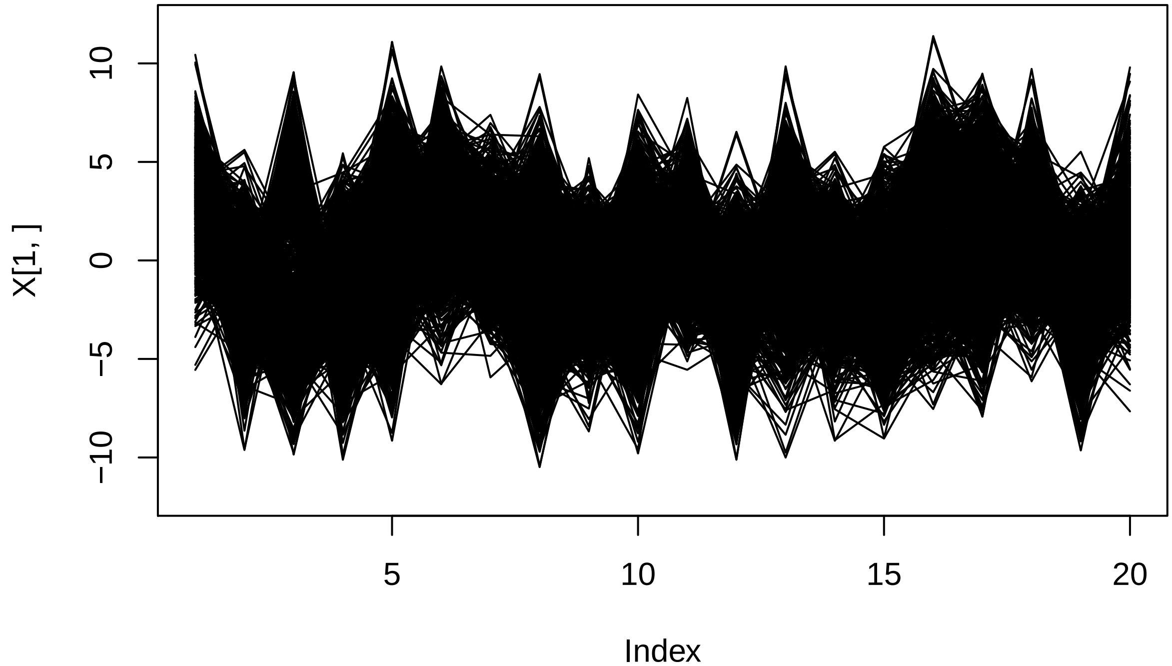
HW5 Yan Liu

10/5/2019

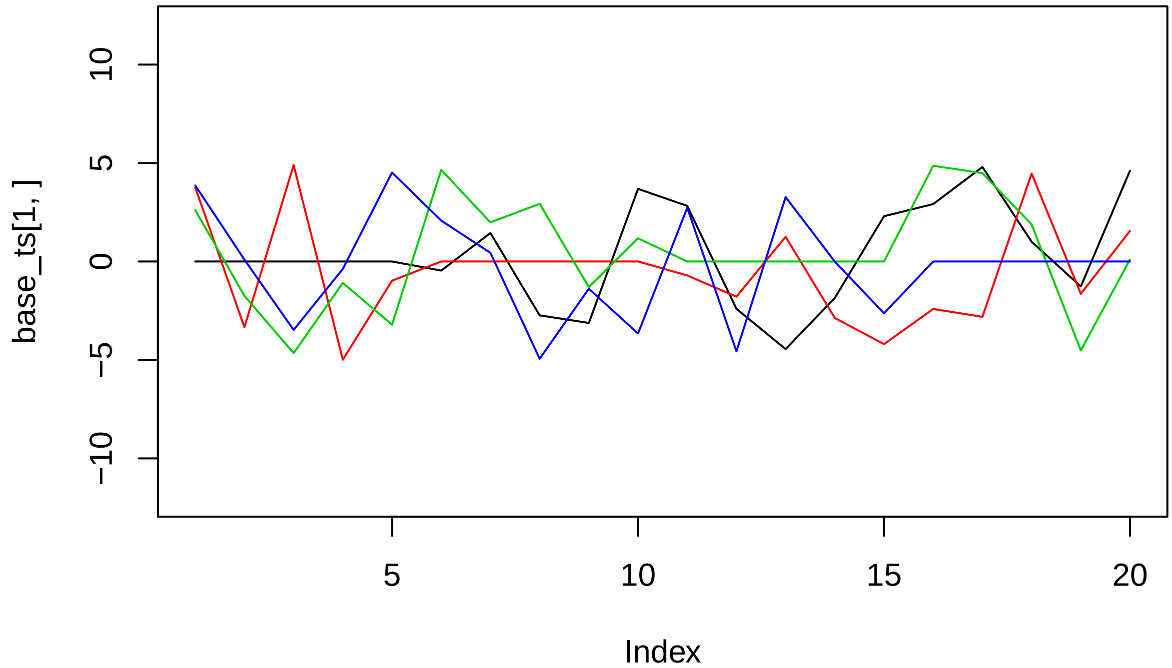
Mixed Time Series and true mean Value

```
setwd("/Users/ly/Desktop/Math\ 611\ @Sivan/Homework/HW5")
X <-  as.matrix(read.csv("TimeSeries.csv"))
plot(X[1,], type="l", ylim=c(-12,12))
      for (i in 2:1000) { lines(X[i,])}
```




```
set.seed(12345)
n <- 20
K <- 4
N <- 1000
# create base time series
base_ts <- matrix(runif(K*n, min=-5, max=5), nrow=K, ncol=n)
# tweek a bit to make them easy to differentiate visually
base_ts[1,1:5] <- 0
base_ts[2,6:10] <- 0
base_ts[3,11:15] <- 0
base_ts[4,16:20] <- 0
# choose a base time series for each sample
assign_prob <- (1:K)/sum(1:K)
sample_assignment <- sample.int(K, N, replace = T, prob = assign_prob)
samples <- base_ts[sample_assignment,]
# add noise
samples <- samples + matrix(rnorm(K*N, mean=0, sd=2), nrow=N, ncol=n)
# plot base time series
plot(base_ts[1,], type="l", ylim=c(-12, 12), col = 1)
for (i in 2:K)
```

```r
  lines(base_ts[i,], col = i)
```




```r
library(mvtnorm)
```

```
## Warning: package 'mvtnorm' was built under R version 3.5.2
```

```r
library(magrittr)
```

```r
logL <- function(theta, X)
{
  mu1 <- theta$mu1
  mu2 <- theta$mu2
  mu3 <- theta$mu3
  mu4 <- theta$mu4
  p1 <- theta$p1
  p2 <- theta$p2
  p3 <- theta$p3
  p4 <- 1 - p1-p2-p3
  Sigma1 <- theta$Sigma1
  Sigma2 <- theta$Sigma2
  Sigma3 <- theta$Sigma3
  Sigma4 <- theta$Sigma4

  N1 <- dmvnorm(X, mean = mu1, sigma = Sigma1)
  N2 <- dmvnorm(X, mean = mu2, sigma = Sigma2)
  N3 <- dmvnorm(X, mean = mu3, sigma = Sigma3)
  N4 <- dmvnorm(X, mean = mu4, sigma = Sigma4)

  ll <- log(p1*N1 + p2*N2 + p3*N3 + p4*N4) %>%  sum

  return (ll)
}
```

```r
Sigma_update <- function (r, X, new_mu){
