derive explicit solutions for the E- and M-steps.

We know from Slide 33 of the EM algorithm notes that the EM algorithm considers the (aggregated) counts for the four phenotypes as observed, incomplete data structure and the incomplete data log-likelihood function is Multinomial$(n, (\pi_A^2 + 2\pi_A\pi_O, \pi_B^2 + 2\pi_B\pi_O, 2\pi_A\pi_B, \pi_O^2))$. So, $Y = (Y_A, Y_B, Y_{AB}, Y_O)$ denoting the ABO phenotype counts for a random sample of n individuals from the population of interest is the observed, incomplete data structure and the likelihood for the incomplete data structure is

$$\mathcal{L}(Y|\pi) = \binom{n}{Y_A, Y_B, Y_{AB}, Y_O} \times \left((\pi_A^2 + 2\pi_A\pi_O)^{Y_A}(\pi_B^2 + 2\pi_B\pi_O)^{Y_B}(2\pi_A\pi_B)^{Y_{AB}}(\pi_O^2)^{Y_O}\right)$$

with log-likelihood

$$l(Y|\pi) = log\Big(\binom{n}{Y_A, Y_B, Y_{AB}, Y_O}\Big) \times \Big(Y_A log(\pi_A^2 + 2\pi_A\pi_O) + Y_B log(\pi_B^2 + 2\pi_B\pi_O) + Y_{AB} log(2\pi_A\pi_B) + 2Y_O log(\pi_O)\Big) \ .$$

We also know from Slide 33 of the EM algorithm notes that the EM algorithm considers the counts for the six unphased genotypes and as the unobserved complete data structure and the the complete data log-likelihood function is Multinomial$(n, (\pi_A^2, 2\pi_A\pi_O, \pi_B^2, 2\pi_B\pi_O, 2\pi_A\pi_B, \pi_O^2))$. So, $X = (X_{AA}, X_{AO}, X_{BB}, X_{BO}, X_{AB}, X_{OO})$ denoting the the six unphased genotypes for a random sample of n individuals from the population of interest is the unobserved, complete data structure and the likelihood for the incomplete data structure is

$$\mathcal{L}(X|\pi) = \Big(\binom{n}{X_{AA}, X_{AO}, X_{BB}, X_{BO}, X_{AB}, X_{OO}}\Big) \times \Big((\pi_A^2)^{X_{AA}} (2\pi_A\pi_O)^{X_{AO}} (\pi_B^2)^{X_{BB}} (2\pi_B\pi_O)^{X_{BO}} (2\pi_A\pi_B)^{X_{AB}} (\pi_O^2)^{X_{OO}}\Big)$$

with log-likelihood

$$l(X|\pi) = log\Big(\binom{n}{X_{AA}, X_{AO}, X_{BB}, X_{BO}, X_{AB}, X_{OO}}\Big) \times \Big(2X_{AA} log(\pi_A) + X_{AO} log(2\pi_A\pi_O) + 2X_{BB} log(\pi_B) + X_{BO} log(2\pi_B\pi_O) + X_{AB} log(2\pi$$

For the initial iteration of the EM algorithm, the E-step calculates $Q(\pi|\pi^0) = E[\updownarrow(X|\pi)|Y, \pi^0]$. So, $\pi^0 = (\pi_A^0, \pi_B^0, \pi_O^0)$, and we want to calculate

$$Q(\pi_A, \pi_B, \pi_O|\pi_A^0, \pi_B^0, \pi_O^0) = 2E[X_{AA}|Y, \pi^0] log(\pi_A) + E[X_{AO}|Y, \pi^0] log(2\pi_A\pi_O) + 2E[X_{BB}|Y, \pi^0] log(\pi_B) + E[X_{BO}|Y, \pi^0] log(2\pi_I$$

where $g(X) = \binom{n}{X_{AA}, X_{AO}, X_{BB}, X_{BO}, X_{AB}, X_{OO}}$ and is not a function of π. Realizing that $Y_A = X_{AA} + X_{AO}$, $Y_B = X_{BB} + X_{BO}$, $Y_{AB} = X_{AB}$, and $Y_O = X_{OO}$ and since $X_{AA}|Y_A \sim Binomial(Y_A, \frac{\pi_A^2}{\pi_A^2 + 2\pi_A\pi_O})$ we have that

$$X_{AA}^0 = E[X_{AA}|Y, \pi^0] = E[X_{AA}|Y_A, \pi^0] = Y_A \frac{(\pi_A^0)^2}{(2\pi_A^0\pi_O^0) + (\pi_A^0)^2} \text{ and}$$

$$X_{AO}^0 = E[X_{AO}|Y, \pi^0] = E[X_{AO}|Y_A, \pi^0] = Y_A \frac{(2\pi_A^0\pi_O^0)}{(2\pi_A^0\pi_O^0) + (\pi_A^0)^2} \ .$$

