Charles Zheng EE 378b HW 5

==

Setup

```
library(knitr)
opts_knit$set(root.dir=normalizePath('../'))
```

Load the data

```
load('ee378b/ratings.RData')
dim(ratings)
```

```
## [1] 100000      4
```

```
n <- dim(ratings)[1]
```

Form training and test sets

```
set.seed(0)
train_inds <- sample(n, .8 * n)
train <- ratings[train_inds, ]
test <- ratings[-train_inds, ]
ntr <- dim(train)[1]
nte <- dim(test)[1]
```

Form training matrix

```
n_u <- max(ratings$user)
n_i <- max(ratings$item)
trmat <- matrix(NA, n_u, n_i)
trmat[cbind(train$user, train$item)] <- train$rating
```

1 Use mean of ratings

Using movie mean

```
rmse <- function(y1, y2) sqrt(sum((y1-y2)^2)/length(y1))
means_i <- colMeans(trmat, na.rm = TRUE)
means_i[is.na(means_i)] <- mean(train$rating)
pr_ratings_i <- means_i[test$item]
rmse_mean_movie <- rmse(pr_ratings_i, test$rating)
rmse_mean_movie
```

```
## [1] 1.024848
```

Using user mean

```r
means_u <- rowMeans(trmat, na.rm = TRUE)
pr_ratings_u <- means_u[test$user]
pr_ratings_u[is.na(pr_ratings_u)] <- mean(train$rating)
rmse_mean_user <- rmse(pr_ratings_u, test$rating)
rmse_mean_user
```

```
## [1] 1.040713
```

2 Use SVD

Center training matrix by means

```r
trmat_c <- t(t(trmat) - means_i)
trmat_c[is.na(trmat_c)] <- 0
library(rARPACK)
res_svd <- svds(trmat_c, k = 10)
pr_k10 <- res_svd$u %*% diag(res_svd$d) %*% t(res_svd$v)
dim(pr_k10)
```

```
## [1]  943 1682
```

```r
adj_k10 <- pr_k10[cbind(test$user, test$item)]
test[1, ]
```

```
##    user item rating timestamp
## 14  210   40      3 891035994
```

```r
pr_k10[244, 51]
```

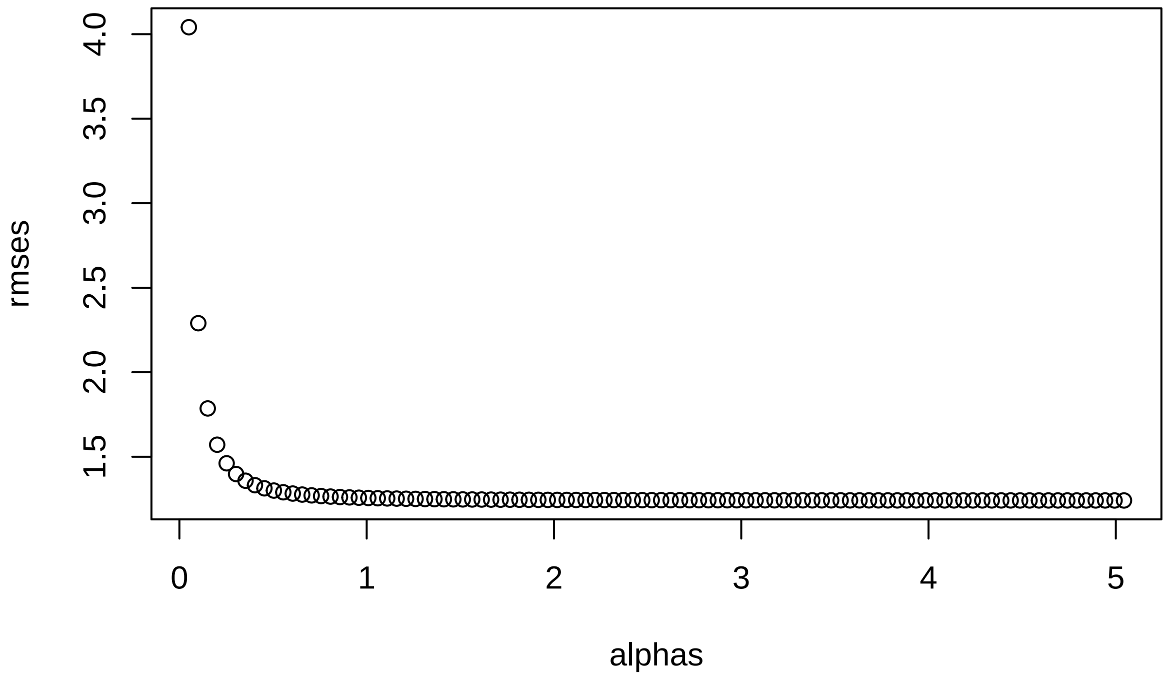
```
## [1] -0.08633954
```

```r
adj_k10[1]
```

```
## [1] -0.02934012
```