  Sigma_list <- list()
  for (i in 1:nrow(X)){
   Sigma_list  <- append(Sigma_list, list(r[i] * (X[i,] - new_mu) %*% t(X[i,] - new_mu)))
  }
  return (Reduce('+', Sigma_list)/sum(r))
}

Shift_Sigma <- function(Sigma, epsilon){
  return(Sigma + diag(x = epsilon, nrow=nrow(Sigma), ncol = ncol(Sigma)))
}

EM_step <- function(theta, X, epsilon, mode = "soft")
  {

  mu1 <- theta$mu1
  mu2 <- theta$mu2
  mu3 <- theta$mu3
  mu4 <- theta$mu4

  Sigma1 <- theta$Sigma1
  Sigma2 <- theta$Sigma2
  Sigma3 <- theta$Sigma3
  Sigma4 <- theta$Sigma4

  p1 <- theta$p1
  p2 <- theta$p2
  p3 <- theta$p3
  p4 <- 1-p1-p2-p3

  N1 <- dmvnorm(X, mean = mu1, sigma = Sigma1)
  N2 <- dmvnorm(X, mean = mu2, sigma = Sigma2)
  N3 <- dmvnorm(X, mean = mu3, sigma = Sigma3)
  N4 <- dmvnorm(X, mean = mu4, sigma = Sigma4)

  N <- nrow(X)

  probX <- p1*N1 + p2*N2 + p3*N3 + p4*N4

  r1 <- p1*N1/probX
  r2 <- p2*N2/probX
  r3 <- p3*N3/probX
  r4 <- 1-r1-r2-r3

 if (mode == "hard"){
    r_matrix <- matrix(cbind(r1,r2,r3,r4), nrow = length(r1), ncol = 4)
    group <- apply(r_matrix, 1, which.max)

    X1 <- matrix(X[group == 1,], ncol = ncol(X), byrow = T)
    X2 <- matrix( X[group == 2,], ncol = ncol(X),byrow = T)
    X3 <- matrix(X[group == 3,],ncol = ncol(X),byrow = T)
    X4 <- matrix(X[group == 4,], ncol = ncol(X),byrow = T)
```

```r
    if (length(X1) > 0){
      new_mu1 <- colSums(X1)/nrow(X1)
      new_Sigma1 <- t(X1 - new_mu1) %*% (X1 - new_mu1)  %>% Shift_Sigma(epsilon)
      new_p1 <- nrow(X1)/N
    }
    else{
      new_mu1 <- mu1
      new_Sigma1 <- Sigma1
      new_p1 <- 0
    }

    if (length(X2) > 0){
          new_mu2 <- colSums(X2)/nrow(X2)
          new_Sigma2 <- t(X2 - new_mu2) %*% (X2 - new_mu2)  %>% Shift_Sigma(epsilon)