Similarly,

$$X_{BB}^0 = E[X_{BB}|Y, \pi^0] = E[X_{BB}|Y_B, \pi^0] = Y_B \frac{(\pi_B^0)^2}{(2\pi_B^0\pi_O^0) + (\pi_B^0)^2} \text{ and}$$

$$X_{BO}^0 = E[X_{BO}|Y, \pi^0] = E[X_{BO}|Y_B, \pi^0] = Y_B \frac{(2\pi_B^0\pi_O^0)}{(2\pi_B^0\pi_O^0) + (\pi_B^0)^2} \ .$$

And obviously,

$$E[X_{AB}|Y, \pi^0] = E[X_{AB}|Y_{AB}, \pi^0] = Y_{AB} \text{ and}$$
$$E[X_{OO}|Y, \pi^0] = E[X_{OO}|Y_O, \pi^0] = Y_O \ .$$

For the M-step, we want $\hat{\pi} = (\hat{\pi_A}, \hat{\pi_B}, \hat{\pi_O})$ which involves maximizing Q, the expected value of the log-likelihood (obtained in the E-step) with respect to $\pi = (\pi_A, \pi_B, \pi_O)$. So, we are restricted in that $\pi_A + \pi_B + \pi_O = 1$ and we can introduce the Lagrange multiplier and maximize $Q_L(\pi, \lambda|\pi^0) = Q(\pi|\pi^0) + \lambda(\pi_A + \pi_B + \pi_O - 1)$ with respect to π an λ.

$$\frac{\partial Q_L(\pi,\lambda|\pi^0)}{\partial \pi_A} = \frac{2X^0_{AA}}{\pi_A} + \frac{X^0_{AO}}{\pi_A} + \frac{X_{AB}}{\pi_A} + \lambda$$

$$\frac{\partial Q_L(\pi,\lambda|\pi^0)}{\partial \pi_B} = \frac{2X^0_{BB}}{\pi_B} + \frac{X^0_{BO}}{\pi_B} + \frac{X_{AB}}{\pi_B} + \lambda$$

$$\frac{\partial Q_L(\pi,\lambda|\pi^0)}{\partial \pi_B} = \frac{2X_{OO}}{\pi_O} + \frac{X^0_{BO}}{\pi_O} + \frac{X^0_{AO}}{\pi_O} + \lambda$$

$$\frac{\partial Q_L(\pi,\lambda|\pi^0)}{\partial \lambda} = \pi_A + \pi_B + \pi_O - 1$$

Taking the sum of the three equations, we get $\lambda = -2n$ which yields for the first three equations the solutions the following MLE's:

$$\hat{\pi_A} = \frac{2X^0_{AA} + X^0_{AO} + X_{AB}}{2n}$$

$$\hat{\pi_B} = \frac{2X^0_{BB} + X^0_{BO} + X_{AB}}{2n}$$

$$\hat{\pi_O} = \frac{2X^0_{OO} + X^0_{AO} + X^0_{BO}}{2n} \quad .$$

The next step is to set $\pi^1_A = \hat{\pi_A}$, $\pi^1_B = \hat{\pi_B}$, and $\pi^1_O = \hat{\pi_O}$ then return to the E-step of the algorithm and compute $Q(\pi|\pi^1)$, where $\pi^1 = (\pi^1_A, \pi^1_B, \pi^1_O)$. We continue iterating between the E- and M-steps until the π^i values converge.

Question 4. Software implementation of EM algorithm.

Write an R function implementing the EM algorithm for maximum likelihood estimation of the ABO allele frequencies $\pi = (\pi_A, \pi_B, \pi_O)$.