Determine the best α

```r
alpha_min <- ntr/(n_u * n_i)
alphas <- alpha_min * (1 + 0:99/1)
rmses <- numeric(100)
for (i in 1:100) rmses[i] <- rmse(train$rating, pr_ratings_i  + 1/alphas[i] * adj_k10)
plot(alphas, rmses)
```



The best $\alpha = \infty$, meaning it is better to not to use the SVD at all.

4 Alternating Least Squares

Track which users have which items, etc.

```
uitems <- lapply(1:n_u, function(i) which(!is.na(trmat[i, ])))
iusers <- lapply(1:n_i, function(i) which(!is.na(trmat[, i])))
u_n <- sapply(uitems, length)
i_n <- sapply(iusers, length)
uratings <- lapply(1:n_u, function(i) trmat[i, !is.na(trmat[i, ])])
iratings <- lapply(1:n_i, function(i) trmat[!is.na(trmat[, i]), i])
```

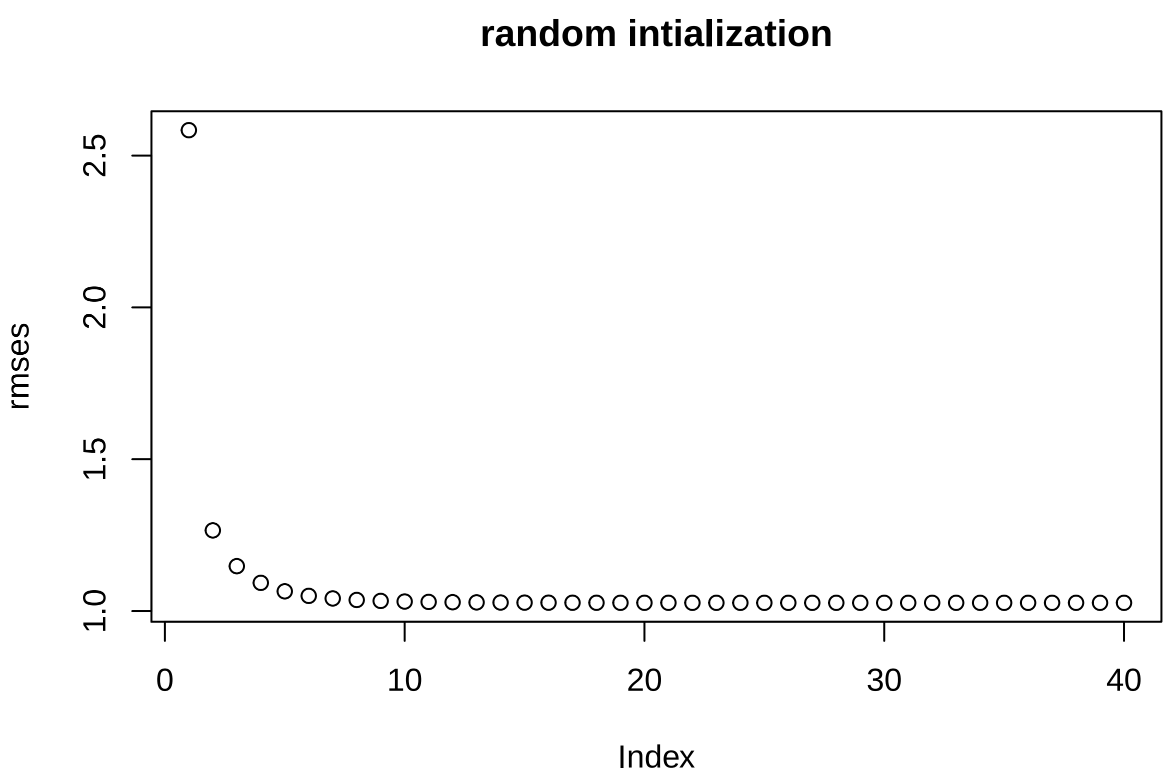
Functions for updating x and y

```
update_x <- function(ymat, lambda) {
  xnew <- xmat
  for (i in unique(train$user)) {
    regmat <- ymat[uitems[[i]], , drop = FALSE]
    xnew[i, ] <- solve(t(regmat) %*% regmat + lambda * diag(rep(1, dim(xmat)[2])),
                      t(regmat) %*% uratings[[i]])
  }
  xnew
}
update_y <- function(xmat, lambda) {
  ynew <- ymat
  for (i in unique(train$item)) {
    regmat <- xmat[iusers[[i]], , drop = FALSE]
    ynew[i, ] <- solve(t(regmat) %*% regmat + lambda * diag(rep(1, dim(ymat)[2])),
                      t(regmat) %*% iratings[[i]])
```

```
  }
  ynew
}
```