          new_p2 <- nrow(X2)/N
    } else{
      new_mu2 <- mu2
      new_Sigma2 <- Sigma2
      new_p2 <- 0
    }
    if(length(X3) > 0){
        new_mu3 <- colSums(X3)/nrow(X3)
        new_Sigma3 <- t(X3 - new_mu3) %*% (X3 - new_mu3)  %>% Shift_Sigma(epsilon)
        new_p3 <- nrow(X3)/N
    }else{
      new_mu3 <- mu3
      new_Sigma3 <- Sigma3
      new_p3 <- 0
    }

    if(length(X4) > 0){
        new_mu4 <- colSums(X4)/nrow(X4)
        new_Sigma4 <- t(X4 - new_mu4) %*% (X4 - new_mu4)  %>% Shift_Sigma(epsilon)
        new_p4 <- nrow(X4)/N
    } else{
      new_mu4 <- mu4
      new_Sigma4 <- Sigma4
      new_p4 <- 0
    }

}
else{
  # Soft Update
  sum_r1 <- sum(r1)
  sum_r2 <- sum(r2)
  sum_r3 <- sum(r3)
  sum_r4 <- sum(r4)

  new_mu1 <- t(r1 %*% X)/sum_r1
  new_mu2 <- t(r2 %*% X)/sum_r2
  new_mu3 <- t(r3 %*% X)/sum_r3
  new_mu4 <- t(r4 %*% X)/sum_r4
```

```r
  new_Sigma1 <- Sigma_update(r1, X, new_mu1) %>% Shift_Sigma(epsilon)
  new_Sigma2 <- Sigma_update(r2, X, new_mu2) %>% Shift_Sigma(epsilon)
  new_Sigma3 <- Sigma_update(r3, X, new_mu3) %>% Shift_Sigma(epsilon)
  new_Sigma4 <- Sigma_update(r4, X, new_mu4) %>% Shift_Sigma(epsilon)

  new_p1 <- sum(r1)/N
  new_p2 <- sum(r2)/N
  new_p3 <- sum(r3)/N
  new_p4 <- sum(r4)/N
   }

  return (list(mu1=new_mu1, mu2=new_mu2,
               mu3=new_mu3, mu4=new_mu4,
               Sigma1=new_Sigma1,Sigma2=new_Sigma2,
               Sigma3=new_Sigma3,Sigma4=new_Sigma4,
               p1=new_p1,p2=new_p2,p3=new_p3))
}
```

```r
EM <- function(theta, X, mode, epsilon, iterations=1000, debug=T)
{
  for (i in 1:iterations) {
    if (debug)
      cat("likelihood=", logL(theta, X), "\n")
    theta <- EM_step(theta, X, epsilon, mode = mode)
  }

  return (theta)
}
```