Arguments to this function should include: the phenotype counts, starting values for the allele frequencies, stopping criteria; it should return candidate MLE for the allele frequencies and the corresponding value of the observed data log-likelihood.

```
EM <- function(pi, Y,  threshold = 1e-7, debug=FALSE){

  # evaluate log-likelihood using initial estimates
  llk <- lnL(pi, Y)

  # count number of iterations so far
  iter <- 1

  # loop until likelihood difference is below threshold
  while(TRUE)
  {
    # E-step
    X_AA = Y[1]*(pi[1]^2 / (pi[1]^2 + 2*pi[1]*pi[3]))
    X_AO = Y[1]*(2*pi[1]*pi[3] / (pi[1]^2 + 2*pi[1]*pi[3]))
```

```
    X_BB = Y[2]*(pi[2]^2 / (pi[2]^2 + 2*pi[2]*pi[3]))
    X_BO = Y[2]*(2*pi[2]*pi[3] / (pi[2]^2 + 2*pi[2]*pi[3]))

    # M-step
    pi_A <- (2*X_AA+X_AO+Y[3]) / (2*(Y[1]+Y[2]+Y[3]+Y[4]))
    pi_B <- (2*X_BB+X_BO+Y[3]) / (2*(Y[1]+Y[2]+Y[3]+Y[4]))
    pi_O <- 1-pi_A-pi_B
    pi <- c(pi_A, pi_B, pi_O)

    # check for convergence
    llk1 <- lnL(pi, Y)

    # if we converge then stop
    if(abs(llk1-llk) < threshold) break
    # otherwise keep iterating
    llk <- llk1
    iter <- iter + 1
  }
  list(pi_A = pi[1], pi_B = pi[2], pi_O = pi[3], llk=llk)
}
```

Question 5. Application of EM algorithm.

Apply the EM algorithm to derive maximum likelihood estimates of the ABO allele frequencies $\pi = (\pi_A, \pi_B, \pi_O)$ for the Clarke et al. (1959) dataset.

Trace the progress of the EM algorithm by providing a table of candidate MLE for the allele frequencies and corresponding values of the observed data log-likelihood at each iteration.

Also provide graphical summaries of these results.

Comment on the EM algorithm's performance in terms of sensitivity to starting values, convergence, and any other features you deem relevant.

Compare the results from your implementation of the EM algorithm to those from one of the R optimization functions (e.g., `optim`).