Intialize factor matrices randomly, apply ALS with $\lambda = 20$.

```
lambda <- 20
xmat <- matrix(rnorm(n_u * 10), n_u, 10)
ymat <- matrix(rnorm(n_i * 10), n_i, 10)
rmses <- numeric(40)
for (i in 1:40) {
  xmat <- update_x(ymat, lambda)
  ymat <- update_y(xmat, lambda)
  prmat <- xmat %*% t(ymat)
  prtest <- prmat[cbind(test$user, test$item)]
  rmses[i] <- rmse(test$rating, prtest)
}
plot(rmses, main = "random intialization")
```



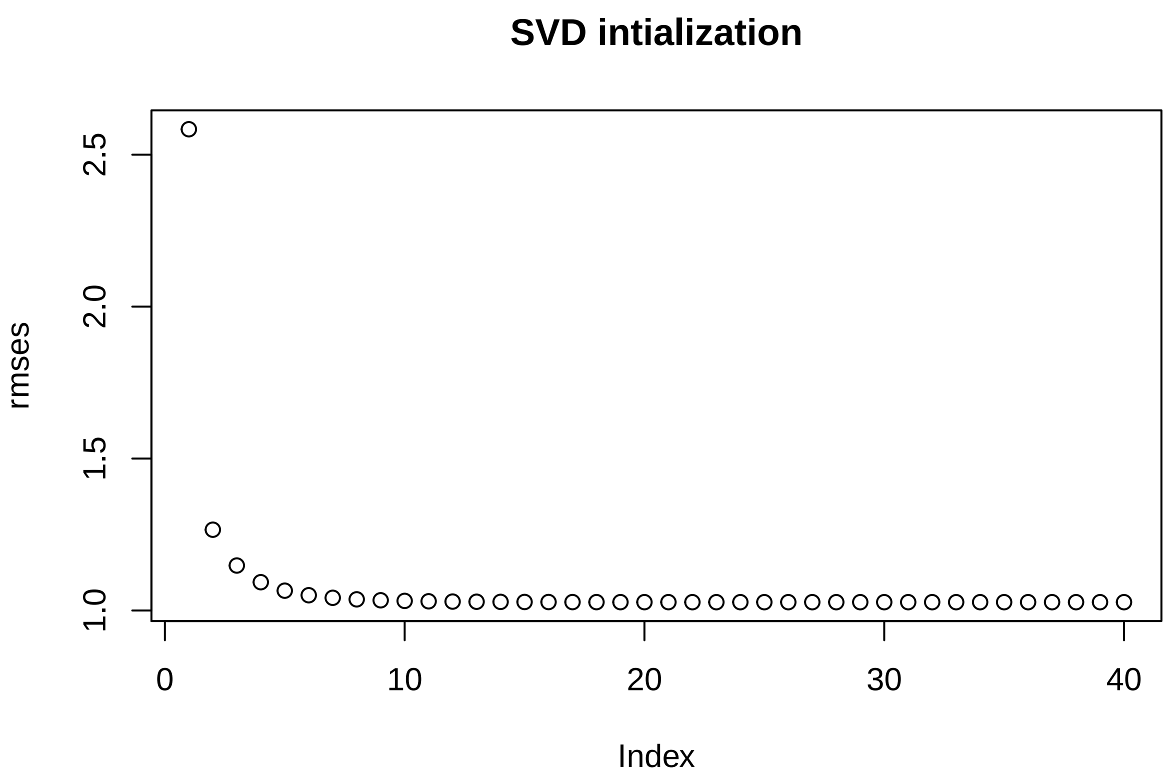

```
data.frame(iterations =  c(5, 10, 20, 40), rmses = rmses[c(5, 10, 20, 40)])
```

```
##   iterations    rmses
## 1          5 1.065160
## 2         10 1.031569
## 3         20 1.027457
## 4         40 1.027401
```