Test if likelihood increases

```r
# select a random theta
start_theta <- list(mu1=rep(1, ncol(X)), mu2=rep(1, ncol(X)),
                    mu3=rep(1, ncol(X)), mu4=rep(1, ncol(X)),
                    Sigma1 = diag(ncol(X)), Sigma2 = diag(ncol(X)),
                    Sigma3 = diag(ncol(X)), Sigma4 = diag(ncol(X)),
                    p1=.25,
                    p2 = .25,
                    p3= .3
                    )
# test to see that likelihood is increasing
test_theta <- EM(start_theta, X, mode = "hard", epsilon =.01 ,iterations=10, debug=T)
```

```
## likelihood= -138292.1
## likelihood= -109542.8
## likelihood= -109542.8
## likelihood= -109542.8
## likelihood= -109542.8
## likelihood= -109542.8
## likelihood= -109542.8
## likelihood= -109542.8
## likelihood= -109542.8
```

```
## likelihood= -109542.8
```

Likelihood increases for the hard EM algorithm.

```r
test_theta <- EM(start_theta, X, mode = "soft", epsilon =.01 ,iterations=10, debug=T)
```

```
## likelihood= -138292.1
## likelihood= -2458.639
## likelihood= -2458.639
## likelihood= -2458.639
## likelihood= -2458.639
## likelihood= -2458.639
## likelihood= -2458.639
## likelihood= -2458.639
## likelihood= -2458.639
## likelihood= -2458.639
```

Likelihood increases for the soft EM algorithm.

Random Initialization and Pick the Best Theta

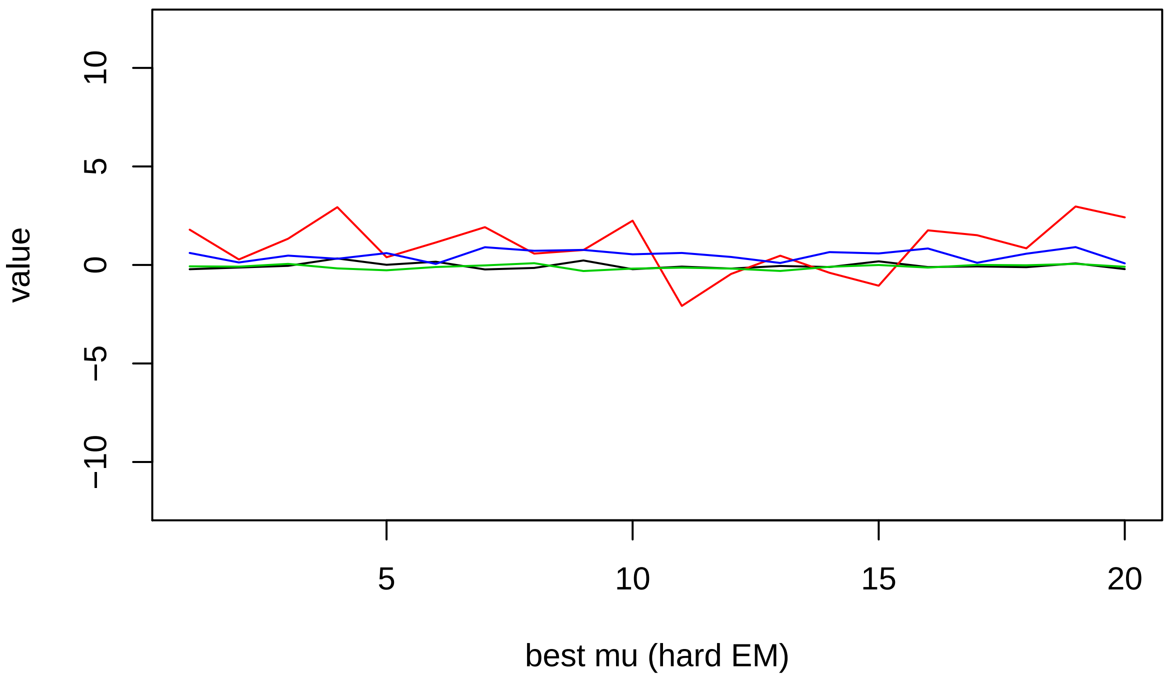
```r
random_init <- function(mode, N_starts){
  likelihoods <- rep(NA, N_starts) %>% as.numeric
  save_thetas <- list()
  for (i in 1:N_starts) {
    start_theta <- list(mu1=runif(ncol(X)), mu2=runif(ncol(X)),
                      mu3=runif(ncol(X)), mu4=runif(ncol(X)),
                      Sigma1 = diag(x = runif(5), ncol(X)),
                      Sigma2 = diag(x = runif(5), ncol(X)),
                      Sigma3 = diag(x = runif(5), ncol(X)),
                      Sigma4 = diag(x = runif(5), ncol(X)),
                      p1=runif(1, max = .3),
                      p2 = runif(1,max=.3),
                      p3= runif(1,max=.3)
                      )
    final_theta <- EM(start_theta, X, mode = mode, epsilon =.01 ,iterations=10, debug=F)
    likelihoods[i] <- logL(theta = final_theta, X=X)
    save_thetas <- append(save_thetas, list(final_theta))
  }
  return(list(likelihoods = likelihoods, save_thetas = save_thetas))
}

pick_best_theta <- function(likelihoods, save_thetas){
  # find the solutions that don't explode or go bad
  good_ind <- !is.nan(likelihoods) & !is.infinite(likelihoods)
  likelihoods <- likelihoods[good_ind]
  save_thetas <- save_thetas[good_ind]
  # print maximum likelihood with best theta
  best_theta <- save_thetas[[which.max(likelihoods)]]
  #print(best_theta)
  cat("Maximum Likelihood:", logL(best_theta, X))
```

```r
  return(best_theta)
}
plot_mu <- function(best_theta, xlab){
  plot(best_theta$mu1, type="l", xlab = xlab, ylab = "value", ylim=c(-12, 12), col = 1)
  lines(best_theta$mu2, col = 2)
  lines(best_theta$mu3, col = 3)
  lines(best_theta$mu4, col = 4)
}
```