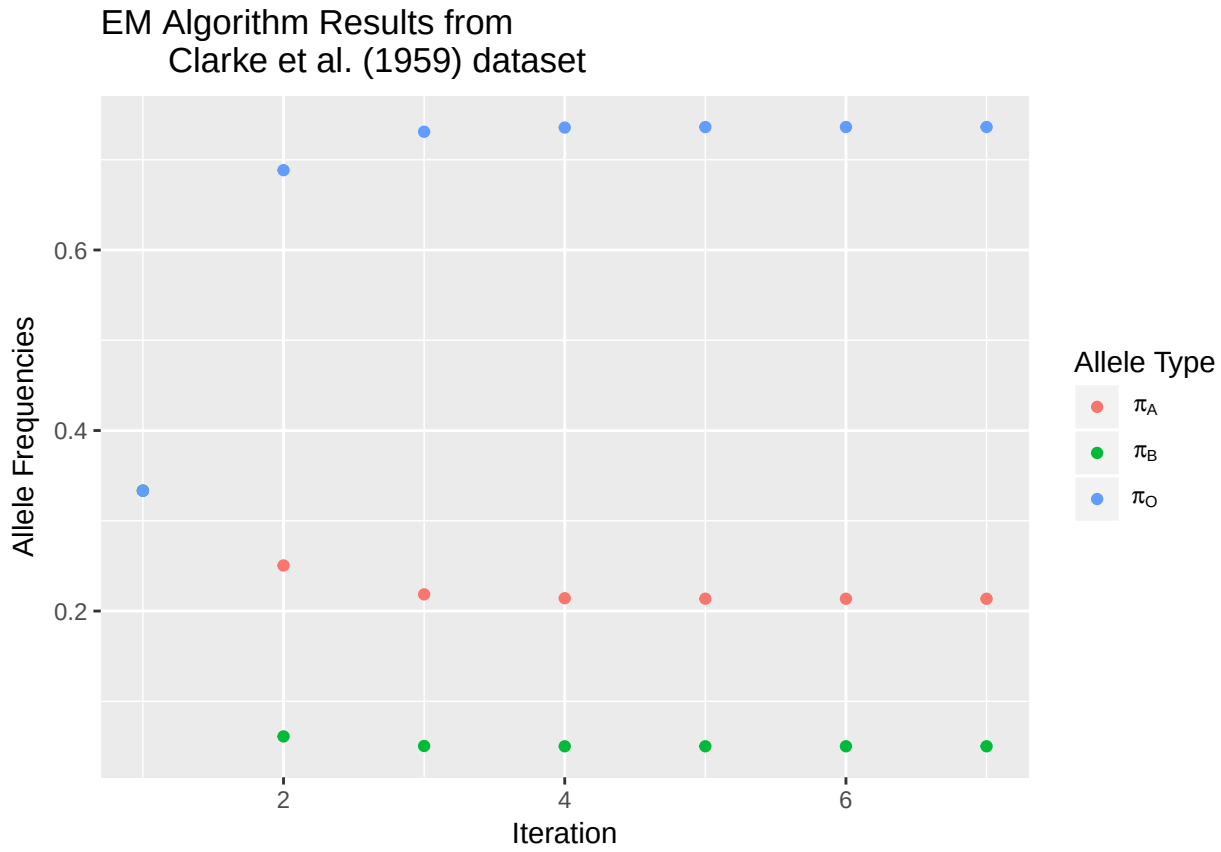
```
# we modify the original EM function because we are
# asked to provide output that differs from the output
# in Question 4

EM <- function(pi, Y, threshold = 1e-7){

  # evaluate log-likelihood using initial estimates
  llk <- lnL(pi, Y)

  # initialize vectors to store at EACH iteration:
  # 1. candidate MLE for allele frequencies
  # 2. values of the observed data log-likelihood
  pi_A_vec <- c()
  pi_B_vec <- c()
  pi_O_vec <- c()
  llk_vec <- c()
  iter_vec <- c()
```

```
  # count number of iterations so far
  iter <- 1

  # loop until likelihood difference is below threshold
  while(TRUE)
  {
    # E-step
    X_AA = Y[1]*(pi[1]^2 / (pi[1]^2 + 2*pi[1]*pi[3]))
    X_AO = Y[1]*(2*pi[1]*pi[3] / (pi[1]^2 + 2*pi[1]*pi[3]))
    X_BB = Y[2]*(pi[2]^2 / (pi[2]^2 + 2*pi[2]*pi[3]))
    X_BO = Y[2]*(2*pi[2]*pi[3] / (pi[2]^2 + 2*pi[2]*pi[3]))

    # populate the vectors
    pi_A_vec <- c(pi_A_vec, pi[1])
    pi_B_vec <- c(pi_B_vec, pi[2])
    pi_O_vec <- c(pi_O_vec, pi[3])
    llk_vec <- c(llk_vec, llk)
    iter_vec <- c(iter_vec, iter)

    # M-step
    pi_A <- (2*X_AA+X_AO+Y[3]) / (2*(Y[1]+Y[2]+Y[3]+Y[4]))
    pi_B <- (2*X_BB+X_BO+Y[3]) / (2*(Y[1]+Y[2]+Y[3]+Y[4]))
    pi_O <- 1-pi_A-pi_B
    pi <- c(pi_A, pi_B, pi_O)

    # check for convergence
    llk1 <- lnL(pi, Y)

    # if we converge then stop
    if(abs(llk1-llk) < threshold) break
    # otherwise keep iterating
    llk <- llk1
    iter <- iter + 1
  }

  # combine vectors to form a table
  df <- data.frame(iteration = iter_vec,
                   pi_A = pi_A_vec,
                   pi_B = pi_B_vec,
                   pi_O = pi_O_vec,
                   llk = llk_vec)
  EM_results <<- df
  return(df)
}

# apply EM function to Clarke et al. (1959) dataset
EM(pi = c(1/3, 1/3, 1/3), Y = c(186,38,13,284))
```

```
##   iteration      pi_A       pi_B      pi_O       llk
## 1         1 0.3333333 0.33333333 0.3333333 -889.6539
## 2         2 0.2504798 0.06110045 0.6884197 -516.7319
## 3         3 0.2184544 0.05049394 0.7310517 -511.6397
## 4         4 0.2141823 0.05016173 0.7356559 -511.5725
```