4 initialization with SVD

Use $X = [1, U_9]$ and $Y = [\mu, V_9]$ where μ are the movie means.

```
plot(rmses, main = "SVD intialization")
```



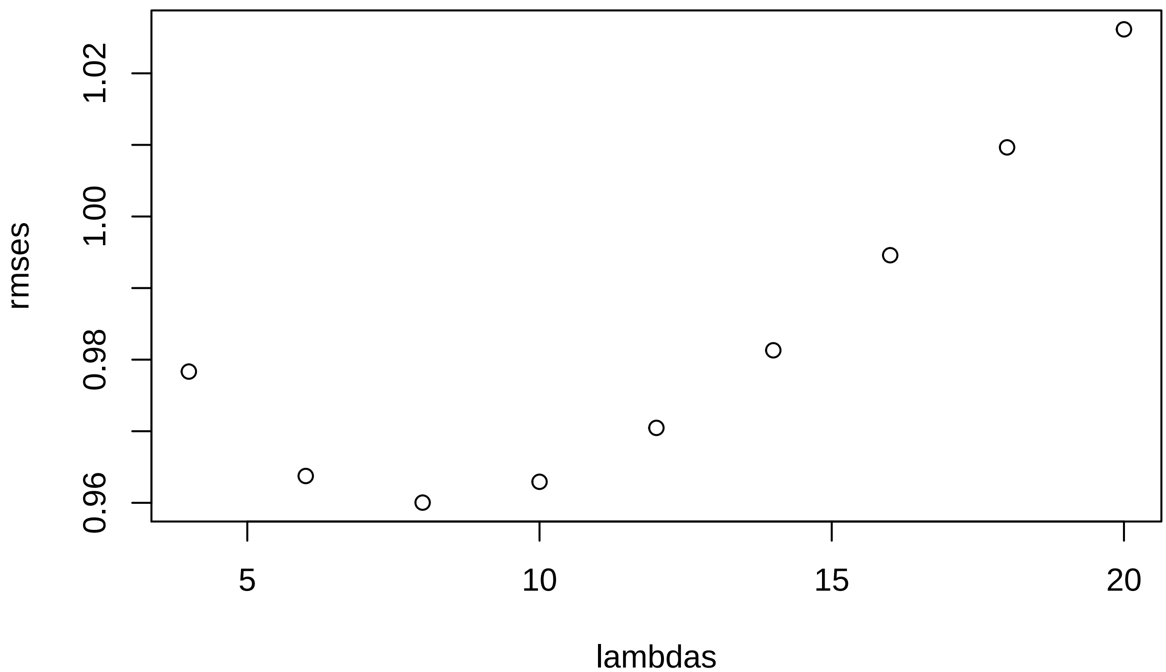

```
data.frame(iterations =  c(5, 10, 20, 40), rmses = rmses[c(5, 10, 20, 40)])
```

```
##   iterations    rmses
## 1          5 1.065160
## 2         10 1.031569
## 3         20 1.027457
## 4         40 1.027401
```

5 Optimal lambda

```
lambdas <- c(4, 6, 8, 10, 12, 14, 16, 18, 20)
rmses <- numeric(length(lambdas))
for (j in 1:length(lambdas)) {
  xmat <- cbind(1, res_svd$u[, 1:9])
  ymat <- cbind(means_i, res_svd$v[, 1:9])
  for (i in 1:40) {
    xmat <- update_x(ymat, lambdas[j])
    ymat <- update_y(xmat, lambdas[j])
  }
```

```
  prmat <- xmat %*% t(ymat)
  prtest <- prmat[cbind(test$user, test$item)]
  rmses[j] <- rmse(test$rating, prtest)
}
plot(lambdas, rmses)
```




```
data.frame(lambdas = lambdas, rmses = rmses)
```

```
##   lambdas     rmses
## 1       4 0.9783330
## 2       6 0.9637467
## 3       8 0.9600300
## 4      10 0.9629380
## 5      12 0.9704678
## 6      14 0.9813074
## 7      16 0.9945885
## 8      18 1.0096549
## 9      20 1.0261435
```

The best $\lambda = 8$.