Hard EM Result Plot

```r
set.seed(1024)
N_starts <- 100
random_init(mode = "hard", N_starts = N_starts) %>%
  {pick_best_theta(.$likelihoods, .$save_thetas)} %>%
    plot_mu(xlab = "best mu (hard EM)")
```

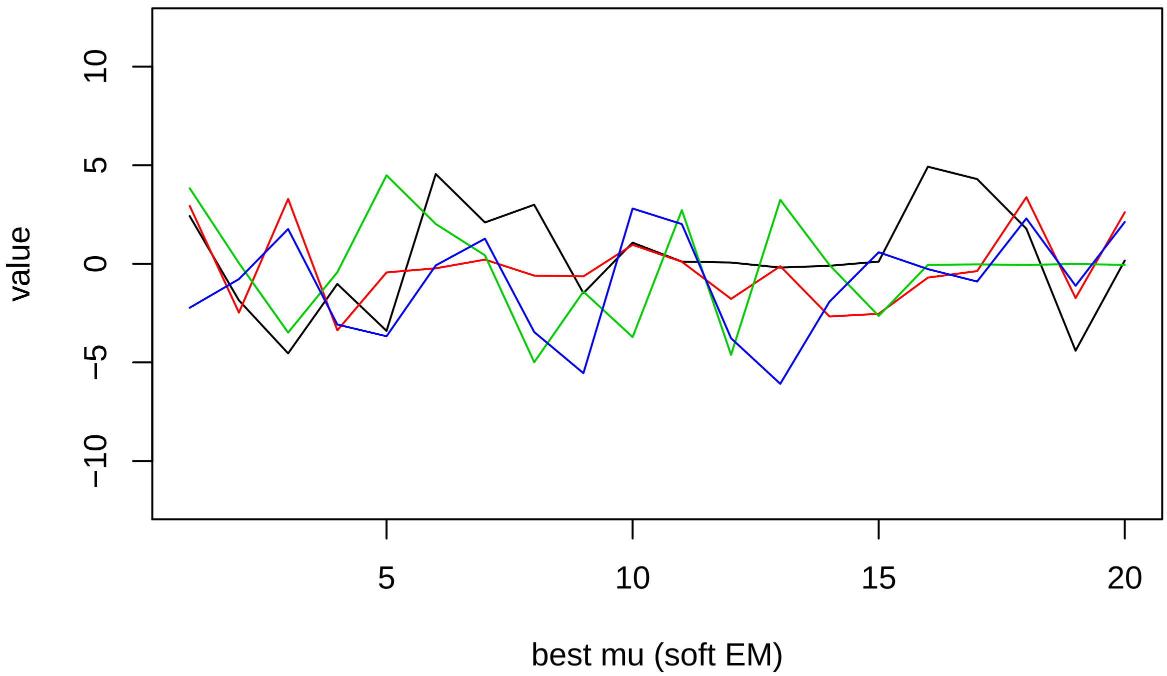
```
## Maximum Likelihood: -109542.8
```



Soft EM Result Plot

```r
set.seed(1024)
random_init(mode = "soft", N_starts = N_starts) %>%
  {pick_best_theta(.$likelihoods, .$save_thetas)} %>%
    plot_mu(xlab = "best mu (soft EM)")
```

```
## Maximum Likelihood: 7986.955
```



Hard EM algorithm fails to recover the underlying time series. But soft EM algorithm is able to find at least 3 underlying time series! From the plot above, we could tell that red line is close to 0 between 5 and 10; black line is close to 0 between 10 and 15; green line is close to 0 between 15 and 20 (though none of them is close to 0 between 0 and 5). This might owes to the fact that the first underlying time series pattern has only 10% samples and they might not be enough to recover.