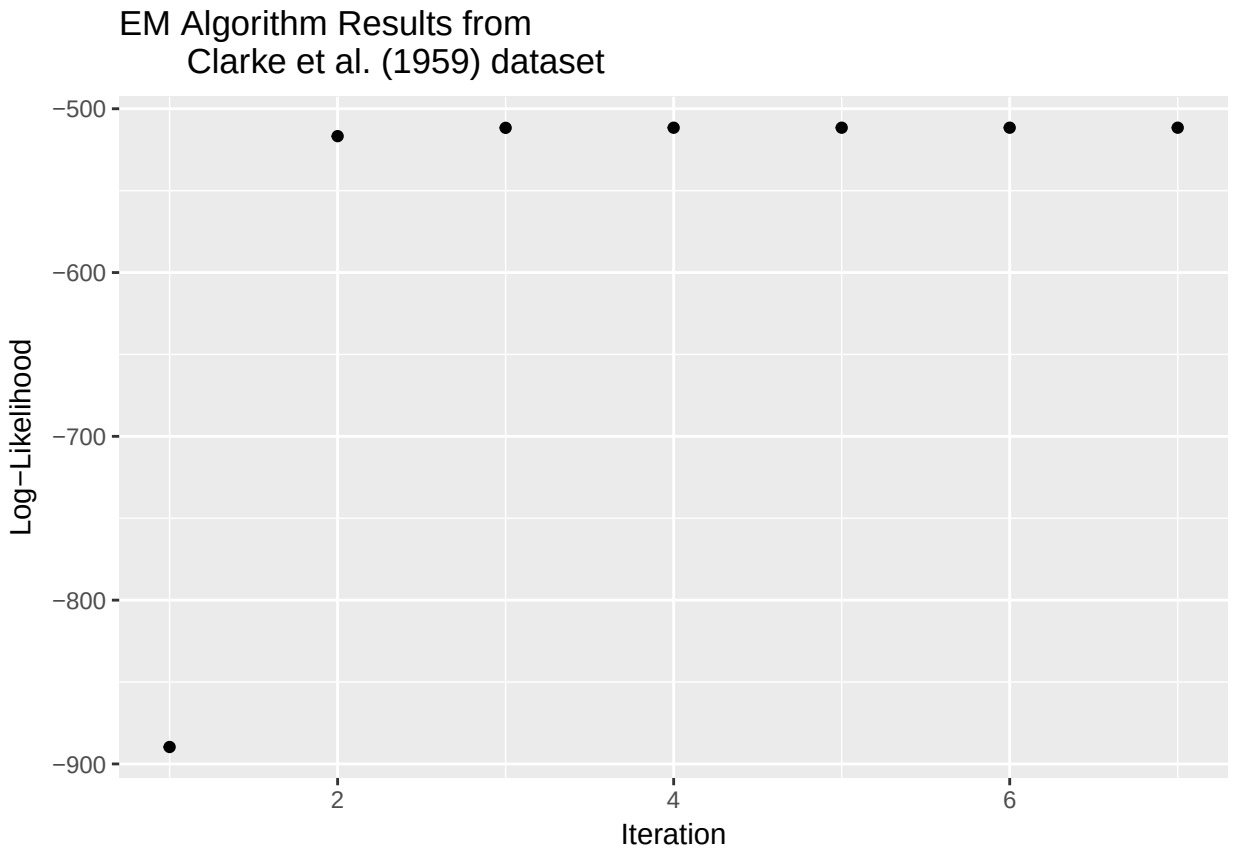
```
## 5         5 0.2136619 0.05014667 0.7361914 -511.5715
## 6         6 0.2135994 0.05014547 0.7362551 -511.5715
## 7         7 0.2135920 0.05014535 0.7362627 -511.5715
```

```
####################################
# graphical summaries of EM results
####################################

# plot of the allele frequencies
EM_melt <- melt(EM_results, id.vars="iteration", measure.vars=c("pi_A","pi_B","pi_O"))
EM_melt %>%
  ggplot(aes(x = iteration, y = value)) +
  geom_point(aes(color = variable)) +
  guides(color=guide_legend("Allele Type"),
         linetype = guide_legend("Allele Type")) +
  scale_color_discrete(labels = c(expression(~pi[A]),
                                  expression(~pi[B]),
                                  expression(~pi[O]))) +
  labs(title="EM Algorithm Results from
       Clarke et al. (1959) dataset",
       y = "Allele Frequencies", x = "Iteration")
```




```
# plot of the log-likelihood
EM_melt2 <- melt(EM_results, id.vars="iteration", measure.vars="llk")
EM_melt2 %>%
  ggplot(aes(x = iteration, y = value))+
  geom_point()+
  labs(title = "EM Algorithm Results from
```

```
       Clarke et al. (1959) dataset",
       x = "Iteration", y = "Log-Likelihood")
```

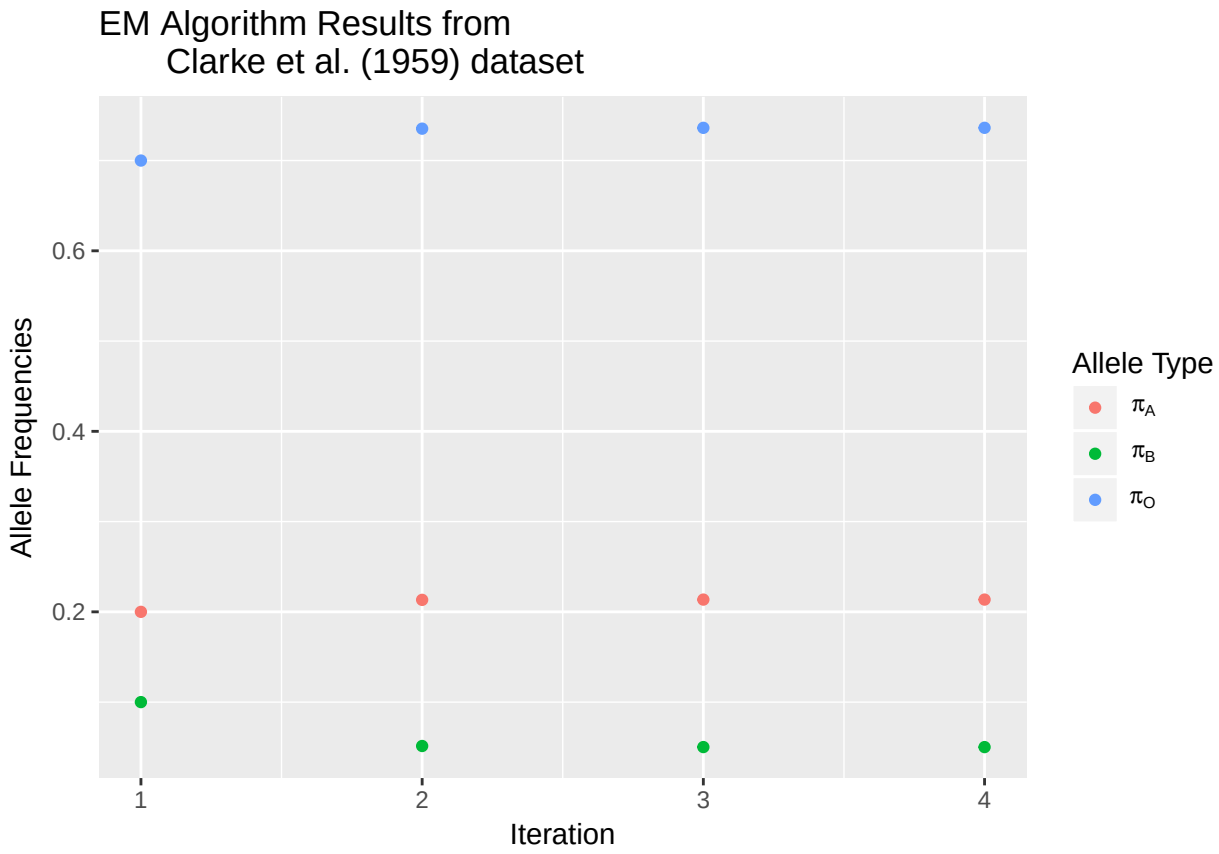



```
##############################
# EM algorithm's performance
##############################

# 1. to starting allele frequencies that are closer to the MLEs
EM(pi = c(.2, .1, .7), Y = c(186,38,13,284))
```

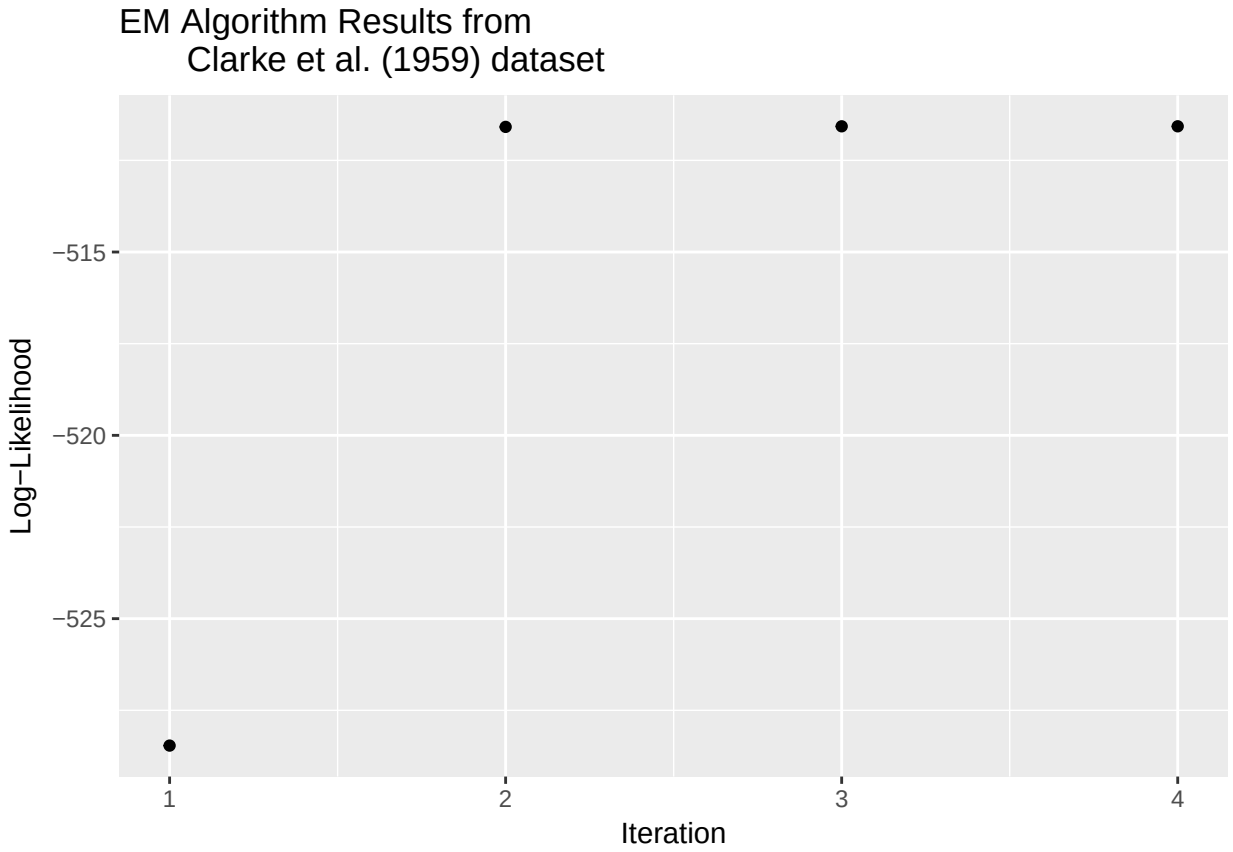
```
##   iteration      pi_A       pi_B      pi_O       llk
## 1         1 0.2000000 0.10000000 0.7000000 -528.4621
## 2         2 0.2132917 0.05137556 0.7353327 -511.5874
## 3         3 0.2135882 0.05017530 0.7362365 -511.5715
## 4         4 0.2135914 0.05014607 0.7362625 -511.5715
```

```
EM_melt <- melt(EM_results, id.vars="iteration", measure.vars=c("pi_A","pi_B","pi_O"))
EM_melt %>%
  ggplot(aes(x = iteration, y = value)) +
  geom_point(aes(color = variable)) +
  guides(color=guide_legend("Allele Type"),
         linetype = guide_legend("Allele Type")) +
  scale_color_discrete(labels = c(expression(~pi[A]),
                                  expression(~pi[B]),
                                  expression(~pi[O]))) +
  labs(title="EM Algorithm Results from
       Clarke et al. (1959) dataset",
```

```
      y = "Allele Frequencies", x = "Iteration")
```




```
EM_melt2 <- melt(EM_results, id.vars="iteration", measure.vars="llk")
EM_melt2 %>%
  ggplot(aes(x = iteration, y = value))+
  geom_point()+
  labs(title = "EM Algorithm Results from
       Clarke et al. (1959) dataset",
       x = "Iteration", y = "Log-Likelihood")
```




```
# 2. to varying thresholds of convergence
EM(pi = c(.2, .1, .7), Y = c(186,38,13,284), threshold = 1e-12)
```

```
##   iteration      pi_A       pi_B      pi_O       llk
## 1         1 0.2000000 0.10000000 0.7000000 -528.4621
## 2         2 0.2132917 0.05137556 0.7353327 -511.5874
## 3         3 0.2135882 0.05017530 0.7362365 -511.5715
## 4         4 0.2135914 0.05014607 0.7362625 -511.5715
## 5         5 0.2135910 0.05014535 0.7362636 -511.5715
## 6         6 0.2135909 0.05014533 0.7362637 -511.5715
```

```
EM(pi = c(1/3, 1/3, 1/3), Y = c(186,38,13,284), threshold = 1e-12)
```

```
##   iteration      pi_A       pi_B      pi_O       llk
## 1         1 0.3333333 0.33333333 0.3333333 -889.6539
## 2         2 0.2504798 0.06110045 0.6884197 -516.7319
## 3         3 0.2184544 0.05049394 0.7310517 -511.6397
## 4         4 0.2141823 0.05016173 0.7356559 -511.5725
## 5         5 0.2136619 0.05014667 0.7361914 -511.5715
## 6         6 0.2135994 0.05014547 0.7362551 -511.5715
## 7         7 0.2135920 0.05014535 0.7362627 -511.5715
## 8         8 0.2135911 0.05014533 0.7362636 -511.5715
## 9         9 0.2135910 0.05014533 0.7362637 -511.5715
```

```
##########################################
# comparison to R optimization functions
##########################################
```

```r
# by default, optim performs mimization so
# we use the negative log-likelihood here
pi <- c(1/3, 1/3, 1/3)
optim(pi[1:2], fn = function(x) - lnL(Y = c(186,38,13,284), pi=c(x[1], x[2], 1-x[1]-x[2])))
```

```
## $par
## [1] 0.21357043 0.05011656
##
## $value
## [1] 511.5715
##
## $counts
## function gradient
##       71       NA
##
## $convergence
## [1] 0
##
## $message
## NULL
```

In terms of sensitivity to starting values and without changing the threshold for convergence, we see that the EM algorithm's performance is affected in the number of iterations. The iterations do decrease if we approximate the maximum likelihood estimators more closely. We also note that as the threshold for convergence decreases the difference in output from later iterations also decreases. These relationships are expected but it is interesting to visualize them.

When we compare the results from the intial implementation of the EM algorithm to those from the R optimization function we see that the maximum likelihood estimators for the allele frequencies ($par) are almost identical. This small difference is due to the R optimization function converging to zero and the EM algorithm set to converge to a very small value that's close to zero, "pseudo-ish-convergence". (Note: I tried setting my threshold to zero for the EM algorithm and R Studio wasn't happy). Also, if the R optimization iteration limit of 10,000 was exceeded, a value of 1 would have be reported for the convergence in the output. Thus, we could probably acheive identical results to the R optimization function if we set an iteration limit to 10,000 in the EM algorithm function. The log-likelihood ($value) is identical for both methods. There were 71 total calls to the log-likelihood function and the gradient ($counts) and this code ran very quickly. I am certain that 71 iterations of my EM alogrithm would run much slower. In conclusion, the R optimization function does perform better than my mediocre EM algorithm implementation but I am no Wizard of R, it was still a fun exercise!

Collaborators & References

Tommy Carpenito

University of Washington Statistical Methods in Genetic Epidemiology Course Notes: The EM Algorithm (http://courses.washington.edu/b516/lectures_2009/EM_ALGORITHM.pdf)

University of Chicago Statistical Theory and Methods III Course Handouts: Allele Frequency Estimation (http://galton.uchicago.edu/~eichler/stat24600/Handouts/s04.pdf)

Stack Exchange Question: Conditional Probability of Multinomial Distribution (https://math.stackexchange.com/questions/1162493/conditional-probability-of-multinomial-distribution)

UMass Amherst Analysis of Environmental Data Course Notes: Conceptual Foundations, Maximum Likelihood Inference (https://www.umass.edu/landeco/teaching/ecodata/schedule/likelihood.pdf)